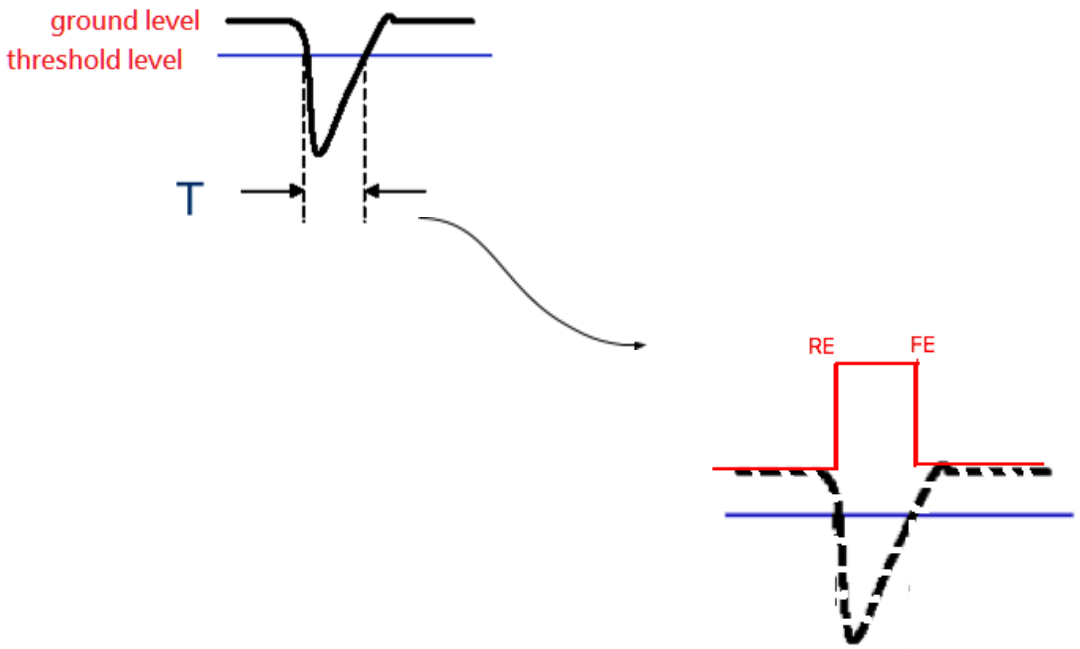
Quarknet Manual

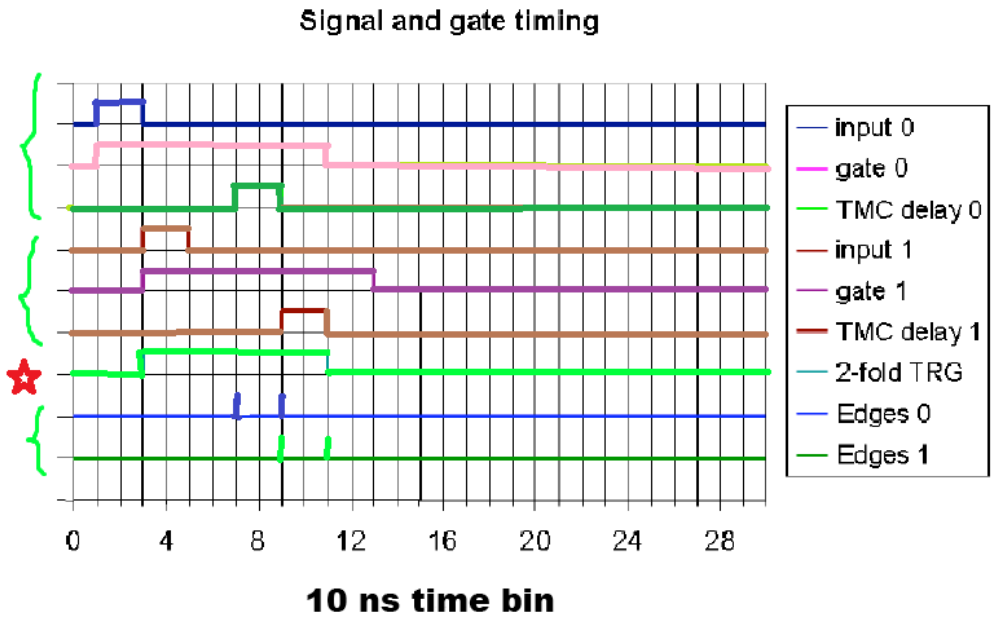
CONTENT:

Quarknet Experiments

QuarkNet設計了一套宇宙射線探測器(Cosmic Ray Detector)，主要元件有以下幾項：1個GPS；1支溫度計；4支閃爍體計數器(Scintillator Counter)；1片數據攫取卡(Data Acquisition Card, DAQ 卡)；1個電源供應器。
閃爍體計數器的訊號送進 DAQ 卡後，會經過閥值(threshold level)篩檢再數位化送往電腦，其過程如下圖所示 ：



訊號的峰值如果沒有超過閥值, 會被視為雜訊而不會被數位化做後續之處理. 數位化後的訊號，是一個包含起落邊界(rising edge, RE 與 falling edge, FE)的方波, 在DAQ卡上再做延遲(d值)及開一個對應的窗(w值)的處理，這個步驟對同時性的判讀是很重要的，請參考下圖。



下面四個實驗需要依照順序進行；第一及第二的目的是校正(calibration)閃爍體計數器，這樣在做第三及第四實驗時，就知道如何讓儀器保持在最佳之狀態了。 這四個實驗是：

(一)設定每一支閃爍體計數器threshold level；

(二)平坦區(Plateau)---- 閃爍體計數器的最佳工作電壓值；

(三)宇宙射線望遠鏡: 測量不同時間或不同方向過來的宇宙射線muon的通量(flux)；

(四)測muon的生命期(life time)。

其他可以考慮的實驗有:

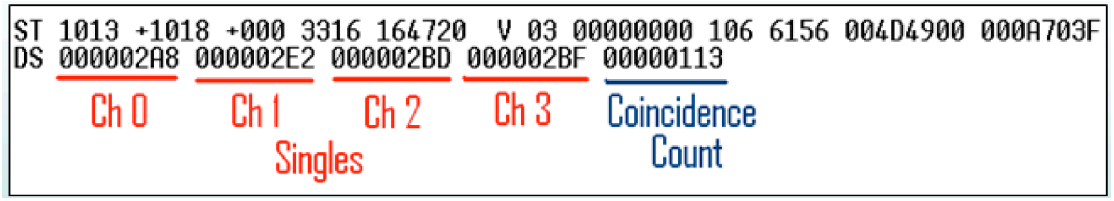
- 通過detector的muon的速度;
- 在不同環境及氣候條件下測宇宙射線的通量變化；
- 不同的高度的通量值，可得知狹義相對論的time dilation效應；
- 廣域的宇宙射線的分布(例如cosmic ray shower)等。

實驗一：設定閃爍體計數器的threshold level(TL)

(1) 先將閃爍體計數器接上DAQ板，調整電源供應器給一個合理的工作電壓(例如0.7V)，如果四支閃爍體計數器都接在DAQ板上，就要下指令 WC 00 0F (1+2+4+8=15=F)告訴DAQ板有那幾支是工作的計數器；如果有三支閃爍體計數器接在DAQ板上，就下指令 WC 00 07(1+2+4=7)，其他依此類推，然後使用TL指令將四支計數器的threshold leve(TL)設到某定值，例如

TL 4 100

是將四支的TL值都設到100mV。接著下RB的指令將計數器的讀數歸零，5分鐘後下DS的指令可得下圖的輸出，記錄其中的數字。



圖中的前四個字串分別對應四支**閃爍體計數器16進位的記數值**，化成十進位除一除時間得到singles counting rate(counts/min)，記下對應的TL值。繼續改變TL指令的參數值改變TL值，當TL值增加時，讀數會減少，當變化要趨緩時(例如下圖0.5V處)，以後的實驗裡就使用這個值為threshold level，如果要更有效地減少雜訊，用稍大的值也可以。

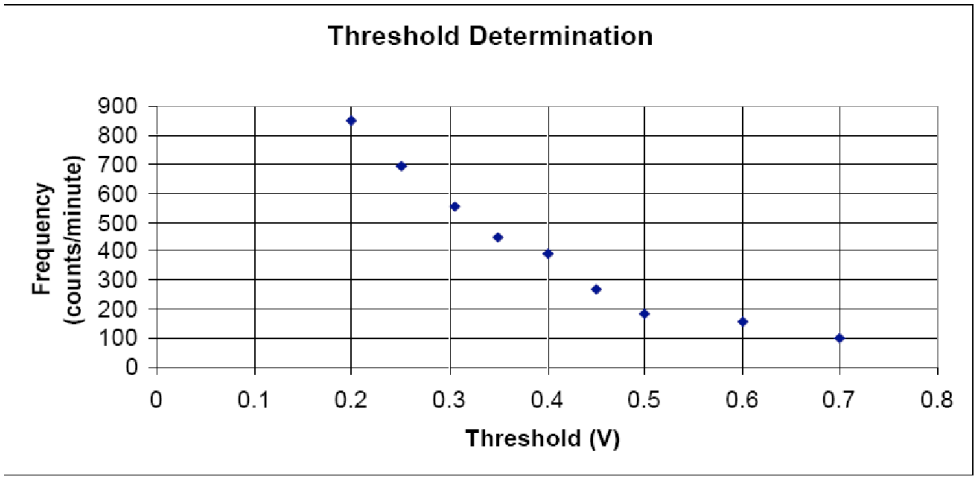
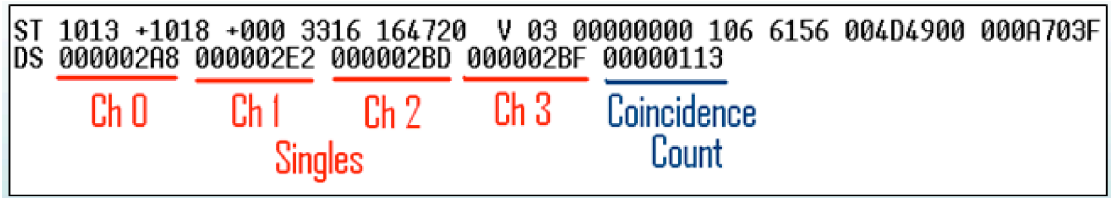


Figure 80. A graph of frequency vs. threshold for a counter used to determine threshold setting.

實驗二：平坦區(Plateau)---- 閃爍體計數器的最佳工作電壓值

用上一個實驗測得的threshold level，以下面兩種方式進行:

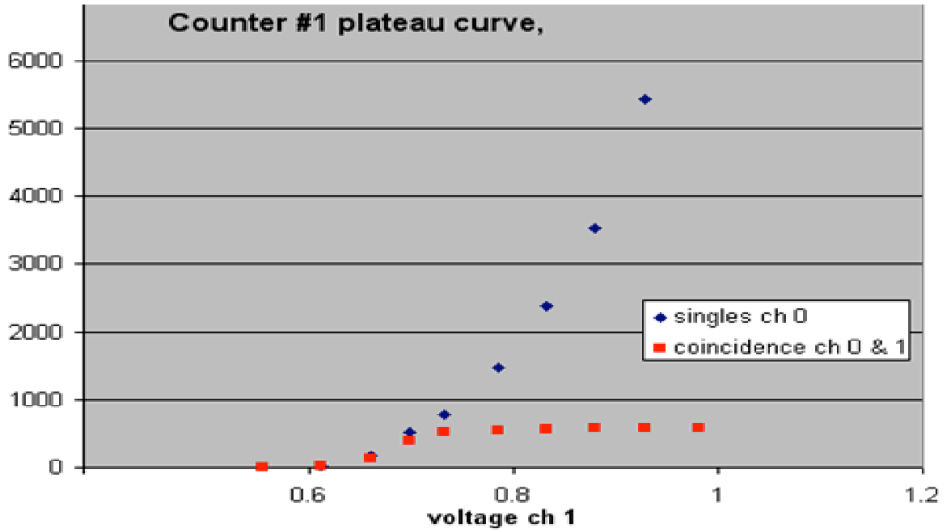
(1)一類是用單一支的**閃爍體計數器**，測singles counting rate (每分鐘的counts)。如果四支**閃爍體計數器都接**在DAQ板上，下指令 WC 00 0F(1+2+4+8=15=F)；如果有三支**閃爍體計數器接**在DAQ板上，下指令 WC 00 07(1+2+4=7)，調整電源供應器上的電壓，下RB的指令， 5分鐘後記錄DS指令所給的數目字， 例如



前四個字串分別對應四支**閃爍體計數器的訊號數目(16進位)**，化成十進位除一除時間得到singles counting rate(counts/min)，並記下電壓。分幾次調整電壓進行量測，依此類推。

(2)第二類是用兩支疊放在一起的**閃爍體計數器**，測double coincidence counting rate(雙重性同時)，先選一支做參考，例如channel 2(Ch2)， 將電壓調到大約 singles counting rate 為2400/分鐘(40Hz)， 然後再選一支為待測的**閃爍體計數器**，例如channel 0(Ch0)，下指令 WC 00 15(1+4=5)，然後調整Ch0的電壓到定值，下指令 RB將讀數歸零，5分鐘後記下DAQ上顯示的讀數(或以DS指令取得第1及第5個數目字)，分別除以5可得Ch0的singles counting rate(counts/min)及Ch0與Ch2的 double coincidence counting rate(counts/min)，同時記下Ch0的電壓，分幾次調整Ch0電壓進行量測，依此類推。

將以上兩種方法得到的數據做圖，如果出現相對於電壓變化的平坦區，取其中央地段的電壓值，但是singles counting rate不要超過200Hz，就是適合這支(Ch0)閃爍體計數器**工作的電壓了**。照這些步驟，每支閃爍體計數器都做一遍。



實驗數據

Figure 32. Graph showing counts per minute of multiple counter plateau using counter 0 as a background and sweeping voltage on counter 1.

(2011/1/10)

A. **實驗一(Threshold Level)**

針對標記Yellow、Green級Red的三支計數器，做的Threshold level如下表，並從附圖得到三支都用500mV(550mV?)為TL值的結論。

TL(V=700mV)	y-singles	g-singles	r-singles
200	4556	4431	4368
250	3695	3573	3504
300	3143	3033	2951
350	2634	2508	2434
400	2253	2126	2053
450	1930	1799	1716
500	1662	1534	1457
550	1488	1362	1295
600	1306	1192	1127

650	1240	1126	1053
700	1154	1041	970
750	1124	997	924
800	1052	927	859

//

/

/

B.實驗二(working Voltage)

針對標記Yellow及Green的兩支計數器(TL=500mV)，做的Plateau如下二表，並從附圖得到Yellow用730mV及Green用750mV為工作電壓的結論。其中參考計數器的工作電壓取了700mV及750mV兩個值，由實驗數據得知，此值的選取，只要在合理範圍內，對double的計數值影響不大，因此呈現同樣的適合工作電壓值。

	y-singles	g-singles(700mV)	2-fold	y-singles	g-singles(750mV)	2-fold
600	114	1138	77	67	2423	48
650	526	1136	348	408	2430	301
675	911	1119	418	701	2335	383
700	1509	1083	403	1198	2456	438
725	2370	1140	417	2017	2413	439
750	3404	1198	461	2978	2445	438
775	4723	1137	446	4055	2399	427
800	6606	1180	456	5668	2349	446
825	10149	1164	463	8886	2437	510
850	15794	1114	450	14134	2400	533
900	81642	1095	449	49689	2449	541
1000	19952903	1176	521			

// /

	y-singles	g-singles(700mV)	2-fold	y-singles	g-singles(750mV)	2-fold
600	1557	88	64	3111	59	46
650	1526	436	298	3084	367	257
675	1583	696	406	3127	568	357
700	1587	1135	443	2857	836	380
725	1648	1862	484	2962	1429	402

750	1519	2831	449	3003	2339	420
775	1538	4059	460	3075	3416	467
800	1421	6181	453	3032	5535	469
825	1570	13473	497	3043	11711	525
850	1515	35991	459	3123	36823	527
900	1559	1296181	485	2950	2441161	559
950	1634	32009345	656			
1000	1600	185664775	1053			

//

Quarknet DAQ卡指令

終端機程式的設定

使用windows的超及終端機程式HyperTerminal，DAQ卡連線的設定是

Bits per Second: nnnnnnn

Data Bits: 8

Parity: None

Stop Bits: 1

Flow Control: None

指令說明

(1) **指令 WT 01 nn，指令 WT 02 nn，指令 WC 02 nn，及 指令 WC 03 nn**

請參考測生命期實驗的說明

(2) **指令 WC 01 nn**

設定veto的width=nn

(三) **指令WC 00 nn**

代表write control， 一般用在要做同時性(coincidence)測量的時候。指令後接著是2位元的十六進位數，要換成二進位的控制碼，一共有8位元 編號是7 6 5 4 3 2 1 0 每一個的意義如下表所示。

7	*6*	*5*	*4*	*3*	*2*	*1*	*0*
選擇做為否決(Veto)訊號的探測器輸入端編碼 =00 不使用否決訊號=01 否決為#1 =10否決為#2 =11否決為#3 注意:#1不能做否決		設定同時的多重性 =00 單一測量 =01 兩重的同時 =10 三重的同時 =11 四重的同時		編號3 啟動	編號2 啟動	編號1 啟動	編號0 啟動

例如，使用全部的探測器，但是做的是單一測量時，要的二進位值是00001111，指令就是

WC 00 0F

如果用#0 #1做同時測量，二進位值就是 00010011，指令是

WC 00 13

(四) **指令 DC，可以顯示使用中的控制碼。**

(五) **指令SS**，忽略所有在單一探測器裡的單一事件，只顯示在單一探測器裡發生雙重同時事件的事件，**指令ES** 恢復SS停掉的功能，也記錄單一探測器裡的單一事件。

(六) **其它指令請參考quarknet的手冊。**

(七) 準備好要進行數據的攫取時，要先resetDAQ卡，然後下以下指令串：

H1, H2, DG, DC, DS, DT, BA, TH, TI,
V1, V2, ST 2 5, SA 1

這樣相關的資料就會寫入檔案裡，請參考下面操作電腦的說明。
(八)要結束取數據時，直接停止攫取就好，然後下CD的指令。
(九)注意：依實驗性質的不同，要用不同的指令，例如

TL 4 500
WC 00 3F

```
[V1
  Run Mode       : Off          CE (cnt enable), CD (cnt disable)
  Ch(s) Enabled  : 3,2,1,0      Cmd DC  Reg C0 using (bits 3-0)
  Veto Enable    : Off
  Veto Select    : Ch0          Cmd DC  Reg C0 using (bits 7,6)
  Coincidence 1-4: 2-Fold       Cmd DC  Reg C0 using (bits 5,4)
  Pipe Line Delay:    40 nS     Cmd DT  Reg T1=rDelay  Reg T2=wDelay  10nS/cnt
  Gate Width     :   100 nS     Cmd DC  Reg C2=LowByte Reg C3=HighByte 10nS/cnt
  Ch0 Threshold  : 0.600 vlts
  Ch1 Threshold  : 0.600 vlts
  Ch2 Threshold  : 0.600 vlts
  Ch3 Threshold  : 0.600 vlts
  Test Pulser Vlt: 3.000 vlts
  Test Pulse Ena : Off

  Example line for 1 of 4 channels. (Line Drawing, Not to Scale)
  Input Pulse edges (begin/end) set rising/falling tags bits.
  ____^^^^^^_____________________________________ Input Pulse, Gate cycle begins
  ___________________~____________________________ Delayed Rise Edge 'RE' Tag Bit
  ________________________~_____________________ Delayed Fall Edge 'FE' Tag Bit
     ______________                               Tag Bits delayed by PipeLnDly
  ___|              |_____________________________ PipeLineDelay :   40nS
                    _________________________
  ___________________|                         |___ Capture Window:   60nS
     ____________________________________
  ___|                                       |___ Gate Width    :  100nS

  If 'RE','FE' are outside Capture Window, data tag bit(s) will be missing.
  CaptureWindow = GateWidth - PipeLineDelay
  The default Pipe Line Delay is 40nS, default Gate Width is 100nS.
  Setup CMD sequence for Pipeline Delay.  CD,  WT 1 0, WT 2 nn (10nS/cnt)
  Setup CMD sequence for Gate Width.  CD, WC 2 nn(10nS/cnt), WC 3 nn (2.56uS/cnt)
```

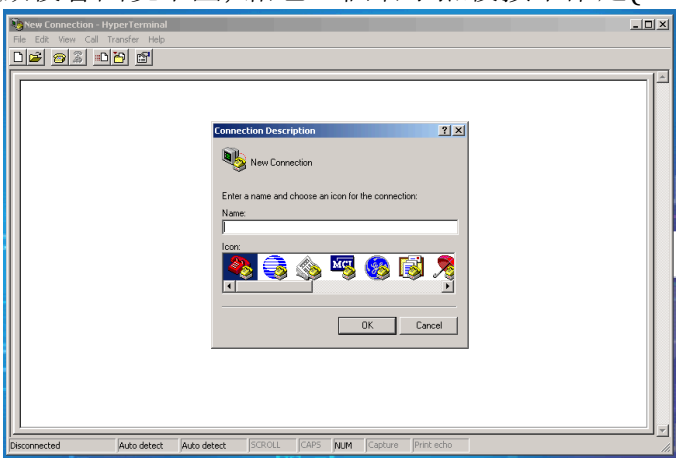
QuarkNet System電腦操作取數據的步驟

Edit by Bei-Zhen

1. 開啟超級終端機(Hyper Terminal)

路徑: 開始→程式集→附屬應用程式→通訊→超級終端機
(Start→Programs→Accessories→Communications→Hyper Terminal)

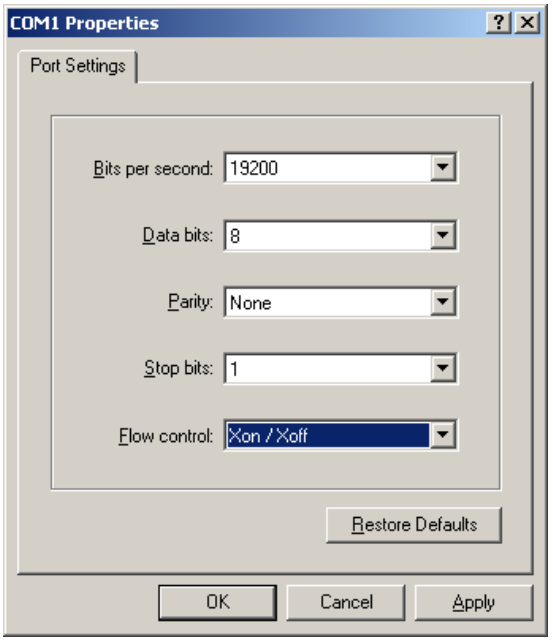
開啟後會出現下圖, 給它一個名字然後按下確定(OK).



2. 設定 (接續1., 按下確定後)

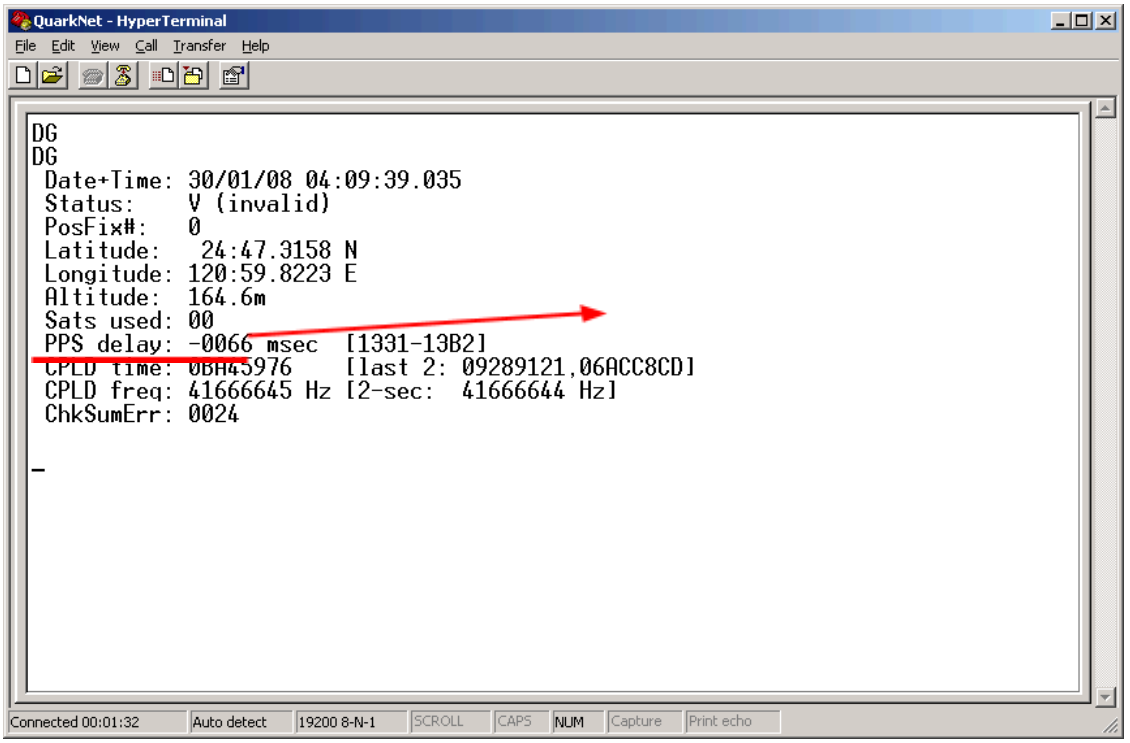
a) "***connect using*** "設定為 " **COM1** "(或其他的com埠)

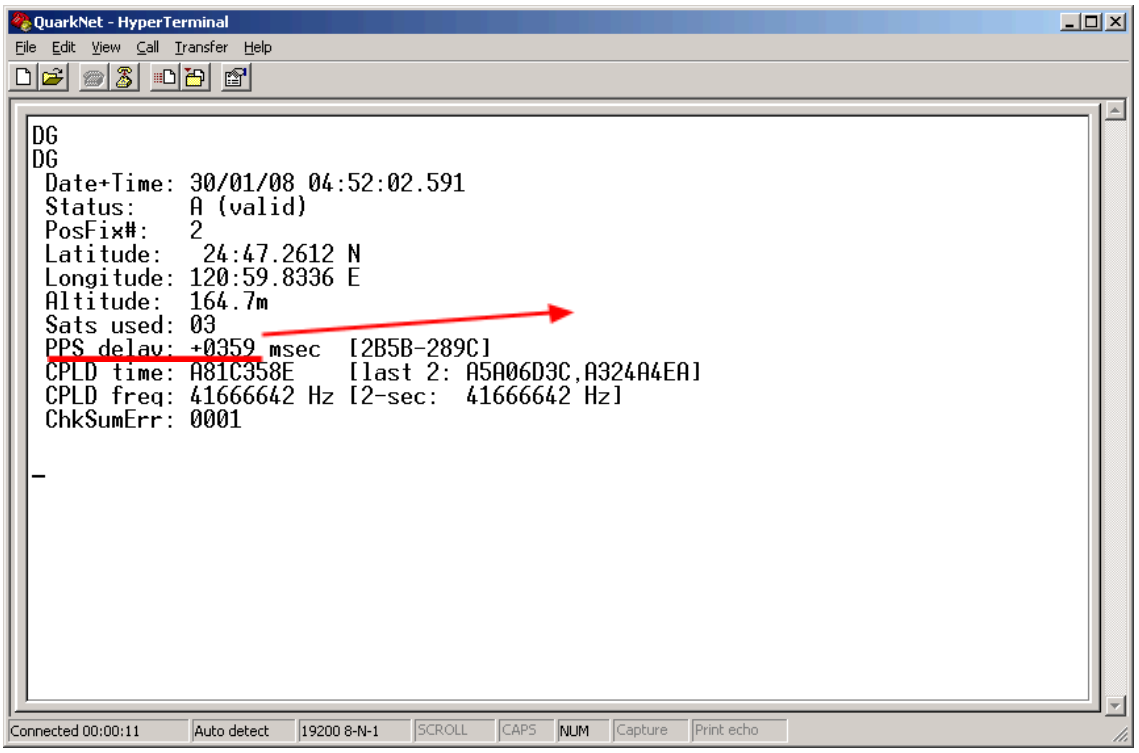
b) ***"Bits per second"*** 設定為 " **115200** "
"Flow control" 設定為 " **Xon/Xoff** "



3. 開始取數

a) 先看看**GPS**訊號, 輸入 " **DG** " (要加Enter)

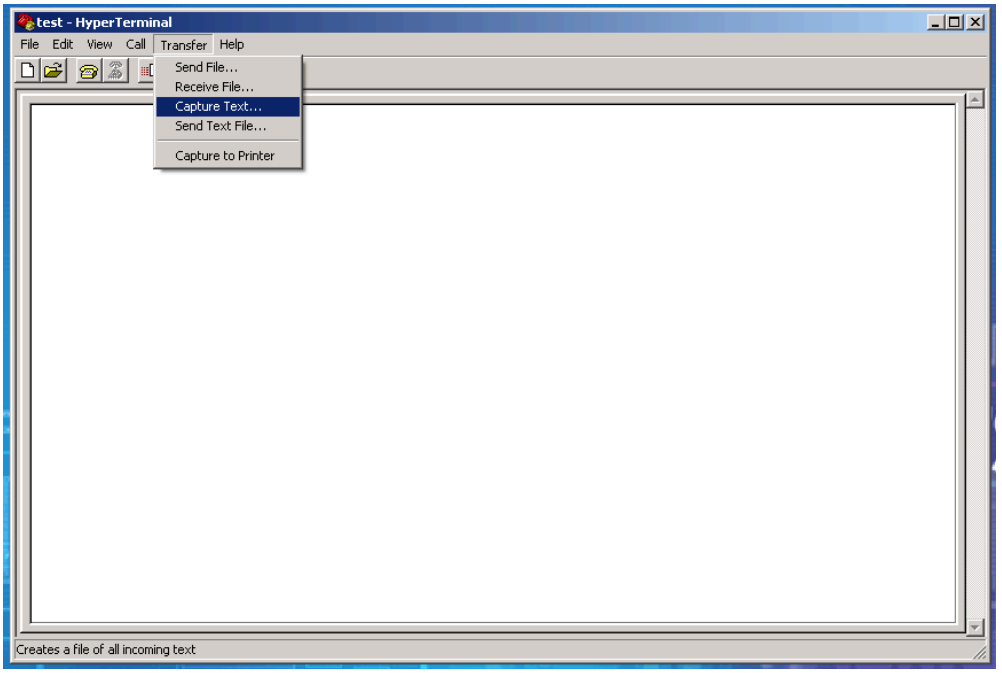




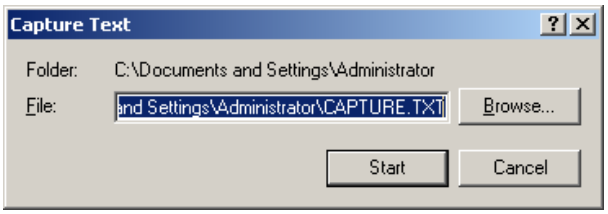
確定**GPS**接收器接收到至少三個衛星訊號再繼續下面動作。

b) 取**data**, 將**data**存成**.txt**

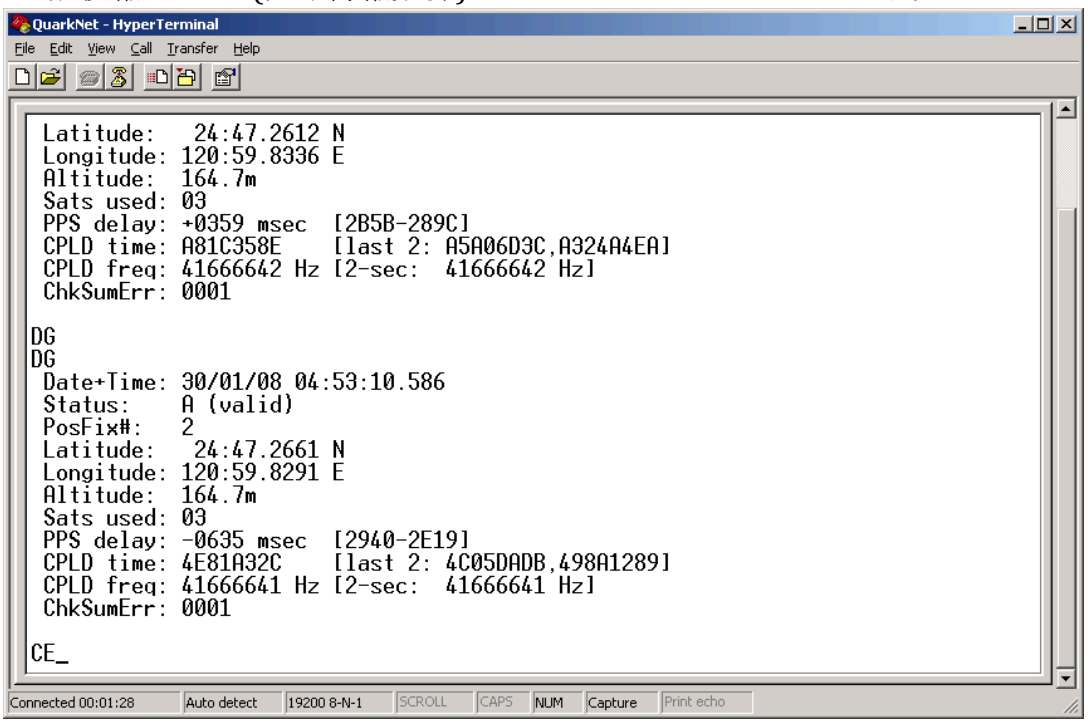
路徑**:** 傳送□擷取文字 **(**Transfer□ Capture Text**)**



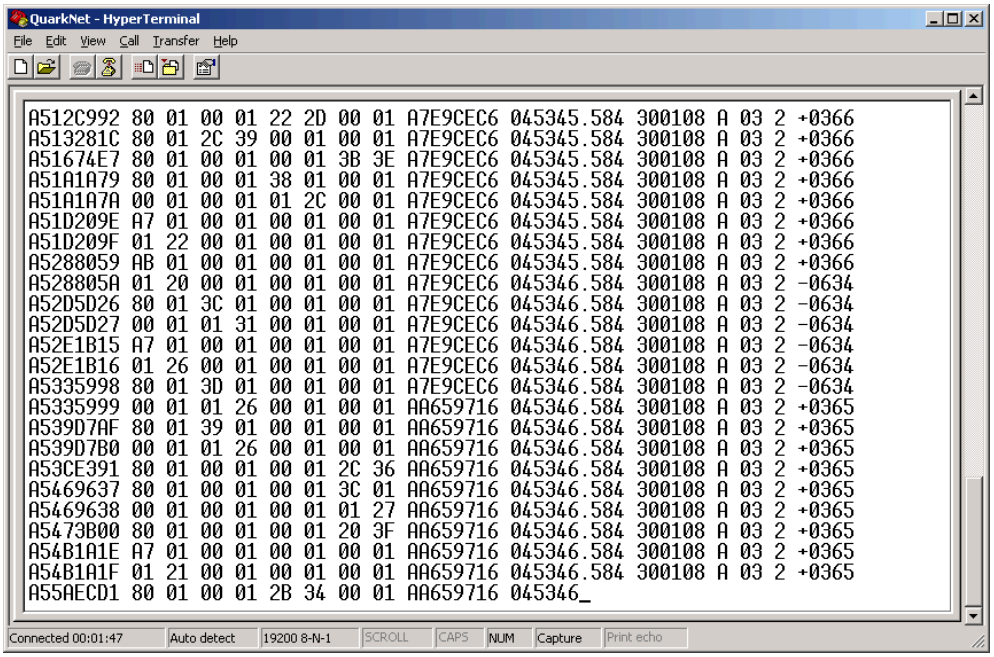
接著就會出現如下的對話視窗, 按**Browse**選擇檔案儲存位置及寫入檔名。



c) 再依序輸入" H1, H2, DG, DC, DS, DT, BA, TH, TI,
V1, V2, ST 2 5, SA 1" (這樣才能把各種設定值寫入檔案內) ~ 要加Enter ~
然後輸入"**CE**" (開始傳輸數據) ~ 要加Enter ~



下圖的情形即開始取數了！



輸入CD停止DAQ card，再到傳送的選項停止攫取數據。

解讀QuarkNet數據卡輸出的數據　07/26/10

QuarkNet數據卡是將數據經過編碼才送到電腦裡儲存起來的，使用者如果知道如何解讀這些碼，就能充分地利用這些數據。以下我們就是要學習如何解這些碼，以獲得有用的資料。

如何讀碼

每一行數據由16個字元(word)組成，舉一例如下：

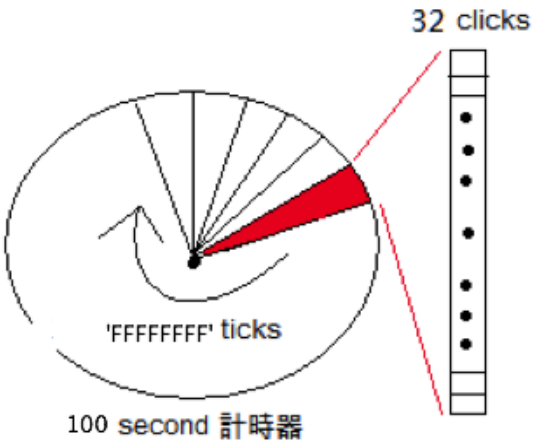
BBA48DB2 80 01 00 01 26 01 00 01 BACCDDDF 054011.721 260105 A 10 2 -0001

其中有幾個字元是以十六進位(HEX)表示(附錄A)，各字元的意義分述如下：

第一個字元

由8個數目字及字母組成，是發生啟動訊號(trigger)時計時器的計數值(clock ticks)，計時值由00000000到FFFFFFFF(十六進位)再重復，經過的時間是1秒, 因此計時值每差1代表40nsec的時間間隔，而每一個計時值的時間間隔稱為一時圈(cycle or clock period)，每一時圈再分成32等份或32個滴答(clicks)，因此每個滴答代表1.25nsec的時間值。我們利用這個字元記錄一個muon到達的時間或一個事件經過的時間長度，另外也可以用在"同時發生"(coincidence)的測量，也就是要記錄同一個muon穿過兩支疊放的探測器的實驗。(在使用delay time 及gate width時，每個count的時間是10nsec，又，在下面的範例裡0.75nsec皆改為1.25nsec，這是因為新的DAQ板的緣故)。

的計時值相差為40ns的整數倍, 因此一個為時120ns的事件, 就需要數行的數據來記錄.

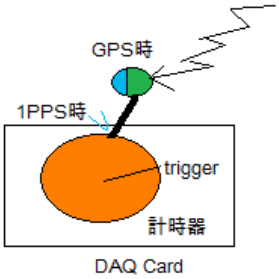


第二個至第九個字元

一共八個字元，分成四對給四支探測器的輸入端(channel)用；每一對的第一個字元記錄輸入訊號上升邊緣(Rising Edge, RE)在當時的時圈裡的第幾個滴答數，第二個字元則記錄輸入訊號下降的邊緣(Falling Edge, FE)，我們給四個輸入端的編號是0,1,2,3。

第十個至第十六個字元

GPS裡有一個線路會固定每秒產生一個100msec的訊號，這就是秒對時訊號(1 Pulse Per Second *1PPS*)，因此第十個字元就是最近產生的一個*1PPS*到達數據卡的計時值，這是一個對時的事件，而當對時事件發生的時候，第十一個字元就是記錄了GPS接收到的UTC時間(以下稱為***GPS時***)。 第十二個字元是日期，再下個是一個字母 V或 A；V代表GPS的訊號是有效的，A則否。再來是兩個數字，代表GPS接收到訊號的衛星的數目，之後的一個數/字(由0至F)，代表數據卡相對於啟動訊號或是*1PPS*有誤差(附錄B) 。GPS每一秒間隔就送出1PPS，有一個對應的***GPS時***，但是到達數據卡的時間(以下稱為***1PPS***時)會不一樣，最後的一個數字(xxxx)就是代表1PPS送到數據卡的時間與 ***GPS***時的差別(0.xxxx Sec)；+表示***1PPS***時相對於***GPS***時有一個延遲 ，–表示***GPS***時相對於***1PPS***時有一個超前(見下圖)。



在進入範例說明之前，需要先提醒幾點值得注意的地方:

1. **一個事件，可能在一個時圈裡結束**，而只有一個記數值；**也可能會連續好幾個時圈**，而涵蓋不同的計數值。通常一個事件會有好幾行的數據。
2. **一般我們要取的只是事件經過的時間長度，但是經過比較複雜的計算(參考以下的範例)，也可以得到實際發生的時間**。
3. **下例裡**，是"**同時發生**"**的測量**，探測器只有編號0及1兩支，**數據卡的指令是** WC 00 13，**訊號輸入端編號0的探測器是放置在編號1的探測器之**

上。

4. 探測器要適當地矯正，設定也要盡量相同，例如閥值(threshold value)的大小就要一樣。
5. 每個數據對，第一個字元記錄訊號RE的時間，第二個字元記錄FE的時間。都需要先換成8個位元的二進位數，使用的編號是 7 6 5 4 3 2 1 0，其中 ：

   編號 7=1代表一事件的開始；=0 代表一事件在延續中。

   編號5=1 代表有效之數據，如果 5=0 則表示無效或無對應之訊號。

   編號43210的五個位元代表由時圈的開始到訊號邊緣的滴答數。由於五個位元的二進位可以表示0至31，所以一個滴答代表 24sec/32=0.75nsce。

範例

只有編號0及1的兩個輸入訊號，儲存到電腦的數據記錄到的某一個事件(含有兩行的記錄)，內容如下:

```
1F7F31B3 BD 01 33 3D 00 01 00 01 1E705220 135455.257 120603 A 04 2
+0609
1F7F31B4 01 2B 00 01 00 01 00 01 1E705220 135455.257 120603 A 04 0
+0609
```

其內涵分述如下：

(1) ***GPS*時 *(Time/Date)=*:** 13:54:55.257 UTC 12Jun03 (**假設當地時間為**9:54 AM)，**可接收到衛星的數目為**4,**有效的**GPS**數據。兩時的差別是** +0.609seconds，所以***1PPS*時 =** 13:54:55.527 + 0.609 = 13:54:56.136 UTC

***(2)*解讀數據的內容**

(a) **第一行的時圈 1F7F31B3:**

數據對(**編號**0**及**1)**的內容**BD 01 33 3D **用二進位表示**：

10111101 00000001 00110011 00111101

第一組位元編號7的值為1，代表這是事件的開始，要先計算此時圈開始的準確時間。

(a-1)啟**動時圈開始的準確時間(Actual trigger time):**

1F7F31B3 = 528429491 (**十進位** decimal)

1E705220 = 510677536 (**十進位** decimal)

528429491 – 510677536 = 17751955 **滴答數**

17751955 * 24 ns = 0.426046920 seconds

答案是 13:54:56.136 + 0.426046920 =13:54:56.562046920 UTC

(a-2)數據對的內容

編號0**的訊號: 有效的**RE，開始處在 11101(BN) =1D(HEX)=29(DEC)

沒有 FE

編號1**的訊號: 有效的**RE，開始處在10011(BN) =13(HEX)=19(DEC)

有效的FE，結束在11101(BN) =1D(HEX) =29 (DEC)

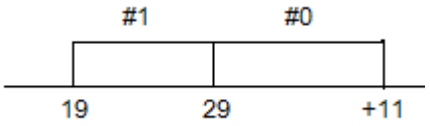
(b) **第二行的時圈1F7F31B4:**
數據對(編號0及1)的內容01 2B 00 01用二進位表示：
00000001 00101011 00000000 00000001
第一組位元編號7的值為0，代表這是事件的延續。

(b-1) 數據對的內容

編號0的訊號: 沒有 RE
有效的FE，結束在 01011(BN) =0B(HEX)=11(DEC)
編號1的訊號: 沒有 RE
沒有 FE

(3) 總結

我們可以看到在這個例子裡，事件發生在時圈1F7F31B3**裡，編號**0**有一個在**(29*0.75) = 21.75 nsec**的地方開始的**RE**，編號**1**有一個在** (19*0.75) = 14.25 nsec**的地方開始的**RE**，還有一個在**(29*0.75) = 21.75 nsec**的地方結束的**FE。此一事件在另一個時圈1F7F31B4**裡結束，其中編號**0**有一個在**(11*0.75) = 8.25 nsec**的地方結束的**FE**，編號**0**則無訊號。**
因此，編號0的訊號長度為 10.50 nsec (=(24 - 21.75) + 8.25)nsec)，
編號1的訊號長度為7.50 nsec (=(21.75 - 14.25)nsec)，各自開始與結束的地方如**下圖所示**。每一個邊緣(edge)的準確時間可以加上(a-1)的計算結果得知。



附錄 A 十六進位(HEX)與二進位(BN)的轉換

0	000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

附錄 B 數據卡狀態指標

先換成二進位 可得四個位元 位元的編號為 0 1 2 3 對應的意義如下
編號 0 = 0 OK
=1 PPS暫停， 警告： 數據卡忙碌中，1PPS的計數受到影響。
編號 1 = 0 OK
=1 啟動訊號受到影響，可能是啟動訊號太頻繁；如果持續的話，數據會錯誤太多。

編號 2 = 0 OK
= 1 GPS訊號受到影響，可能是數據卡當時正在忙碌中。
編號 3 = 0 OK
= 1 計數值發生錯誤，計時器或有GPS問題，也可能是數據卡太忙碌。

如何管理**QuarkNet e-Lab** 帳號

介紹

在2009年10月，Bob Peterson(Education Program Leader of Quarknet)建好幾個台灣(TWN)Lead Teacher的帳號，可以讓Lead Teacher進入QuarkNet的e-Lab管理自己帳號下的group。Lead Teacher在 http://www18.i2u2.org/elab/cosmic/home/project.jsp (或 http://quarknet.fnal.gov)的帳號及密碼如下：

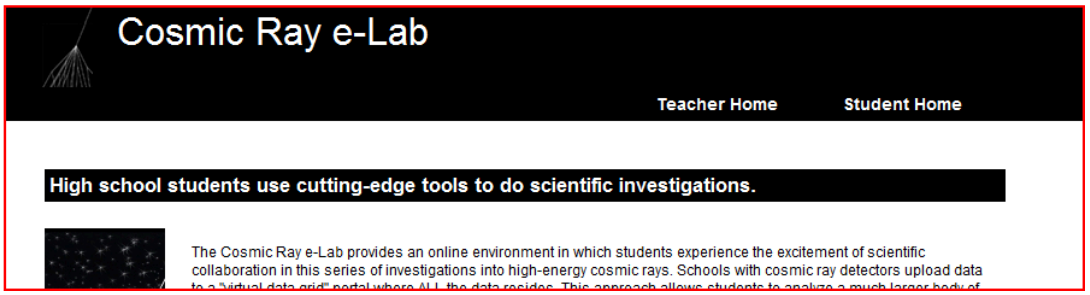
(1) **台北市**(Taipei)**東吳大學**(Soochow University)**的**Lead Teacher
帳號TWQuarkNet
密碼xxxxxx
(2) **台北市**(Taipei)**中央研究院物理所**
(Institute_of_Physics_at_Academia_Sinica_IPAS)**的**Lead Teacher
帳號CRipas
密碼xxxxxx
(3) **台北縣**(Taipei_County)**輔仁大學**(Fu_Jen_Catholic_University_FJU)**的**Lead Teacher
帳號CRfju
密碼xxxxx
(4) **台南市**(Tainan x)**成功大學**(National_Cheng_Kung_University_NCKU)**的**Lead Teacher
帳號CRncku
密碼xxxxxx
(5) **高雄市**(Kao-Hsiung)**高雄師範大學**
(National_Kaohsiung_Normal_University_NKNU)**的**Lead Teacher
帳號CRnknu
密碼xxxxxx

e-Lab使用者(簡稱group)的帳號，除了guest之外，分成三類(role)：teacher，user 及 upload，其中user類的帳號不能上傳data，不能建其他帳號，upload類的帳號可以上傳data，不能建其他帳號，teacher(包括 lead teacher)類的帳號可以上傳data，也可以建立以上三種自己帳號下的group。台灣group上傳的資料，目前會標記到以下其中的一個地方:

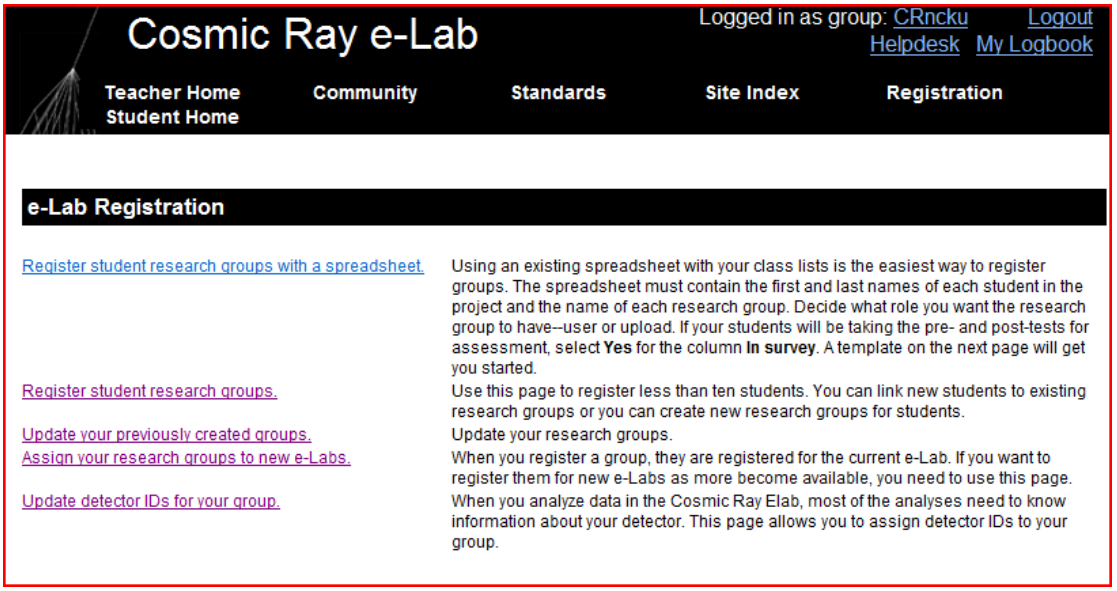
SoochowU---Taipei---TWN， ipas---Taipei---TWN，
FJU---Taipei_County---TWN，NCKU—Tainan---TWN，
NKNU---Kao-Hsiung---TWN。

Group的管理

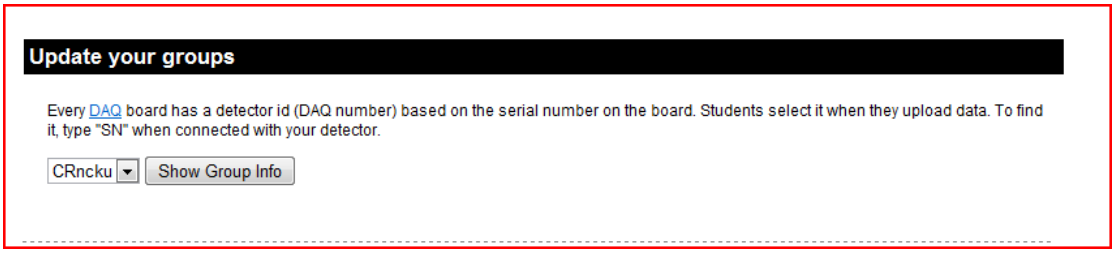
具teacher身分的使用者(例如CRncku)進入e-Lab的頁面如下，選擇 Teacher Home 進入，然後選擇下一頁面的Registration， 幫你的group建立帳號及做設定。



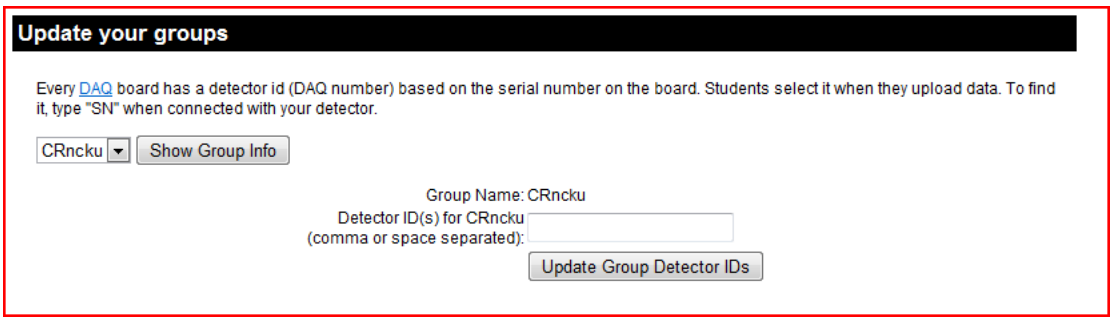
以下有5個選項 ，一般只要用到2,3及5項。



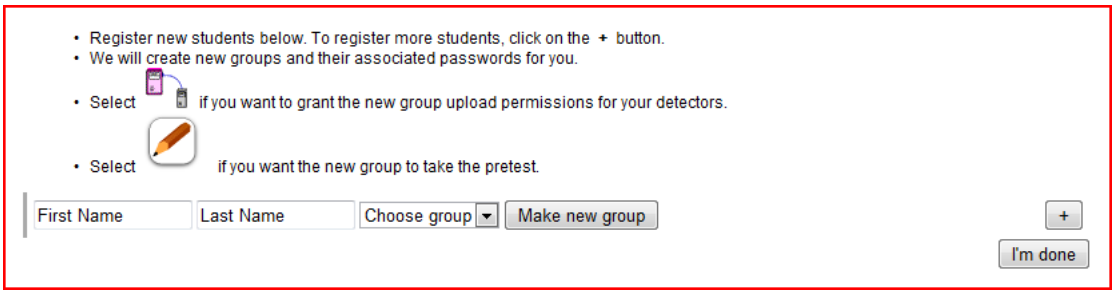
(1)先進入第5項，目前只有CRncku一個帳號 按Show Group Info。



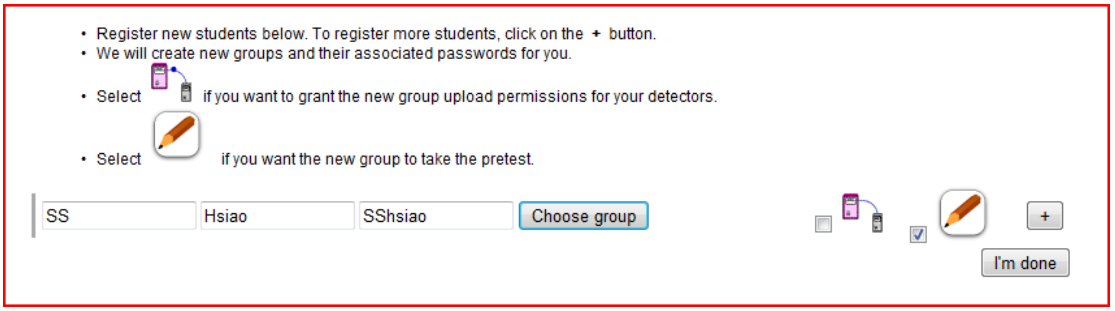
在輸入DAQ卡編號的地方輸入DAQ卡背面的4位數號碼，如果還有其他的DAQ卡，就接著輸入其他號碼，以逗號分開，結束就按 Update Group Detector IDs，以後你的group就可以上傳這幾個DAQ取得的 data。



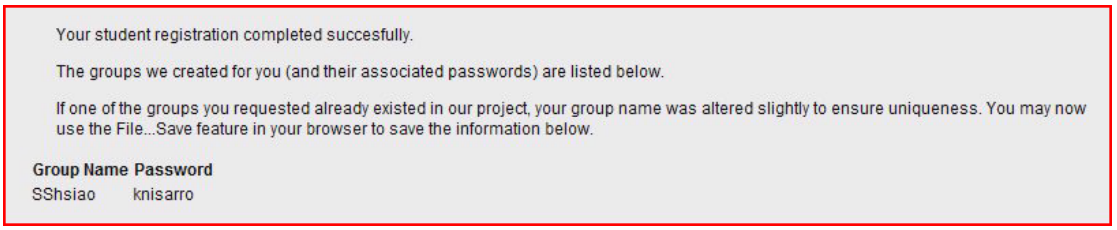
(2) 進入regristration的第2項 Register student research groups， 跳出以下的頁面，



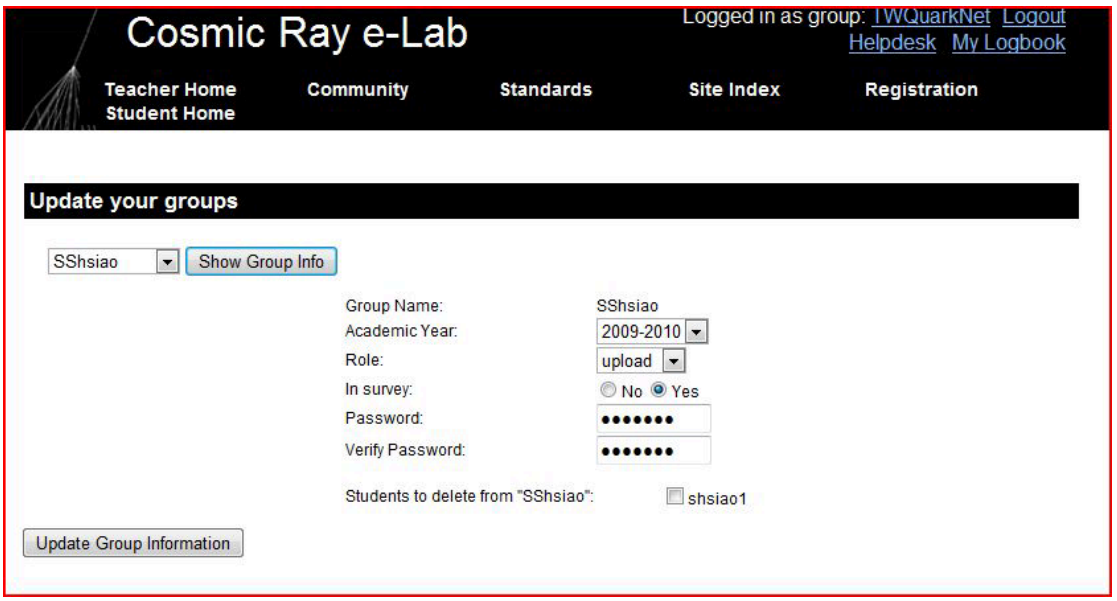
輸入 First Name Last Name， 然後選 Make new group，即換成以下的畫面，在Group Name的地方給一個group name(就是使用者的帳號)，在右邊選要不要讓這帳號上傳data及要不要做一個測驗，如果還要加其他的group name 就按 + 號，依此類推，告一段落就按I'm done。



接著就出現剛剛建好的帳號及以下的訊息，告訴你新的group name 及password，但是系統給的password都不好記，請馬上寫下來。如果密碼忘記了，可以由建帳號的teacher來重設，只要回到Registration的頁面，可以進行密碼重設及其他的設定。

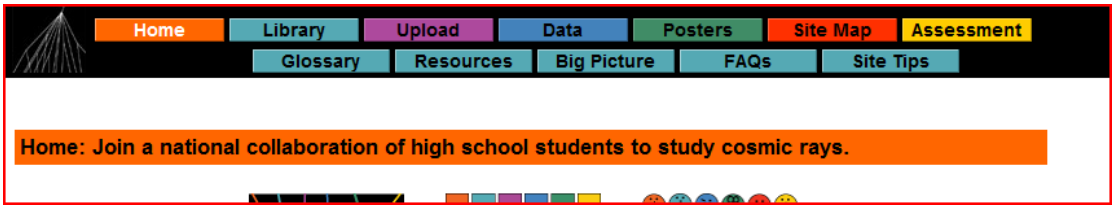


(3) 進入Regristration的第三個選項 update your previous created groups，在左邊選要處理的帳號， 再按Show Group Info就可以開始更改帳號的類型(role)為user, upload或teacher，及是否要學生做測驗 以調查學生是否有進步(In survey= No or Yes)，還有更改密碼(Password)，而且自己的密碼也可以在此更改，最後再按 Update Group Information就存檔了。

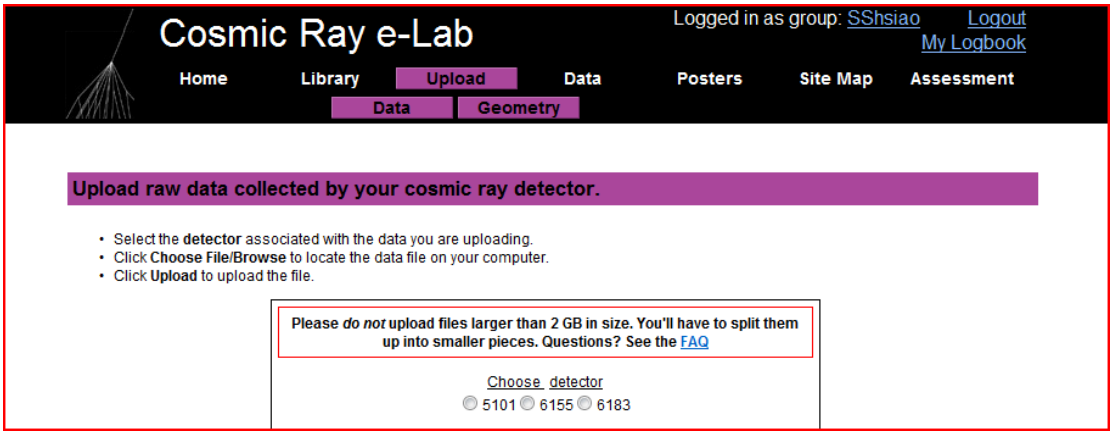


探測器及**DAQ**卡必須在**e-Lab**裡先作設定

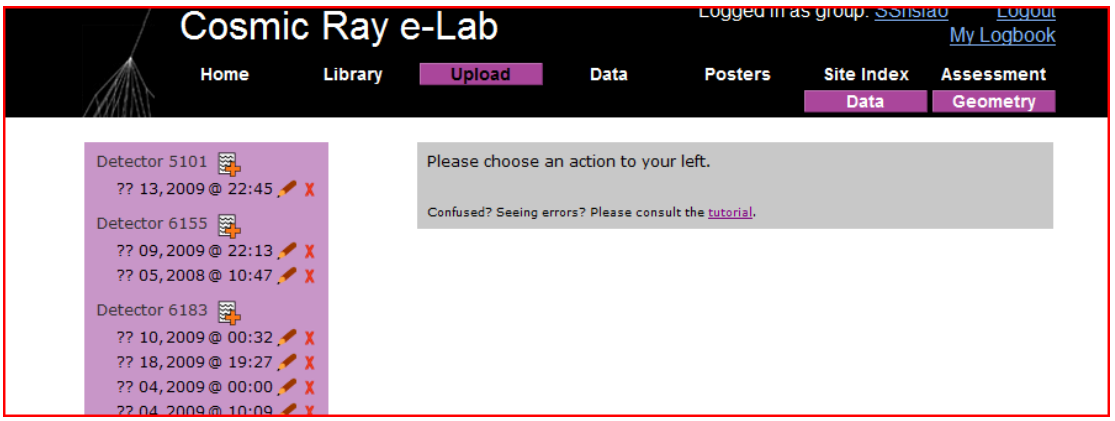
以下還要做一些設定才能上傳data。不需要重新登記進入，可以由上面的工具列到Student Home。



選擇Upload，出現Data及 Geometry兩個選項，選擇Geometry。



在上傳資料前，一定要在這裡將 探測器及DAQ卡的位置設好，左邊有之前設定帳號時輸入的DAQ號碼，在號碼旁有+的地方按下做設定(enter a new entry)，設定完就會多出一行記錄來。要修改之前的設定，就在有一個小鉛筆的地方按下，如果在有x的地方按下，設定就會消除。



TW-Quarknet-data-preparation

名詞對照

Benchmark File 腳本檔

Blessed file 角色檔

Blessing 對角色

Singles 單時率

Single coincidence 單符合

Double coincidence 雙符合

Triple coincidence 三符合

Geometry 擺置幾何

Trigger rate 觸發率

Configuration 配置

前言

2013年起, 夸克網的e-lab 陸續做了三個改變;

1, DAQ卡的參數都設定好之後, 在取數據前要先下指令:

ST 2 5 或 ST 3 5

TL

SA 1

這是為了能夠將狀態參數寫到數據檔,在benchmark (腳本)與blessing(對角色) 時要用到.

2. 檔案完成上傳的時候, 系統會馬上檢查有沒有照上面的步驟做, 同時上傳檔會自動依照UTC的時間分成不同日期的幾個檔. 沒有通過檢查的數據不會被存入系統.

3, 上傳到e-lab的數據檔, 要做benchmark與blessing, 才能夠與別人分享數據, 否則該數據就只能自己使用.

A **Benchmark File** represents a high quality one day data file for a single geometry and coincidence.

每一個擺置每一種同時性, 都需要有一個"一天"數據量的腳本檔.

Other data files will be compared to this file to **bless** the new upload.

其他的上傳數據與腳本檔比對, 通過對角色的步驟, 才能成為角色檔.

When the detector is reconfigured with a new coincidence or geometry you have to designate a new benchmark (See "Add Benchmark").

如果探測器的擺置與同時性有做改變, 你就要重新做一個腳本檔.

How do you judge a "Benchmark File"?

Steady singles & trigger rates.

Sufficient rates to assure adequate S/N.

如何可以成為腳本檔?
. 穩定的單時率及觸發率
. 秒率有夠好的訊號雜訊比

Note: for the "6000" series QuarkNet counters, the singles rate is usually between 12 to 35 Hz.
Other style counters will be different and is best determined by plateauing counters (see CR e-Lab→LIBRARY→Resources→Tutorials for plateauing instructions).
6000序列的quarknet計數器, 單時率大約在12到35Hz. 其他類型的計數器都要先做過平坦區再做決定.

使用輔仁(CRfju/cosmic)的儀器, 數據卡是# 6226, 配3 支計數器(700,760,820/TL600/~20Hz), 上傳數據用的帳號是: fjupha/cosmic. 使用軟體muonic, 獲得工作電壓, 然後用 minicom 將數據卡取得的數據存檔; 先得3隻計數器一天的單符合訊號, 再取得一天的雙符合訊號, 之後可以作為腳本檔 (benchmark file).

第一套腳本:8-26-26single

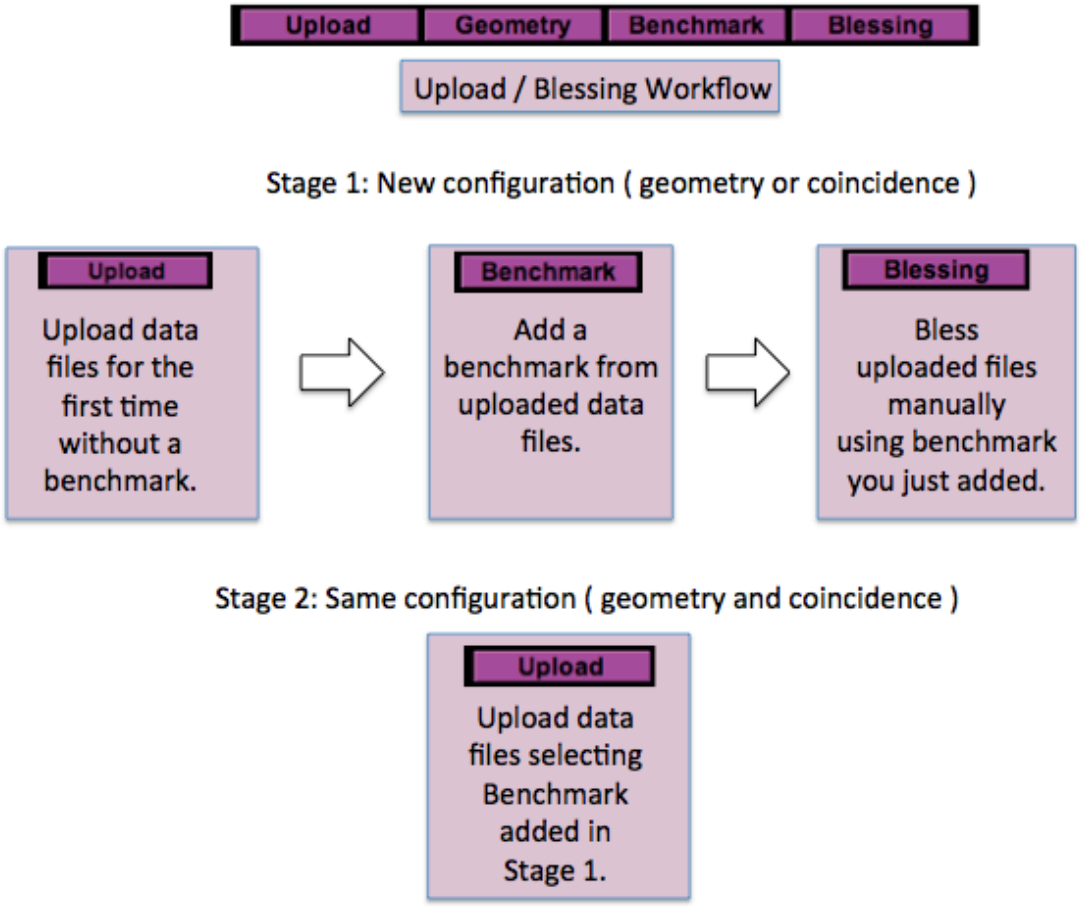
第二套腳本: 8-27-26double

(成大CRncku/cosmic, CRncku251)

Overview of Data Blessing簡介對角色的程序

Data Blessing checks the rates in each channel and the trigger rate for consistency. Blessing involves two stages in your workflow depending on whether you are uploading data with a new configuration or with the same configuration.
對角色是檢查對應到每個通道的計數器都是維持在正確工作的狀態, 同時通道的計數率與觸發率是否有一致性. 對角色有兩套做法, 一個是手動, 一個是自動, 分別如下圖所示. 這兩

種作法都要求要先有腳本檔.

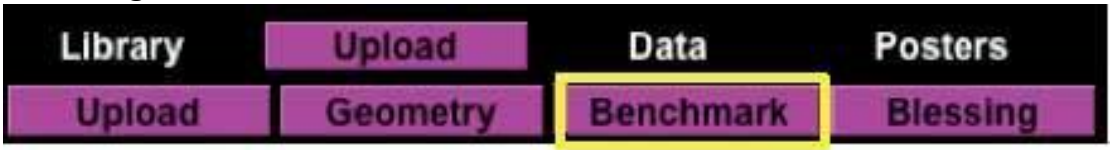


These two stages use two methods for you to "bless" your data and make it available to e-Lab users. Each requires a benchmark file.

- **Manual blessing:** In manual blessing, you have an opportunity to bless files that did not have a benchmark when you uploaded them (Used in stage 1 as shown in the workflow above).

- **Automatic blessing:** In automatic blessing, we compare your upload to a benchmark file you identify (Used in stage 2 as shown in the workflow above).

Choosing a Benchmark



To select a benchmark file, click on the Benchmark menu item and inspect the blessing plots for several existing uploads. After you inspect the plots, choose an appropriate file to label as benchmark. Read this tutorial for more information.
You MUST change the benchmark file when you change the configuration (coincidence or geometry) of your detector. You can use the same benchmark

over and over if you never change the coincidence or geometry of your detector.

Manual Blessing

You can use this method to bless files that you uploaded without automatic blessing. This method still requires you to select a benchmark file or gives you an opportunity to bless data that had been compared to the wrong benchmark at upload time.
Read the tutorial on manual blessing

Automatic Blessing

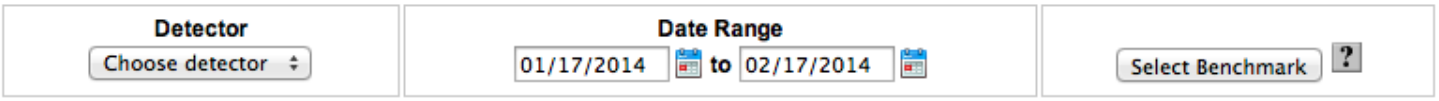
On each data upload you will select a "benchmark" file to compare your data with. You define the benchmark files; they have consistent data rates and similar geometry and triggers to the subsequent upload files that you wish to compare them to.

Benchmark Tutorial 如何設定腳本

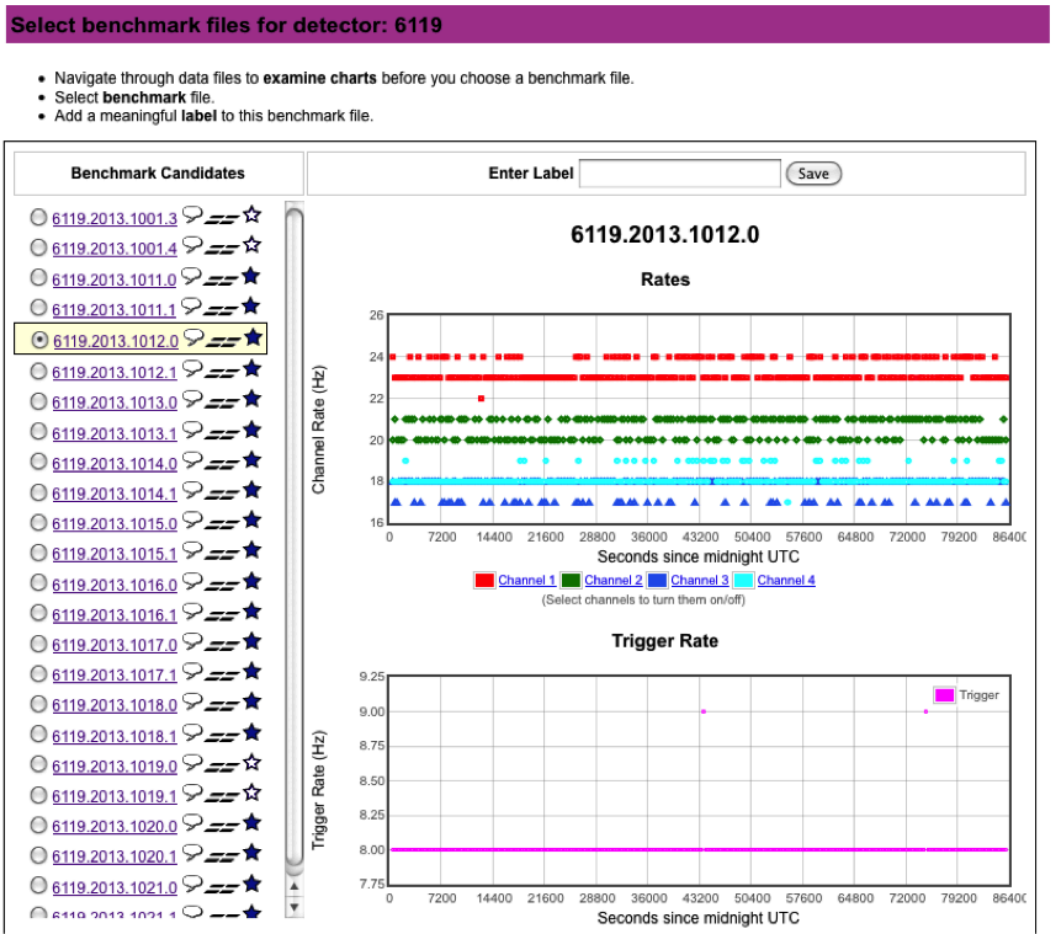
Go to: "Cosmic Ray e-Lab→Upload →Benchmark"

This is where a detector owner will select a **best standard** data file for comparison to other files in order to bless the other.
The file owner needs to review the blessing plots and use judgement to select one standard file.

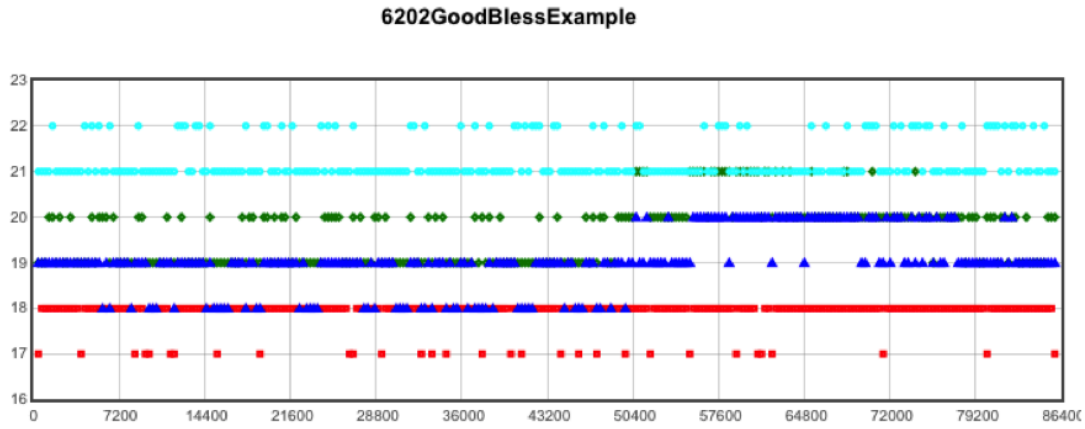
So, first pull down the DAQ list to see your DAQ#s,



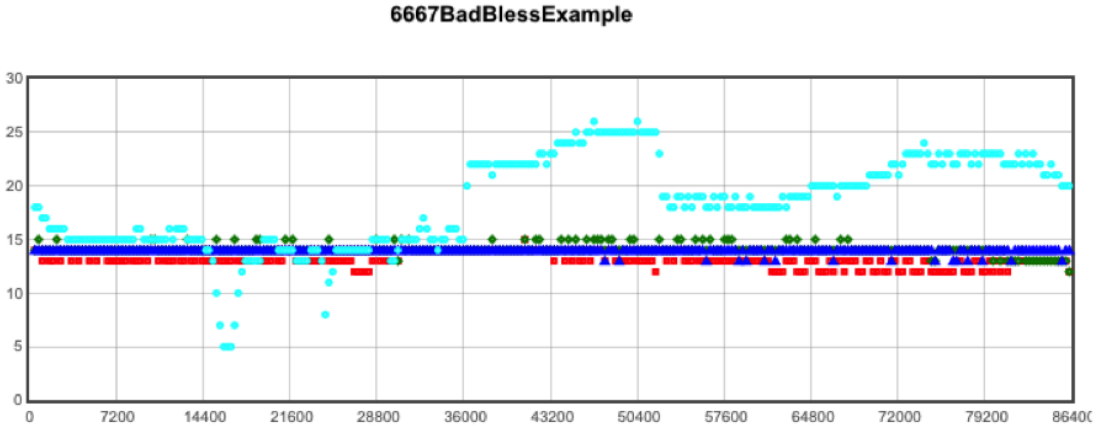
and select your appropriate DAQ#.
Now, set the date range to include one CRMD configuration (coincidence and geometry).
Click "Select Benchmark". This will present a list of candidates for selection.



Review the list of candidates by clicking on the file number (such as "6119.2013.1012.0") and looking at the Singles Rates and Trigger Rate.
Use your judgment to decide if this file is representative of all the data for your particular CRMD configuration. A high quality file will have steady Singles & Trigger Rates, plus Singles will fall between 12 to 35 Hz for the standard QuarkNet counters. Here is an example of good blessing plot:



and this shows a bad blessing plot:



Notice how one of the channels wanders around indicating irregular rates.

After you have selected a representative file, type a helpful name that you will recognize later and click "Save".
Now, when you go back to "Upload" → "Upload" the chosen benchmark will appear on the pull-down list next to the DAQ#.

For the DAQ# for which you are uploading data, select a benchmark file from the pull-down list. This benchmark file will now be compared to the uploaded data to measure its quality.

Later, if the detector configuration changes (coincidence, geometry) then a new benchmark file needs to be selected.

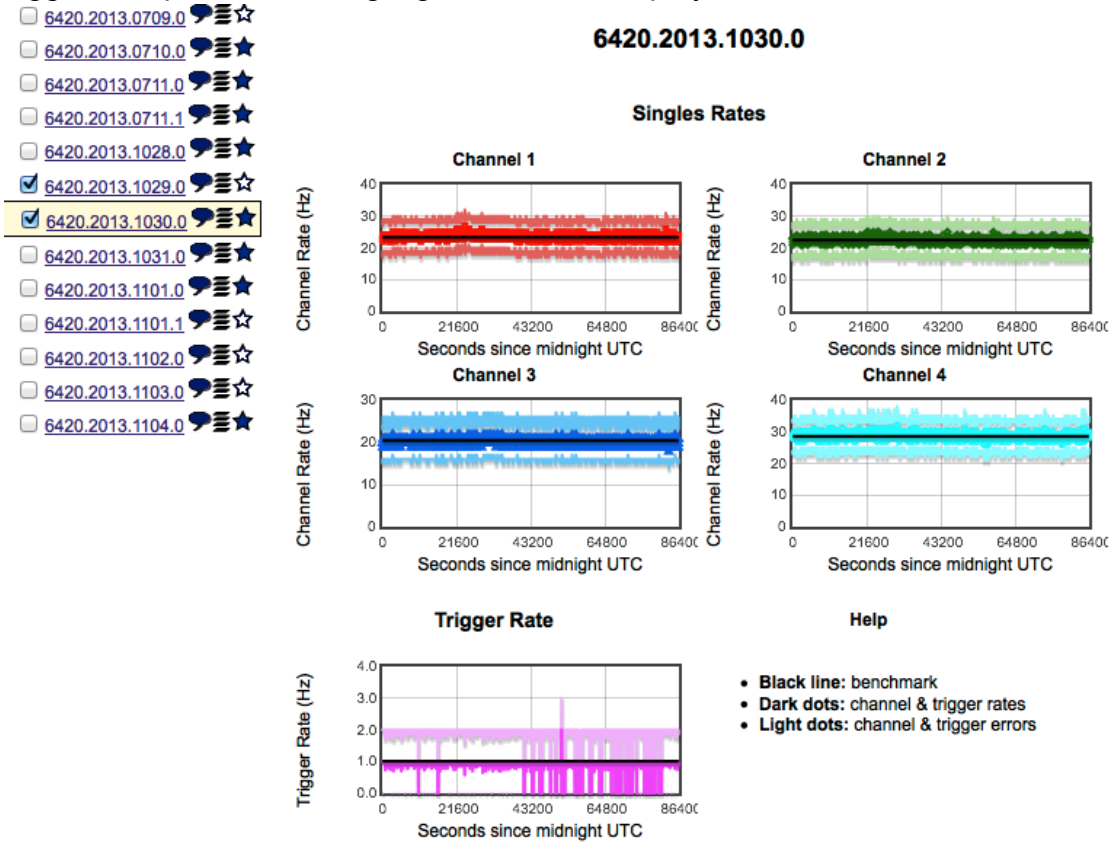
When selecting files for analysis, review the blessing plots by clicking on the "star". See if you agree with the owner assessment for quality data.

Tutorial for Manual Blessing of Data Files如何將數據檔以手動方式來對角色

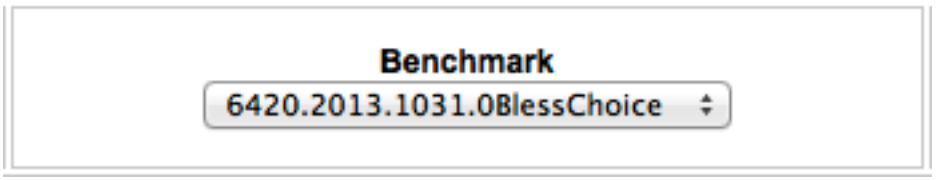
To manually "bless" a data file or a set of files, first select the DAQ# from the pull-down list:

To display the list of potential files to be "blessed", click the Retrieve all box. The list will then appear though it may take a moment to generate the list; please be patient.

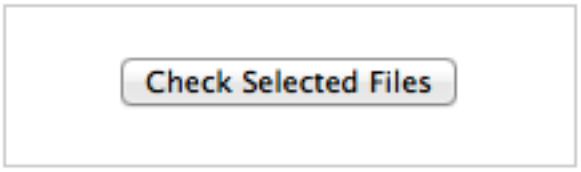
From the candidates list, you can select individual files or "Bless all files" will attempt to verify the entire list. Shown are two files selected. The singles and trigger rate plots for the highlighted file are displayed.



From the Benchmark pull-down list, select the standard file you wish to compare to the candidate file:



and now click "Check Selected Files" and the CR e-Lab will compare the rates and errors to see if passes.



In this case, one file was "blessed" and the one file failed.

6420.2013.1029.0 has NOT been blessed. Fail reason: This file failed at: 50550(14:2:30) because the benchmark channel 4 rate: 28.569793282251904 (metadata value) was not between the ranges of comparison set by 14 and 4(10.0 and 18.0) - for these last values, look at the .bless file of the just split file.

6420.2013.1030.0 has been blessed.

Hardware notes: 24MHz clock on board, 1.25ns/tick , on board 10nsDT, 50nsDT-GPS
Software notes: muonic (Linux and python based), EQUIP(windows,Mac and Java based)

So, for each of these 2 character hexadecimal words, you essentially have 8 binary bits of information.
The way that this information is encoded is as follows: Bits 0, 1, 2, 3, and 4 denote an even more preciseclock count on the board of 1.25 nanoseconds. Bit 5 should be 1 for every valid event, while bit 7 should ONLY be 1 for word 2 - it denotes the start event of a cluster of events. Bits 6 and 7 are always 0. Therefore,the binary word 1011 0010 would mean that a new event in a new cluster has started at 22.5 nanoseconds after the initial time measurement.

Looking at one channel may require you to look between several lines in a cluster. The first line of the
19 cluster is always indicated by the 1 in Word 2, and typically it is easy to follow along the cluster by looking at Word 1 and seeing when the hexadecimal stamps stop being consecutive.

GPS firmware update procedure

這一塊是下載更新軟體的步驟:
Download Firmware Update

1.On a PC, down load the update files at http://www.conwin.com/support.html

2. The filename is “wi125_update_v1.79.00.29.14.rar”.

3. Unpack all files to a local working directory. (password: navsync). WinZip may work to unpack, but Dave and Mark downloaded PeaZip.

這一塊是處理GPS盒子裡面GPS卡的初始步驟:
Setup for GPS Card

1. Access GPS card by removing plastic case cover.
   打開GPS外盒

2. Note the location of JP5 (Figure1). Older versions have R9 surface mount pads instead. 參考圖一, 找到GPS卡上JP5的所在. 舊版的GPS卡只有R9(表面鍍層)

3. Done with the GPS card for now.
   GPS的初始步驟完成.

這一塊是設定DAQ卡: **Setup for DAQ Card**

4. Connect DAQ to PC usb port and run your Terminal App (i.e.TeraTerm). Download Tera Term because EQUIP can’t configure the required baud rate easily.
   將GPS接到DAQ卡, PC上準備好 TeraTerm的終端機程式, 其他的終端機程式也可以, 但是必須要能做以下的設定.
   再將DAQ卡插入PC的USB槽, 用終端機程式連接上DAQ卡.

5. Set DAQ baud rate to 38.4K (command ‘SB 123 2’).
在終端機連接**DAQ**卡的畫面上設**DAQ**卡的**baud rate** 為**38400**，**DAQ**卡的指令是 **SB 123 2**

6. Now Configure your Terminal App to communicate at 38.4K baud – select Setup; serial port; set speed.
這時**DAQ**卡的**baud rate(38400)**跟終端機程式的**baud rate (115200)**不一樣, 因此終端機程式與**DAQ**卡失去連線, 無法控制**DAQ**卡. 為了重新獲得連線與再控制**DAQ**卡, 必須修改終端機程式的**baud rate**為**38400**, 程式的選項依序為: 設定，序列埠，設速率(即**baud rate**). 這樣可以控制**DAQ**卡的畫面就會重新出現.

7. Enter DAQ command ‘UL 123’. DAQ will report “This is a USB to GPS link” 在新的**baud rate** 連線的情況下，輸入**DAQ**卡的控制指令: **UL 123** 這時候 **DAQ**卡會回應: **This is a USB to GPS link**

8. If at any point the DAQ card is reset, repeat steps4-8.
如果發生**DAQ**卡重設的情況，就要重覆步驟**4-8.**

9. DAQ is now in download mode, ready to send in coming data to the GPS board.
現在**DAQ**卡處在下載的模式裡，預備將更新的資料送往**GPS**卡.

10. Now **disconnect t**he RJ45 cable from the GPS to DAQ at the DAQ card. 將**GPS**連接到**DAQ**卡的**RJ45**線線頭拔出, 這樣**DAQ**卡與**GPS**失去聯絡, 但是也將**GPS**處在無電源供應的情形, 可以安全地做下一個步驟.

11. Disconnect Terminal App from the com port. If Tera Term – select File and disconnect. 操作終端機程式, 選擇離線, 釋出對應到**DAQ**卡的序列埠

12. Do not reset the DAQ card. 不要讓**DAQ**卡重設**!**

這一塊是要用**Win32 GUI** 應用程式來更新 **GPS**卡**:**

Using Win32 GUI Application to update GPS

13. Open application upd_wi125.exe in your local working directory. 在之前下載更新軟體的目錄區裡, 選擇執行 **upd_wi125.exe**

14. In the NavSync Firmware Update Window, select DAQ COM Port (same as above). 這時會出現 **NavSync** 更新軟體的視窗, 選擇對應到 **DAQ**卡的序列埠

15. On the GPS board short out JP5. (tweezers, paperclip) (If you have an older GPS card and there is no JP5, short out R9)
取一小塊導體(例如將平的螺絲頭立起來), 穩穩地放在**GPS**卡上, 將**JP5**的兩個小點短路, 在之後的動作裡, 要確保這個導體不會移動. 舊版的**GPS**卡, 就要將**R9**短路.

16. **Connect** the GPS to DAQ using a RJ45 cable. A download timer has now started. Wait 5 seconds. 把**GPS**的**RJ45**線連接到**DAQ**卡, **GPS**獲得了電源, 再等**5**秒,~~**DAQ**卡就可以開始跟**GPS**卡送出更新的資料子~~.

17. Now on UPD_WI125 app click 'Update' within 15 seconds else download fails.
現在, 要在**15**秒內, 必須在**upd_wi125** 的應用程式上按下**Update**的選項, 否則更新就失敗了.

18. Programming will take about 90 seconds. upd_wi125 will show status information and report "Writing flash memory", "Verifying flash", and "Firmware update complete".

    You can remove the short out jumper on the GPS card after "Writing flash memory" appears.

    接下來, 會需要約**90**秒的時間來完成更新. 在進行中, **Upd_wi125**顯現的狀態有: **Writing flash memory, Verifying flash** 及 **Firmware update complete.** 當狀態顯示的是 **Writing flash memory** 的時候, 你就可以移開

GPS卡上短路JP5的小塊導體了. 當然, 顯示狀態為Firmware update complete的時候, GPS卡的更新就完成了!

19. If programming fails disconnect the GPS RJ45 cable and repeat step 13-18. 如果更新失敗了, 就要重覆步驟13-18.

20. Reset DAQ board.
關掉upd_wi125, 重新開DAQ卡的電源, 這樣DAQ卡就會重設, 再用原本quarknet的baud rate將終端機程式連接上DAQ卡.

21. Use command 'DG' to check GPS time and date (only valid after satellite lock). 用指令 DG檢查GPS的時間對不對(要先接到數目足夠的衛星).

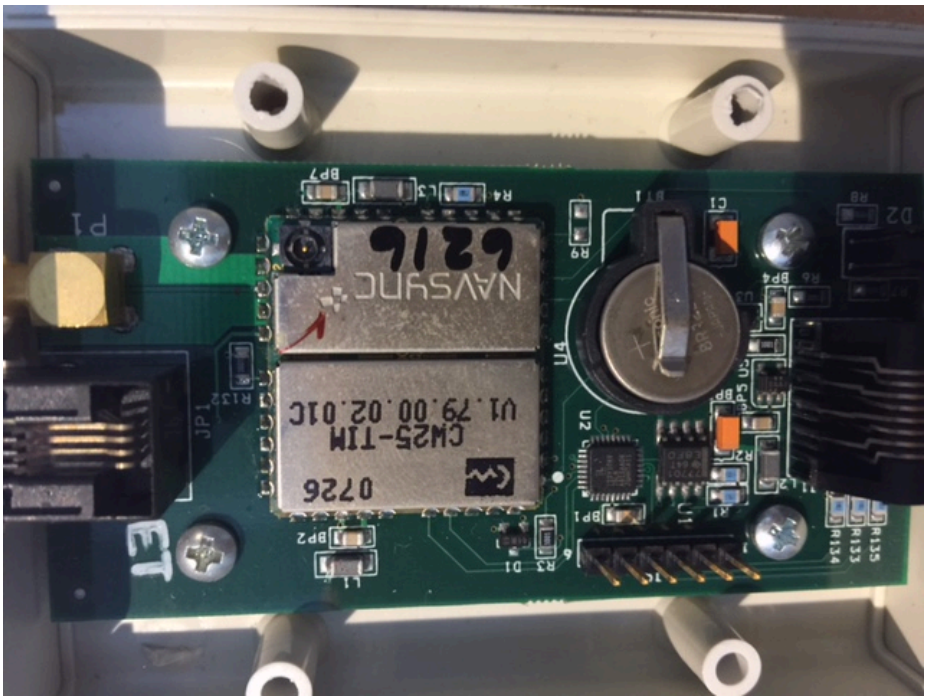


Figure 1, Qnet GPS CARD

圖一, GPS卡, 我的GPS是舊版, 所以在電池旁邊有兩顆小點 的R9, 是以上步驟裡需要短路的地方.

2016 QuarkNet-TW維修工作坊

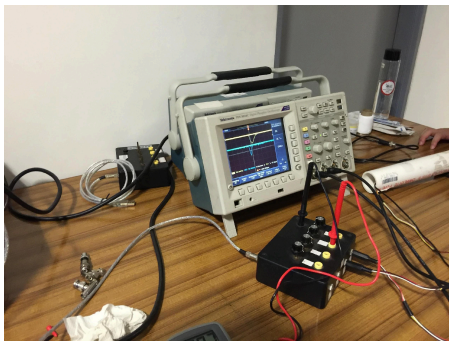
主辦單位: 中央研究院物理所, 輔仁大學物理系, 成功大學物理系, 東吳大學物理系

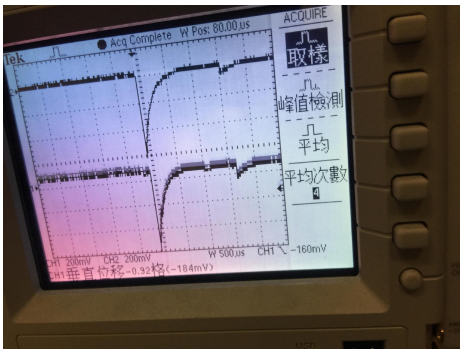
7/15	星期五	7/16	星期六
8:30 | 9:00	報到	8:30 | 9:00	報到
9:00 | 9:30	簡介QuarkNet(夸克網) (主講:蕭先雄)	9:00 | 9:50	1.分組組裝計數器 2.測試計數器 3.架設quarknet儀器: 一套包含4支計數器 (主講: 康慧玲)
9:30 | 10:50	1. 介紹Raspberry pi的使用: 開機, 用筆電連線, Linux, GPIO, Python程式 2. 製作檢測PMT的LED線路板, 上機練習 (主講:蕭先雄)	10:00 | 10:50	如何使用取數據的軟體(主講:蕭先雄): 1. minicom 2. muonic 3. EQUIP 4. python程式 上機練習
11:00 | 11:50	拆解quarknet計數器, 清潔零件. (主講:蕭先雄)	11:00- 11:50	量測計數器單符合與雙符合的平坦區(plateau):工作電壓與閥值 (主講: 康慧玲,蕭先雄)
12:00 | 1:00	1. 休息 2. 自由討論與示範	12:00 | 1:00	1. 休息 2. 自由討論與示範
1:00 | 2:00	1. **製作光罩: 將PMT與LED線路板放進光罩.** 2. **示波器與樹梅派檢測拆下來的PMT** (主講:蕭先雄)	1:00 | 2:00	量測計數器單符合與雙符合的平坦區(plateau):工作電壓與閥值 (主講: 康慧玲,蕭先雄)

2:00 | 5:00	1.分組組裝計數器(主講: 康慧玲) 2.以示波器測試計數器(主講: 鄧炳坤) 3.架設quarknet探測器: 一套包含4支計數器(主講: 康慧玲, 蕭先雄)	2:10 | 5:00	上傳數據到e-Lab (主講:蕭先雄)

地點**:** 中央研究院物理所**4**樓**P4H**會議室 **.**

自帶物品**:** 筆電**.**



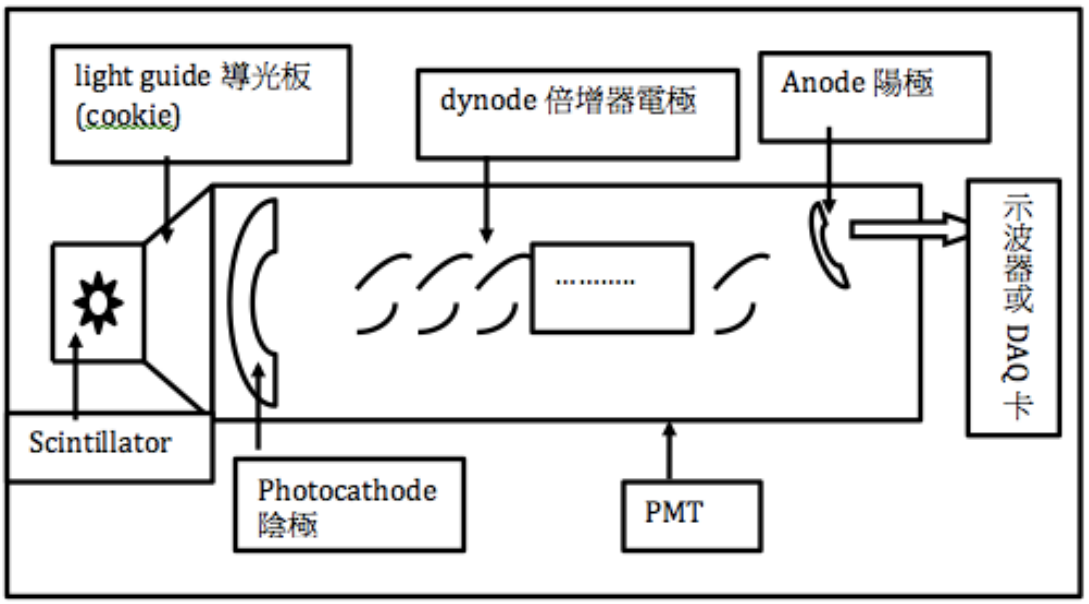


維修Quarknet閃爍計數器(Scintillator Counter)

拆解, 檢測, 組裝與量測

閃爍計數器工作原理

閃爍計數器包含光電倍增管(PMT)閃爍板(scintillator)兩部分.
當宇宙線撞擊到閃爍板, 使閃爍板發光, 這個光會在閃爍板中散射,
最後經由導光板進入光電倍增管(PMT)中, 將光轉變成電流, 由於PMT可
將訊號放大百萬倍以上, 就可輕易利用示波器或DAQ卡讀取訊號。



Quarknet用的PMT是Sens-Tech的設計(型號P30CW5), tube base裡加上小型HV power supply , http://www.sens-tech.com/products/modules/

(1) 拆解quarknet閃爍計數器

零件與材料：

quarknet計數器 +剪刀+刀片 + 手套 + 酒精 + 拭鏡紙 .

1. 將Quarknet 計數器(圖一)的白色塑膠管拆開移除, 儘量不要拉扯到接線, 白膠管放在一旁. 如果白膠管上面殘留的的黏膠很多, 嘗試以大量酒精塗抹清理.
2. 露出來的部分是光電倍增管(Photomultiplier Tube , PMT), 圖二顯示這隻偵測器的PMT有鬆脫
   的現象.
3. 拆開接合PMT的部分, 移除纏繞的膠帶, 取出PMT, 把PMT的玻璃面先用酒精擦淨, 再用鋁箔紙蓋住, 先放在一旁安全的地方.
4. 接PMT的另一邊是包好的閃爍板(如圖三), 閃爍板一角有一個餅乾狀突出(cookie), 將白膠布拿掉, 用拭鏡紙擦拭cookie, 用拭鏡紙蓋住cookie, 整片放在一旁.

5. 揭開包裝閃爍板的膠帶, 打開黑紙, 拆下的黑紙如果是有亮面的樣式, 可以放在一旁備後用.
6. 稍微移除些鋁箔紙上的膠帶, 儘量保留鋁箔紙的完整, 不要打開包覆閃爍板的鋁箔紙, , 以避免手污染到閃爍板的表面, 閃爍板連鋁箔紙一起放在安全乾淨的地方.

********* 留意計數器原先包裝的方法, 可以作為後面重新包裝的參考 **********

圖二, 鬆脫的PMT接頭

. 圖三, 原先已經包好的閃爍板.

(二)檢測quarknet的光電倍增管(PMT)

零件與材料

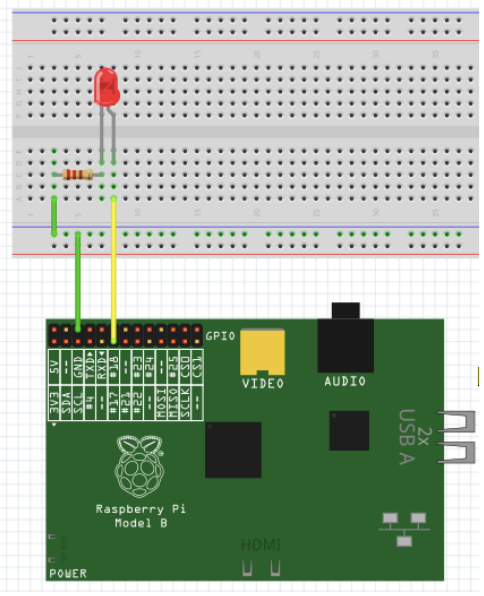
PMT管 + 剪刀 + 圓厚紙板一小片 +拆下來的白色PVC管+拆下來的黑紙與鋁箔紙+膠帶+樹梅派一套+麵包板 + 綠LED 1 + 紅黑長導線各一條+公母杜邦線2條+電阻(1k ohm)+三用電錶+2塊麵包板.

檢測流程

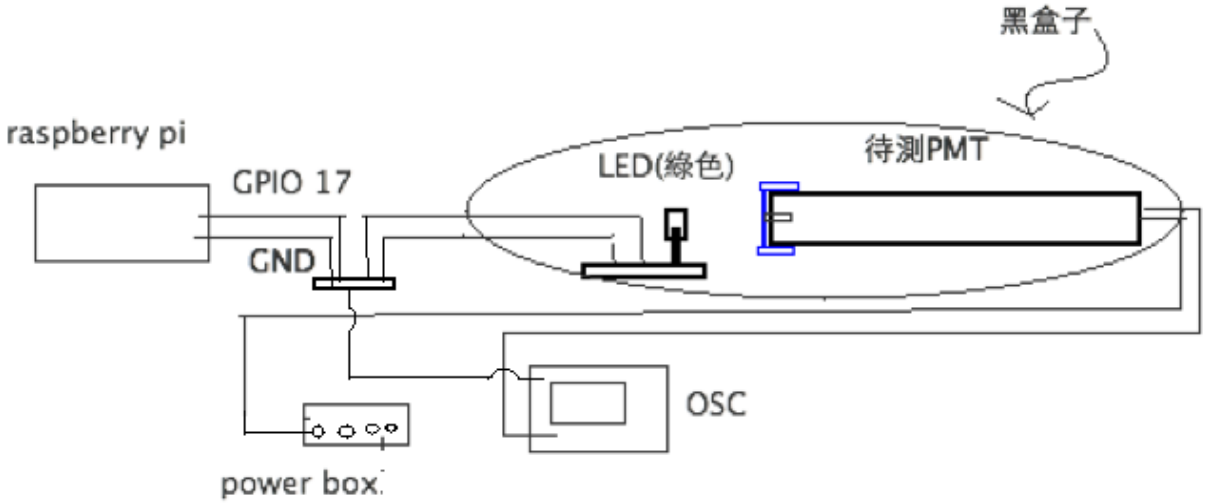
1. **用一般的厚紙板剪成PMT窗戶大小的圓, 中間鑽一小洞, 圓紙板用鋁鉑紙與PMT一起包住,**
   稍微用膠帶固定在PMT上, 以免脫落.
2. 在麵包板上完成LED線路, 電阻用的是1k 歐姆. 連結到樹梅派的第6隻針腳與第11隻針腳(圖
   四).連結的導線要有足夠的長度, 要另外再用杜邦線及麵包板做橋接.
3. 利用quarknet的DAQ卡與電源供應盒給PMT所需要的工作電壓, PMT與LED的線路板放入一無
   光的盒子裡(以白色PVC管與黑紙共同組成), 各部分依照圖五接好. 檢測的過程裡, 樹梅派扮演訊號
   產生器的角色.
4. 程式pwm.py送出高度3.3V週期性的方波(100Hz), 程式裡的x值對應到每個週期裡3.3V方波所
   占的百分比(參考圖六), 下指令sudo python pwm.py 就可以從GPIO17送出 這個可以調方波百分
   比的波形(PWM), 點亮LED燈.

```
# pwm.py
import RPi.GPIO as GPIO
#from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setup(17,GPIO.OUT)
p1 = GPIO.PWM(17,100)
try:
        while True:
                x= float(raw_input("(0~100)brightness= ? "))
                p1.start(x)
except KeyboardInterrupt:
        p1.stop()
   GPIO.cleanup()
```

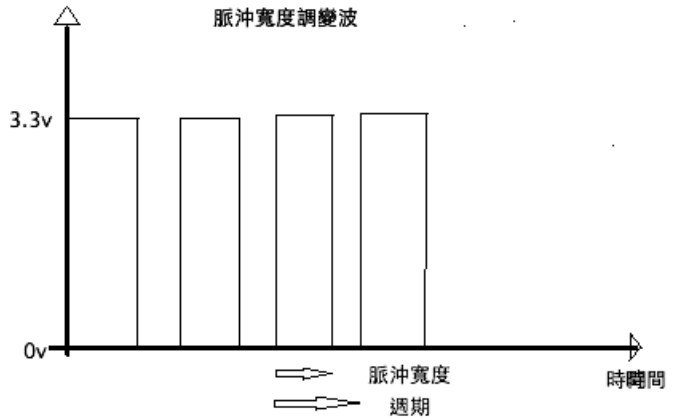
5. LED的亮度呈現PWM一樣的波形變化, 慢慢增加PMT的工作電壓(以三用電錶量測), 觀察在示波器上PMT轉換訊號的情況, 判斷此PMT是否良好. 最後附表是此檢測的總結.



圖四，LED的線路, 第6隻針腳接到電阻, 第11隻針腳接到LED的正極端(長針腳)

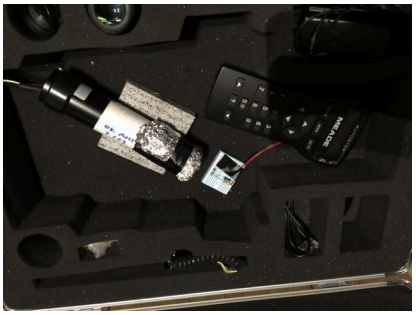
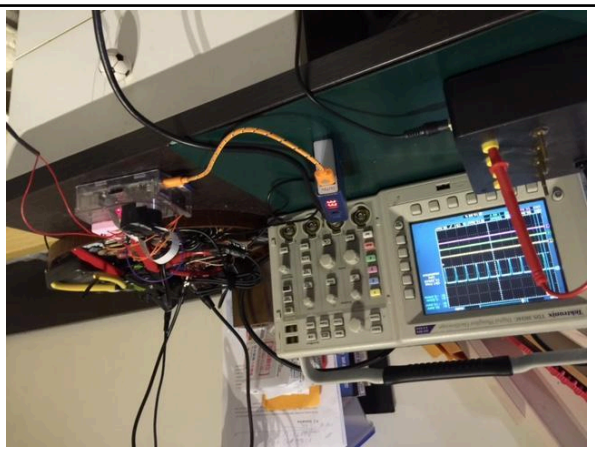


圖五，測試的接線示意圖



圖六，PWM 波形，脈沖寬度由程式pwm.py裡的x值決定.

總結附表

	檢測的對象要放在不透光箱裡, 只拉出幾條連接線.
	箱子打開, 顯示內置物品.
	左邊是示波器(顯示PMT的反應波形), 右邊是樹梅派(送出PWM的檢測波).

(三)組裝計數器

零件與材料：

閃爍板+光電倍增管 +包裝材料+刀片+ 剪刀 + 1”黑膠帶 + 3”黑膠帶 + 一般透明膠帶 + 水龍頭

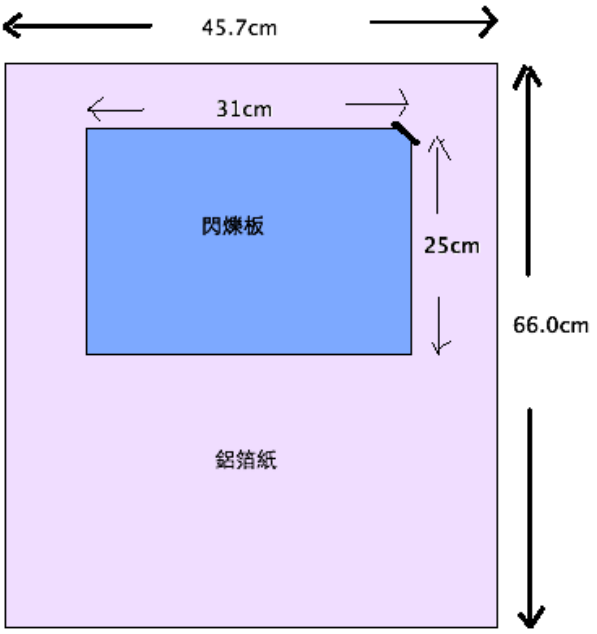
封鐵氟龍白膠布 + 100cm直尺 + 量角器 + 棉手套 + 酒精 + 拭鏡紙 + 壓舌棒 .

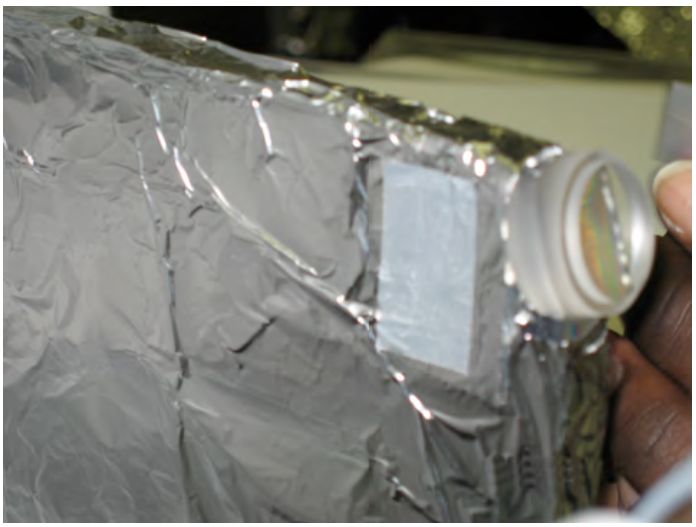
組裝流程

1. 戴上白手套將閃爍板取出, 先放置在乾淨的表面上, 測量閃爍板的常寬高, 並記錄之。

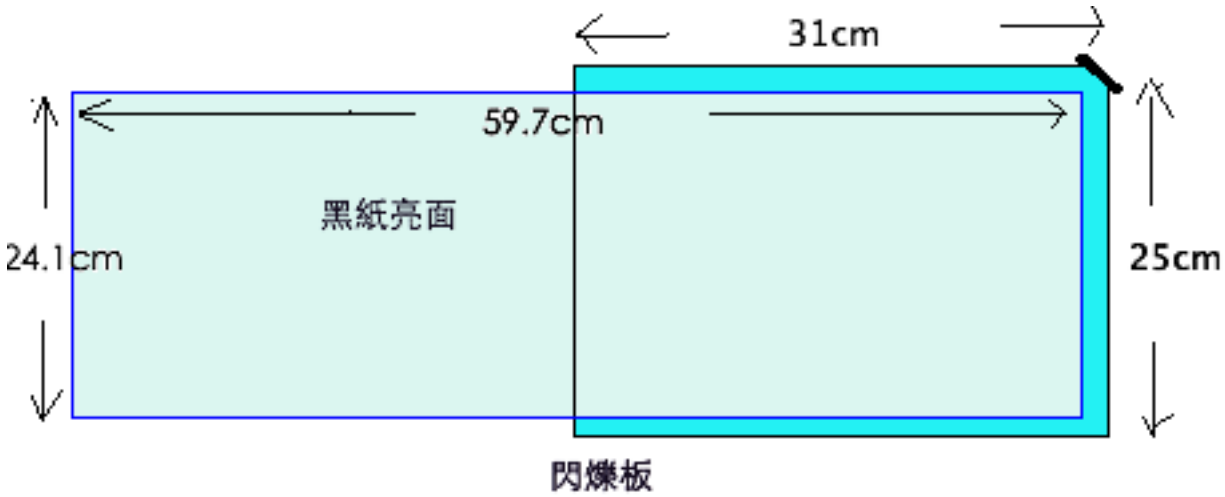
   注意:測量時戴上白手套, 以免手上油污沾在閃爍板上, 影響正式測量時的光線反射。

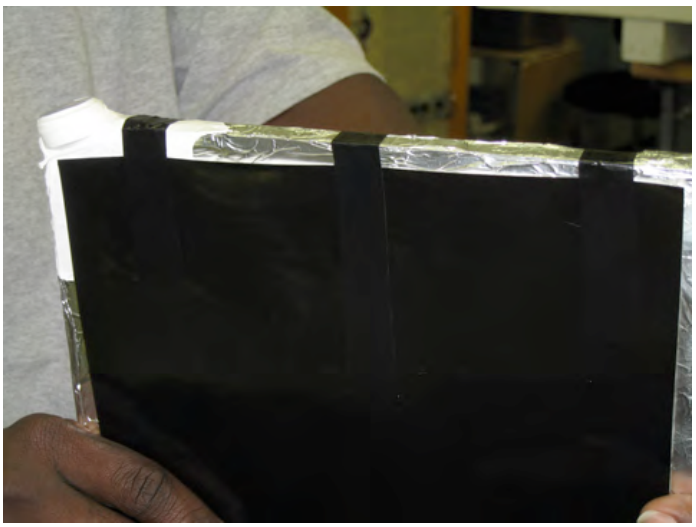
2. 裁切鋁箔紙(45.7cm x 66.0cm), 亮面向上, 先從閃爍板的長邊包起, 然後再在其他邊折好, 角落與邊邊的材料就像包禮物的包裝紙一樣處理. 閃爍板除了cookie外的其他地方全部包覆。用透明膠帶將打開的折邊貼好, cookie後邊用刀片修剪整齊, 讓cookie全部都露出來. 鋁箔紙不需要特別平整, 但是不能有撕破及小洞, 但是可以稍微修補.
3. 白膠帶繞cookie幾圈, 不能包到cookie的正面.


45.7cm
31cm
閃爍板
25cm
66.0cm
鋁箔紙



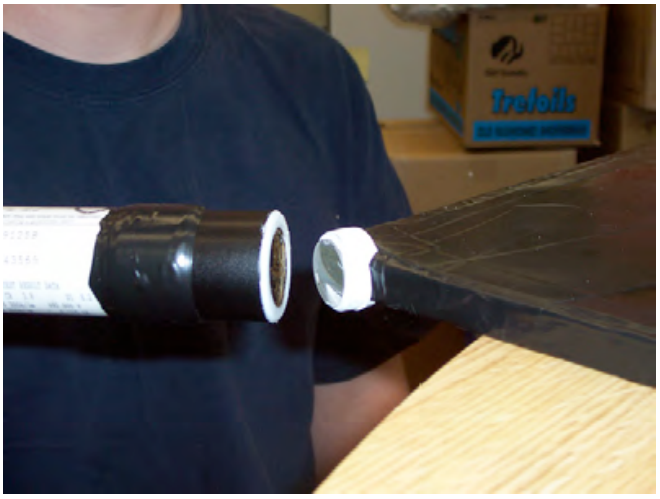
4. 裁切黑紙一張(24.1cm x 59.7cm), 反光面相上, 黑面向下. 將閃爍板放在黑紙上, 沿cookie對面的短邊折起包覆, 用會1”的膠帶將黑紙在4邊貼好固定.



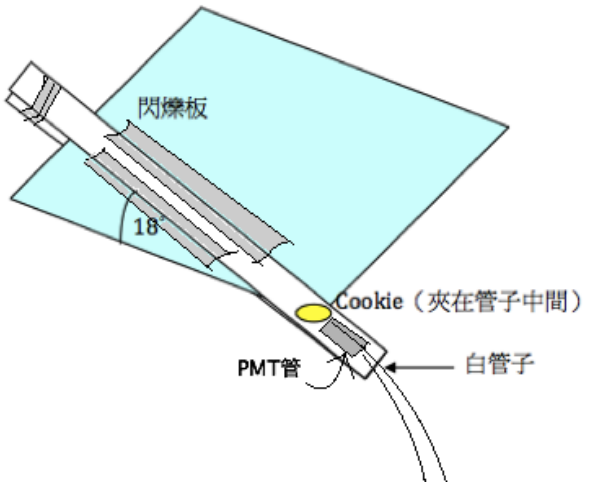


5. 利用1”與3”黑色膠帶將所有縫隙貼死。並需一再檢查所有角落和膠帶邊緣, 不容許任何突起和氣泡。角落地方用膠帶加強封好, cookie後面的部分用3”黑膠帶額外地再斜貼上一層.
6. 沿cookie旁邊與長邊夾18度的直線上貼一條黑膠帶, 這樣當PVC管夾上來的時候, 可以保護閃爍板的包覆層, 兩邊都要這樣做.

7. 將PMT(光電管)前端塗上「光學膠」。
8. 將PMT輕輕地和cookie對正結合，並稍微轉動，以便讓光學膠中的氣泡排出。
9. 用黑膠帶將PMT和cookie接合處黏貼。並需一再檢查，確定位置沒有變歪，不會漏光。



10. 白色PVC管開口的一端沿PMT方向插入, 將閃爍板夾住(注意不能卡住PMT的接線), 使用膠帶貼合閃爍板, 固定PVC管子兩端。PVC管確保之後使用探測器的時候, 不會任意碰到cookie和PMT，避免它們的接合處發生鬆脫的情形。



RP的應用/利用樹莓派的實作IV

夸克網的應用硬體篇

如何改變quarknet的實驗?

2016, 7/15~16兩天的維修工作坊, 我們整理了**11**支**quarknet**計數器, 光電倍增管經過檢測, 確定裡面有**2**支是壞的, 重新組裝了**6**支計數器, 也都通過以示波器做的反應檢測, 這**6**隻計數器必須要再經過**plateau**的檢驗, 得知了工作電壓後, 才能夠正式啟用收取宇宙射線的訊號.
2012年開始, 就以樹梅派小電腦取代一般電腦, 使用程式套件**minicom**, 證明在**quarknet**的應用上, 可以很穩定地長時間取數據. 又因為樹梅派與**quarknet**的**DAQ**卡一樣都是使用**5VDC**, 可以用一般**5V**行動電源啟動, 因此 **2015**年將儀器裝箱, 沿著高速公路, 進行一趟跨越台灣南北的測量. 當然這樣的可行性還需要有**USB**的螢幕與**wifi**連線才能夠實現.

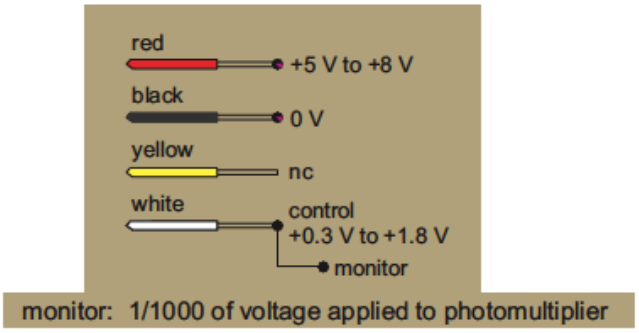
在維修工作坊裡, 我們將樹梅派的角色做了一個提升; 在**7/15**檢測**PMT**的工作裡, 樹梅派化身為訊號產生器, 用**pwm**波形推動**LED**, 作為檢測**PMT**的光源.
能不能夠讓樹梅派做更多的事? 能不能夠提供學生更豐富的學習內容? 能否將實驗的過程更有啓發性? 自動化也許是最佳的做法.
現在全面的自動化已經完成, 以下三節描述硬體方面的配合, 分別是:

一, 如何利用樹梅派量測**quarknet**電源盒的電壓
二, 如何利用樹梅派讀取**PMT**的訊號
三, 自動設定"控制電壓"

以下解釋硬體的部分, 另外有一份資料解釋軟體的部分與自動化的流程!

一, 如何利用樹梅派量測**quarknet**電源盒的電壓

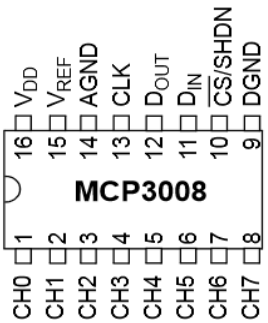
quarknet的電源盒有4個輸出, 分別接到4隻光電倍增管(PMT)的紅黑白3條電線上(參考圖一), 每個輸出有一個固定的5V(紅黑線)與一個可變電壓(白線), 其中白線的控制電壓是要給PMT一個適合的工作電壓, 這是經由轉動電源盒上方的旋鈕來做設定的, 板面上也有測量"控制電壓"的插孔, 一般是用三用電錶來做這個工作, 現在我們的目的是用樹梅派測量這個電源盒的"控制電壓".
然而"控制電壓"是類比訊號, 類比訊號要轉換成數位訊號, 才能被電腦記錄下來, 因此選擇一顆類比數位轉換器(ADC, Analogue to Digital Converter) MCP3008, 這是便宜又被廣泛使用的IC. 參考圖二完成與樹梅派的連線.



圖一, **PMT**的電源線, 用到的是紅黑白**3**條線, 黃線未用.

The following list shows how the MCP3008 can be connected. It requires 4 GPIO pins on the Pi P1 Header.

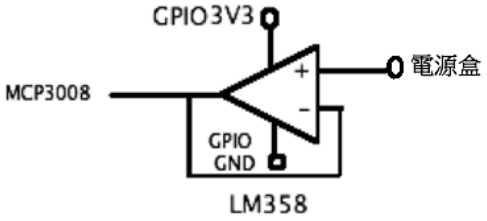
```
VDD   3.3V
VREF  3.3V
AGND  GROUND
CLK   GPIO11 (P1-23)
DOUT  GPIO9  (P1-21)
DIN   GPIO10 (P1-19)
CS    GPIO8  (P1-24)
DGND  GROUND
```

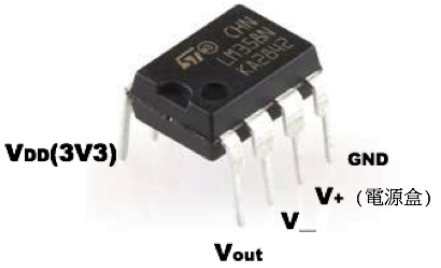


The CH0-CH7 pins are the 8 analogue inputs.

圖二, 右邊是MCP3008的16個針腳的名稱; CH0到CH7接8個外部訊號的部分, 其他的8個針腳分別接到樹梅派的GPIO針腳, 對應的接線如左邊所示. ~~**注意: 2017/7/19更改VDD與VREF使用外部電源(例如, 兩個1.5V的乾電池, 提供3.0V); 圖三裡的3.3V也改用這樣的電源.**~~

但是MCP3008 的輸入電阻相對於"電源盒插口電阻"是較小, 使得MCP3008的讀值會偏低. 為了解決這個問題, 在圖二裡, 我們選擇將電源盒的訊號先通過一個運算放大器(LM358), 而LM358的接法是要作為放大倍率G=1的訊號分隔器, 這樣執行附錄一的程式, voltage.py, 就可以得到準確的"控制電壓"的訊號.





圖三, 左, 運算放大器**LM358**的接線示意圖. 右, **LM358**的外形與接腳的用法.

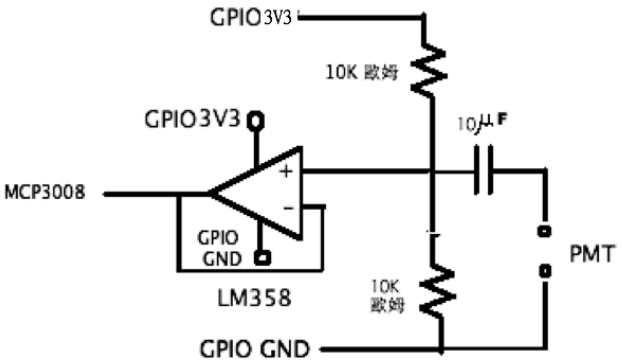
~~注意事項**2021/3/31:** 電源盒上面用三用電錶量測控制電壓的插孔, 對應地線的黑色孔並不是0,V! 而是有一個偏移的正值, 因此正確的PMT電壓執是紅色插孔的電壓質減去黑色插孔的電壓值, 這也是以三用電錶量到的PMT控制電壓值. 因此要將紅色插孔接到MCP3008的0埠, 黑色插孔接到1埠, 然後在程式裡將兩個讀值相減, 才能等於三用電錶的量測.~~

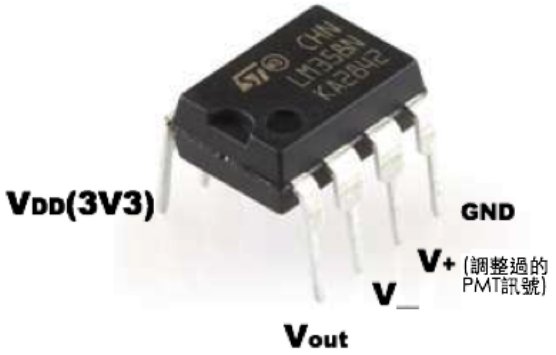
二, 如何利用樹梅派讀取**PMT**的訊號

我們的目的是能夠用樹梅派測量quarknet光電倍增管(PMT)的訊號. 然而我們用示波器檢視PMT, 可以看到PMT輸出訊號是在負電壓的範圍, 但是MCP3008 能夠量測的是只有正電壓訊號, , 因此要將PMT輸入到MCP3008的訊號先做一個處理, 才能讓MCP3008讀取.

依照圖一的線路, 加上了一個3.3V正電壓的偏壓, 當PMT訊號為0V時, 兩個10K歐姆之間的電壓是1.65V, 當訊號源PMT是負電壓的時候, 電容的電壓下降, 兩電阻間電壓會跟著調降, 可以預備輸入到MCP3008, 只要PMT負電壓的值小於正偏壓(1.65V) , 輸入MCP3008的就會是正電壓, MCP3008就可以正常地運作.

但是MCP3008 的輸入電阻相對於"PMT"是較小, 使得MCP3008的讀值會偏低. 為了解決這個問題, 在圖四裡, 我們選擇將調整過的PMT訊號先通過一個運算放大器(LM358), 而LM358的接法是要作為放大倍率G=1的訊號分隔器 , 這樣執行附錄二的程式, SCI.py, 就可以得到準確的PMT的訊號.



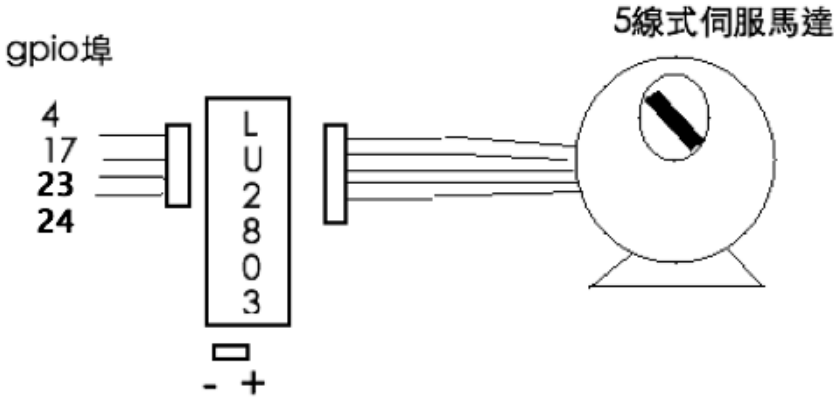


圖四, 左, 以正偏壓處理負電壓的輸入訊號, 然後輸入LM358再輸出到MCP3008以符合MCP3008的要求. 右, LM358的接線圖.
https://rheingoldheavy.com/mcp3008-tutorial-04-sampling-audio-frequency-signals-01/

三, 自動設定"控制電壓"

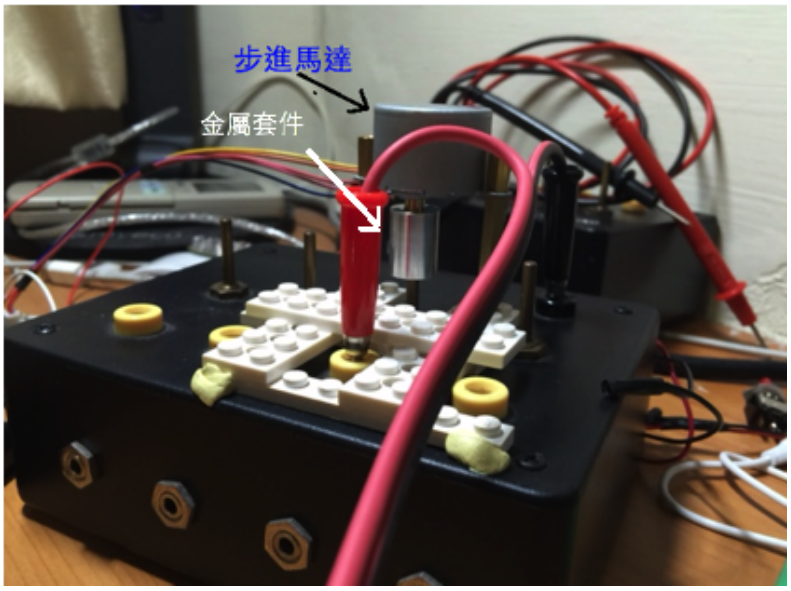
以上用樹梅派讀取"控制電壓"的值, 但還是要用指頭轉動電源盒上的旋鈕. 在這裡我們要用一顆步進馬達透過樹梅派來轉動旋鈕.

精密機器裡常用到步進馬達, 是因為步進馬達能夠一步一步的轉動, 而每一步約只轉不到1度的角度. 我們用的就是其中的一個類型---5線式步進馬達, 先將馬達的排線與驅動器ULN2803接好, 完成gpio針腳4,17,23,24接線, 電源部分可以接Raspi的3.3v來用, 也可以用外接電源. 執行程式(python uln.py), 就會要求輸入每一步間隔的時間是多少毫秒(msec)? 要向前進幾步? 向後幾步? 每次在輸入步數之後, 馬達就會轉動. 程式內容在附錄三.



圖五，5線式步進馬達與樹梅派GPIO的接線示意圖.

我們需要製作一個套件將步進馬達的轉軸與旋鈕結合，完成圖如圖六.



圖六，步進馬達與電源盒的結合.

參考網站:

1. http://www.raspberrypi-spy.co.uk/2013/10/analogue-sensors-on-the-raspberry-pi-using-an-mcp3008/

2. https://vimeo.com/album/4064140

附錄一, voltage.py

```
## 舊版，以Adafruit_GPIO 取代
#import spidev
#import time
#import os

## Open SPI bus
#spi = spidev.SpiDev()
#spi.open(0,0)
## Function to read SPI data from MCP3008 chip
## Channel must be an integer 0-7
#def ReadChannel(channel):
#  adc = spi.xfer2([1,(8+channel)<<4,0])
#  data = ((adc[1]&3) << 8) + adc[2]
#  return data
################################################################################
#######
import Adafruit_GPIO.SPI as SPI
import Adafruit_MCP3008
import os
import time

SPI_PORT=0
SPI_DEVICE=0
mcp = Adafruit_MCP3008.MCP3008(spi=SPI.SpiDev(SPI_PORT, SPI_DEVICE))

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
  data=mcp.read_adc(channel)
  return data
# Function to convert data to voltage level,
# rounded to specified number of decimal places.
def ConvertVolts(data,places):
  volts = (data * 3.3) / 1023
  volts = round(volts,places)
  print "data=",data
  return volts


# Define sensor channels
sensor_channel = 0

# Define delay between readings
delay = 5

while True:

  # Read the light sensor data
```

```
  sensor_level = ReadChannel(sensor_channel)
  sensor_volts = ConvertVolts(sensor_level,2)


  # Print out results
  print "--------------------------------------------"
  print("Sensor : {} ({}V)".format(sensor_level,sensor_votls))

  # Wait before repeating loop
  time.sleep(delay)
```

附錄二, SCI.py

```
import matplotlib
matplotlib.use("TkAgg")
from pylab import *
import numpy as np
import spidev
import time
import os

# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
  adc = spi.xfer2([1,(8+channel)<<4,0])
  data = ((adc[1]&3) << 8) + adc[2]
  return data

# Function to convert data to voltage level,
# rounded to specified number of decimal places.
def ConvertVolts(data,places):
  volts = (data * 3.3) / 1023
  volts = round(volts,places)
  return volts


# Define sensor channels
light_channel = 0

NT=300
t=np.zeros(NT)
for i in range(NT):
```

```
  t[i]=i
ax=np.zeros(NT)

fig = plt.figure()

bx = fig.add_subplot(111)
plt.subplots_adjust(hspace = .2)
li1, = bx.plot(t, ax,'r-')
plt.grid(True)                #Turn the grid on^M
plt.ylabel('V')
plt.ylim([1.1,1.7])
bx.axes.get_xaxis().set_visible(True)
fig.canvas.draw()
plt.show(block=False)


# Read the light sensor data
cond=True
#while cond:
for k in range(1000):
  tin=time.time()
  for i in range(NT):
     light_level = ReadChannel(light_channel)
     x = ConvertVolts(light_level,2)
     ax[i]=x
#  print time.time()-tin
  li1.set_data(t,ax)
  plt.subplot(111)
  fig.canvas.draw()
  time.sleep(0.001)
```

附錄三, **uln.py**

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

enable_pin = 18
coil_A_1_pin = 4
coil_A_2_pin = 17
coil_B_1_pin = 23
coil_B_2_pin = 24

GPIO.setup(enable_pin, GPIO.OUT)
GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
```

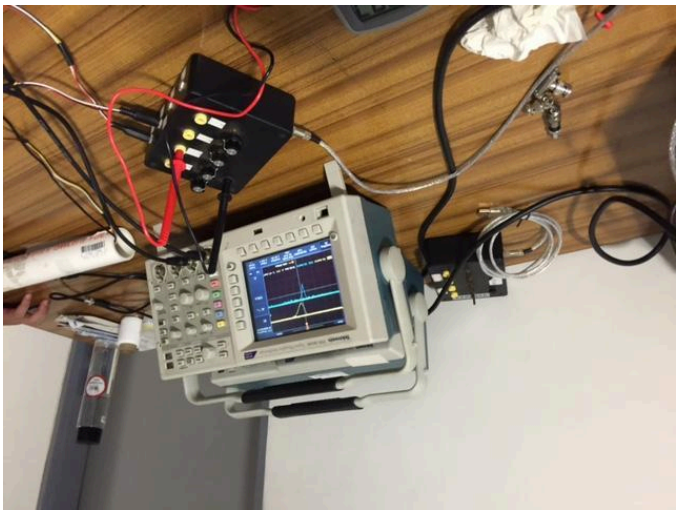
```
GPIO.setup(coil_B_1_pin, GPIO.OUT)
GPIO.setup(coil_B_2_pin, GPIO.OUT)

GPIO.output(enable_pin, 1)
def forward(delay, steps):
  for i in range(0, steps):
    setStep(0, 0, 0, 1)
    time.sleep(delay)
    setStep(0, 0, 1, 1)
    time.sleep(delay)
    setStep(0, 0, 1, 0)
    time.sleep(delay)
    setStep(0, 1, 1, 0)
    time.sleep(delay)
    setStep(0, 1, 0, 0)
    time.sleep(delay)
    setStep(1, 1, 0, 0)
    time.sleep(delay)
    setStep(1, 0, 0, 0)
    time.sleep(delay)
    setStep(1, 0, 0, 1)
    time.sleep(delay)

def backwards(delay, steps):
  for i in range(0, steps):
    setStep(1, 0, 0, 1)
    time.sleep(delay)
    setStep(1, 0, 0, 0)
    time.sleep(delay)
    setStep(1, 1, 0, 0)
    time.sleep(delay)
    setStep(0, 1, 0, 0)
    time.sleep(delay)
    setStep(0, 1, 1, 0)
    time.sleep(delay)
    setStep(0, 0, 1, 0)
    time.sleep(delay)
    setStep(0, 0, 1, 1)
    time.sleep(delay)
    setStep(0, 0, 0, 1)
    time.sleep(delay)


def setStep(w1, w2, w3, w4):
  GPIO.output(coil_A_1_pin, w1)
  GPIO.output(coil_A_2_pin, w2)
  GPIO.output(coil_B_1_pin, w3)
  GPIO.output(coil_B_2_pin, w4)

try:
   while True:
     delay = raw_input("Delay between steps (milliseconds)?")
```

```
    steps = raw_input("How many steps forward? ")
    forward(int(delay) / 1000.0, int(steps))
    steps = raw_input("How many steps backwards? ")
    backwards(int(delay) / 1000.0, int(steps))
except KeyboardInterrupt:
        setStep(0,0,0,0)
```

RP的應用/利用樹莓派的實作v
夸克網的應用/軟體篇

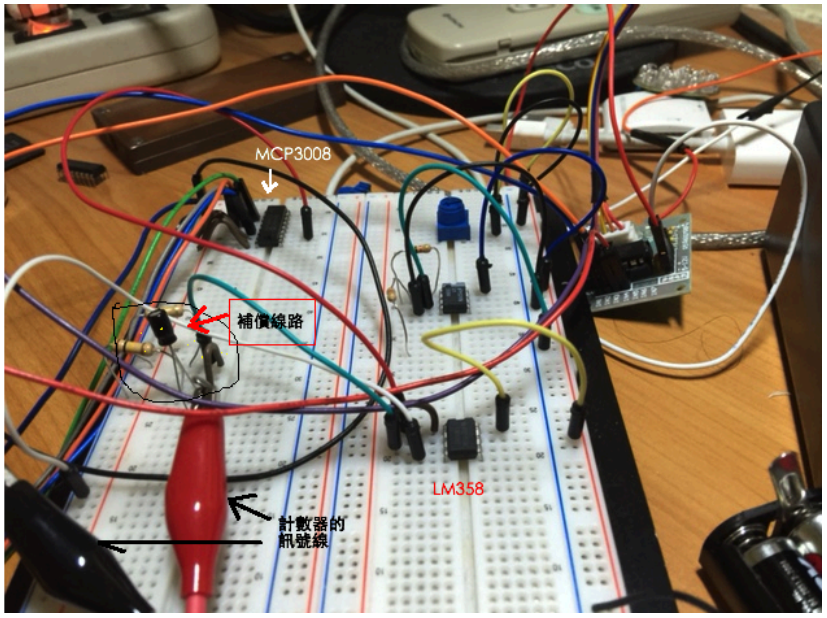
一, 以樹梅派代替示波器檢測計數器
(~/quarknet/PMT-SCI)

計數器組裝好, PMT的訊號線接上示波器, 調整電源盒的控制電壓到一個適當的值 , 宇宙射線穿過閃爍板就會在示波器上出現明顯的脈沖(圖一, 約數10毫伏與數10奈秒 , 負電壓).



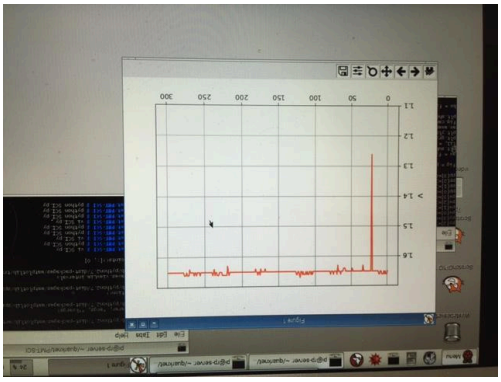
圖一, 計數器輸出宇宙射線的訊號.

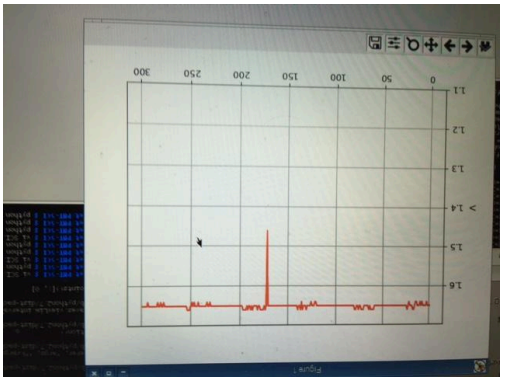
實驗(三)的第2節解釋了如何利用樹梅派扮演示波器的角色, 讀取**PMT**的訊號呈現宇宙射線的脈沖波, 圖二是接好的線路.

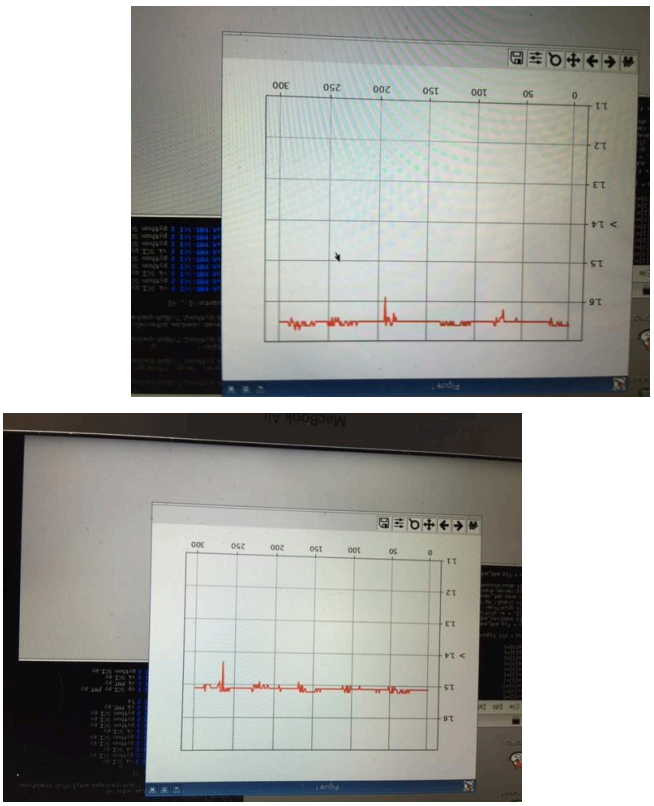


圖二, 樹梅派代替示波器檢測計數器PMT的訊號, 需要的零件有ML385, MCP3008 與一個補償線路.

我們執行一個檢測的程式, **SCI.py**在螢幕上就會出現一序列的快照**(**圖三**),**我們可以看到補償線路將PMT的負電壓訊號提升了1.65V, 達到MCP3008能正確接收的範圍, MCP3008的量測誤差是+/- 3mV, 時間的解析只能到0.01毫秒(**10^{-5} s**), 達不到示波器ns(**10^{-9} s**)的攫取速度, 但是已經能夠檢測與判斷計數器好壞. 完整的程式與說明在附錄一.







圖三, 程式SCI.py 的到一序列的宇宙射線脈沖的快照.

二, 樹梅派與**DAQ**卡的溝通

現在我們開始使用quarknet的DAQ卡了. 一般的通訊軟體(minicom, hyperterminal)都可以讓電腦連上quarknet的DAQ卡, 然後在電腦鍵盤上下達指令(附錄二), 設定取數據的參數與存檔的檔名. 有些資料可以由下指令取得, 例如計數器的 singles 的通量, 或是幾支計數器的多重同時通量等, 其他的分析工具在quarknet的e-Lab 上, 需要先將數據檔上傳到quarknet的 伺服器, 才能夠進行分析知道結果. 為了能夠在收取數據的電腦裡就可以先做數據的分析, 減少等待的時間, 陸陸續續有幾套線上程式的出現, 例如早先有在LABVIEW上執行的程式, 最近的EQUIPT, muonic等, 都是使用者圖形界面的軟體, quarknet的實驗開始有比較多的變化. 我們將quarknet的實驗做了幾項硬體的變動, 因此需要一個能夠與DAQ卡溝通, 又能配合我們硬體環境的程式, 在這個階段, 我們沒有搭配圖形界面的使用方式, 這是為了讓學生充分了解軟硬體之間溝通的方式, 進而培養撰寫程式的能力.
首先, 我們解釋在我們的程式裡與DAQ卡通訊有關的部份;

```
1 from serial import Serial
2 ser = Serial('/dev/ttyUSB0', 115200,
        bytesize=8,parity='N',stopbits=1,timeout=None,xonxoff=False)
```

第1行是引入序列通訊模組 serial 的類別 Serial. 第2行是定義對應到DAQ卡的對象ser 與連線所需要的參數值.
接下來就可以使用ser的方法與DAQ卡進行溝通了.

```
3  ser.write('CD'+'\r'+'WC 00 27'+'\r')
4 time.sleep(1,)
5 ser.flushInput()
```

第3行, \r代表鍵盤上的enter鍵, 加上它才能對DAQ卡送出兩條指令:
CD
WC 00 27
分別代表:counter disable, 三重同時計數與有效的計數器是ch0,ch1與ch2.
第4行是等待1秒, 第5行是清掉任何在DAQ卡裡等待要送出的數據.
由DAQ卡讀取資料,可以如此做:

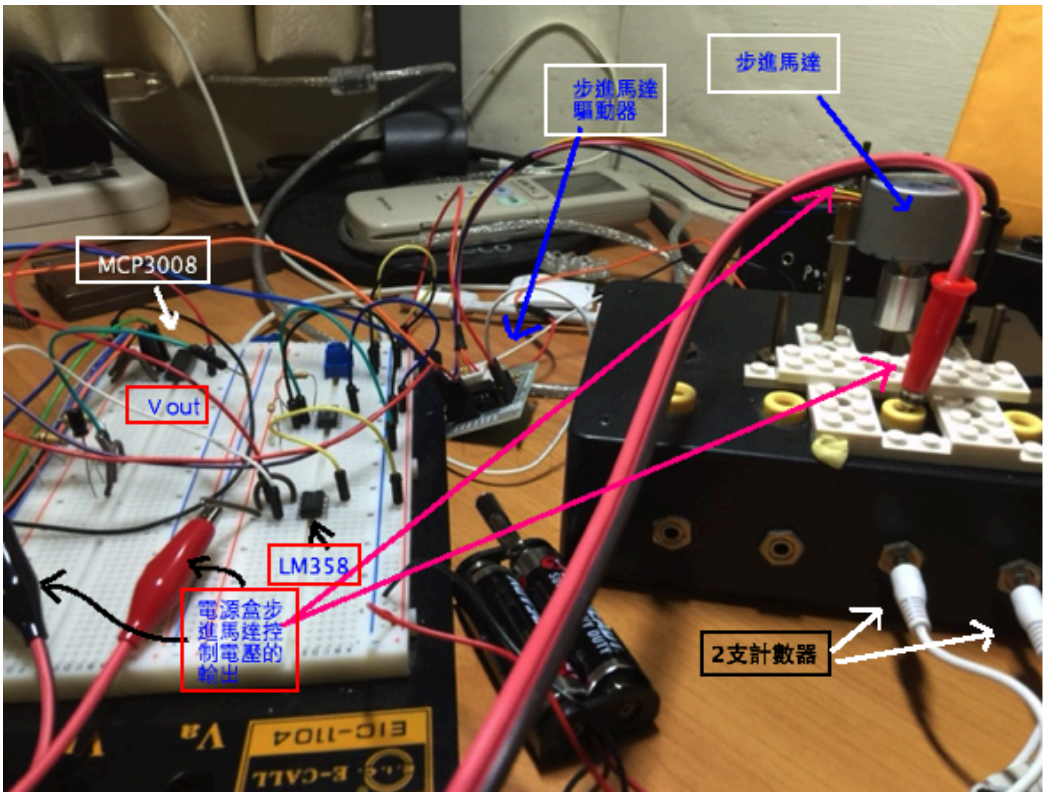
```
6 ser.write('TL 4 '+str(600)+'\r')
7 ser.readline(ser.inWaiting())
8 ser.flushInput()
```

第6行是送出指令 TL 4 600, 設定4支計數器的threshold值是 600mV, 第7行讀DAQ卡的回應,
但是只讀一行(readline), 其他的就在第8行清除掉.
基本上這就是與DAQ卡溝通的方式, 在這個實驗裡, 有更很多的地方都要再用到.

三, 使用計數器的準備工作

(~/quarknet/plateau)

要先知道最佳工作的平坦區域(plateau)的範圍與閥值(threshold), 計數器才算是準備妥當, 才能夠加入探測器的行列. 在這節裏我們要尋找計數器的工作電壓與閥值.
這個工作需要兩支計數器, 先都接上電源盒與DAQ卡, 然後以步進馬達用套件固定在其中一支計數器在電源盒的旋鈕上, 完成如圖四的接線, 步進馬達有獨立的電源.



圖四, 進行計數器準備工作的接線圖

先將沒有接步進馬達的計數器的電壓設定在700mv~800mV之間, 使得這個計數器的通量約為20~30 Hz. 然後使用程式: adjust_volt.py 與 plateau.py. 第1個程式讓我們將步進馬達轉動的控制電壓調到一個電壓值, 第2個程式負責掃描電壓並作圖, 程式說明在附錄三之一～三之四.

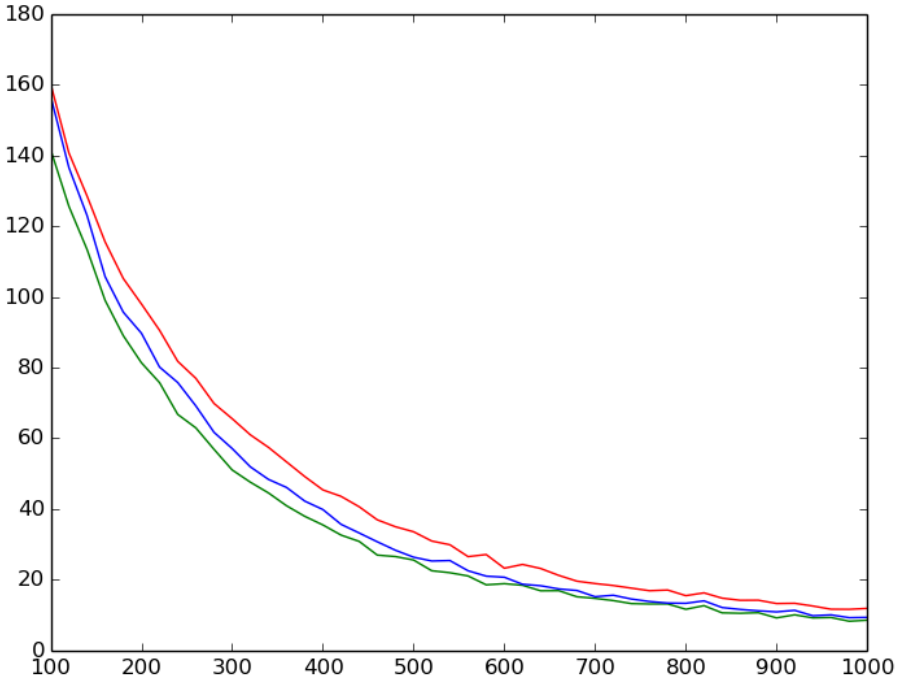
分別用步進馬達掃描這2支計數器的控制電壓, 由單一與雙重同時的通量圖決定工作電壓~~(參考: (三)校正閃爍體計數器)~~.

參考: calibrate-PMT.docx裡面的(三)校正閃爍體計數器.

四, 計數器的閥值
(~/quarknet/threshold)

第三節裡的工作完成後, 我們就有了2支計數器的工作電壓值, 但是用的閥值是任意設定的.

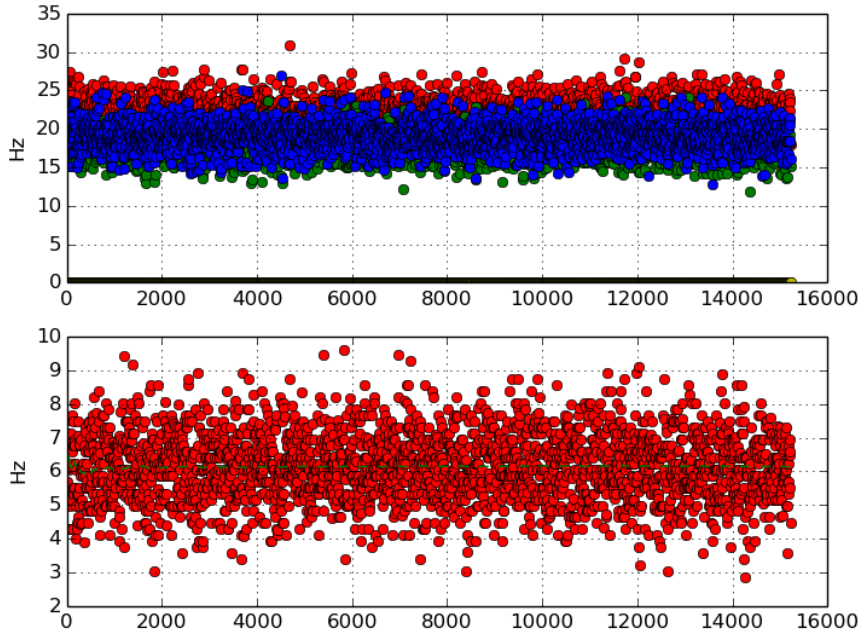
在這裡我們只要將工作電壓設定好, 執行程式threshold.py(附錄四)就可以像DAQ卡下指令, 將閥值做一個掃描, 由取得的數據圖(圖五)判斷最適當的閥值.



圖五，3支計數器的singles-threshold 的圖

五，宇宙射線的通量

有了兩支以上的計數器準備妥當後, 架好GPS, 啟動DAQ卡就組成一套宇宙射線探測器, 用EQUIPT, muonic或minicom就可以正式連線取數據了.
程式flux_PKT.py可以長時間顯示探測器的單一通量與多重同時通量, 如圖六所示. 這個程式的內容與plateau.py, threshold.py都很相似. 因此不再重複說明.



圖六, 3支計數器組成的宇宙射線探測器, 程式flux.py會顯示每一支計數器的單一性通量(上半圖)與 三重同時通量(下半圖), 橫軸是時間(second). 每點是5second的計數得到的平均通量值(Hz), 時間加長可以減少分佈的範圍.

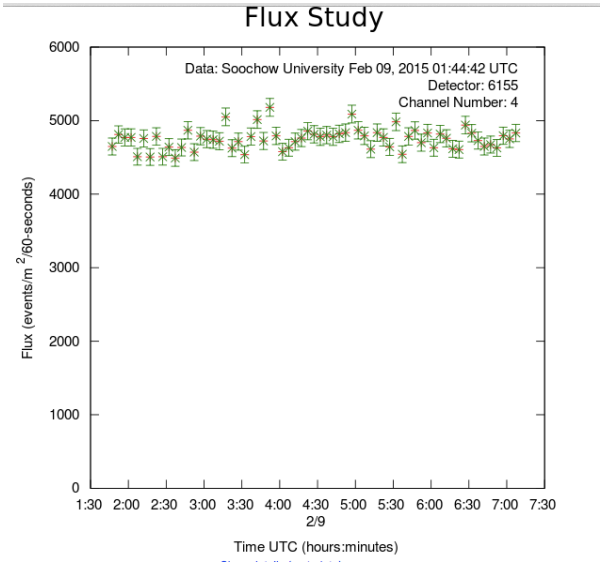
六, 數據存檔

到目前為止, 我們的程式是以指令DS來取得各個計數器的通量值, 但是上傳到quarknet的e-Lab, 必須要下指令,CE(計數器啟動), 數據不斷地由DAQ卡送出, 連線的程式將這些數據統統存檔, 再進入e-Lab的網頁上傳, e-Lab有計算通量的工具, 例如圖七.

附錄程式upload.py 是將下了CE指令的數據快速地存檔. 這裡DAQ卡與樹梅派之間快速交換資料使得失誤的機會增加, 所以程式裡要做處理, 才不會損失過多的有效數據. 執行upload.py的方式是:

```
python upload.py test.txt
```

其中test.txt是要儲存數據的檔名, 而執行中的upload.py 可以用CNTL-C 強迫停止, 留下來的檔案上傳到e-Lab做後續的分析.



.

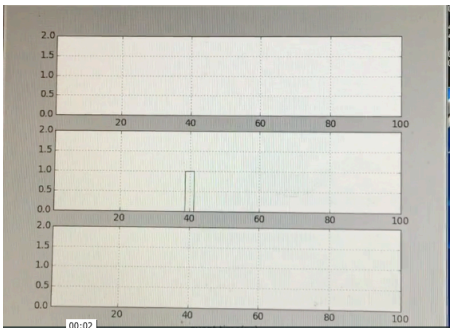
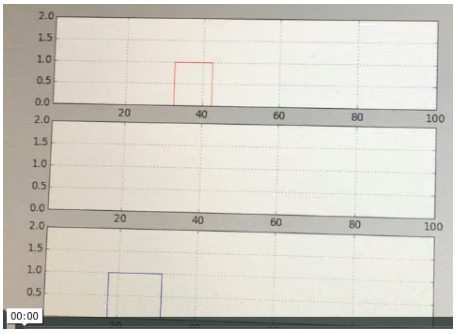
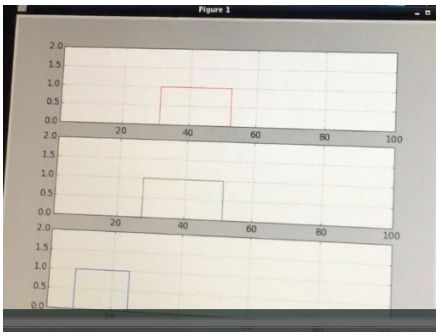
圖七, 夸克網e-Lab的通量工具分析結果, 一次只能繪製一支計數器.

七, 監看數據

DAQ卡處理的數據是以事件為單位, 每個事件又涵蓋好幾行的數據, 這幾行的數據恰好是滿足設好的邏輯參數:

1. 幾重同時性,
2. 訊號延遲後, 是否落在"時間窗戶"的長度內.

因為DAQ卡有標示事件的方法, 我們可以照辦, 將該事件的幾行數據拿處來處理(event.py), 然後將該事件送回主程式(monitor.py)繪圖. 這樣我們就可以坐在一旁觀看計數器此起彼落的訊號, 如圖八.

圖八, 左圖是singles的事件,中圖是雙重同時的事件, 右圖是三重同時的事件.

相應的影片:

https://vimeo.com/178687749
https://vimeo.com/178687867
https://vimeo.com/178687947

附錄一, 解釋SCI.py

~/quarknet/PMT-SCI

```
1. import matplotlib
2. matplotlib.use("TkAgg")
3. from pylab import *
4. import numpy as np
5. import spidev
6. import time
7. import os
```

第1行是將matplotlib模組引入, 第2行 是決定後端繪圖的介面是 TkAgg, 第3~7行是引入其他模組, 其中spidev是與MCP3008溝通的模組.

```
# Open SPI bus
8. spi = spidev.SpiDev()
9. spi.open(0,0)
```

第8~9行 定義spi是一個SPI的裝置及溝通的管道

```
# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
10. def ReadChannel(channel):
11.     adc = spi.xfer2([1,(8+channel)<<4,0])
12.     data = ((adc[1]&3) << 8) + adc[2]
13.     return data
```

定義從MCP3008的通道數目=channel讀取SPI數據的函數, 其中 channel是0至7的整數.
將MCP3008的數據轉換成十進位數值.

```
# Function to convert data to voltage level,
# rounded to specified number of decimal places.
14. def ConvertVolts(data,places):
15.     volts = (data * 3.3) / 1023
16.     volts = round(volts,places)
17.     return volts
```

換成在MCP3008參考電壓範圍(0V-3.3V)的電壓值.

```
# Define sensor channels
18. light_channel = 0
```

第18行對應PMT訊號接到MCP3008的channel 0(=CS0)

```
19. NT=300
20. t=np.zeros(NT)
21. for i in range(NT):
22.     t[i]=i
23. ax=np.zeros(NT)
```

NT是每次快照的數據點數, t是數據點的順序, ax是每點的電壓值.

```
24. fig = plt.figure()
25. bx = fig.add_subplot(111)
26. plt.subplots_adjust(hspace = .2)
27. li1, = bx.plot(t, ax,'r-')
28. plt.grid(True)              #Turn the grid on^M
29. plt.ylabel('V')
30. plt.ylim([1.1,1.7])
31. bx.axes.get_xaxis().set_visible(True)
32. fig.canvas.draw()
33. plt.show(block=False)
```

第24行-第33行是定繪圖的參數.

```
# Read the light sensor data
34. cond=True
35. for k in range(1000):
```

```
36.     tin=time.time()
37.     for i in range(NT):
        light_level = ReadChannel(light_channel)
        x = ConvertVolts(light_level,2)
        ax[i]=x
38.  #  print time.time()-tin
39.       li1.set_data(t,ax)
40.       plt.subplot(111)
41.       fig.canvas.draw()
42.       time.sleep(0.001)
```

快照 1000張, 每張取NT個數據點.

附錄二 **DAQ**卡的指令集

H1 or H2　顯示所有的指令與簡短的說明.
DG　顯示GPS的數據資料
RB　DAQ卡上面的計數器歸零
RE　重新啟動DAQ卡, 所有設定回到預設值.

自探測器各計數器取得數據:
　方法一**:** 至少每一分鐘取一次整體性的數據
ST　　馬上送出狀態列
ST 0　狀態列 Disabled
ST 1　狀態列 Enabled
ST 1 m　DAQ卡每 m分鐘送出一次狀態列
ST 2 m　每次狀態列之後都要列出每個計數器的總計數值
ST 3 m　每次狀態列之後都要列出每個計數器的總計數值, 並將計數(DS)歸零.
　方法二**:** 連續不斷地送出數據
CE　啟動計數器 , 數據開始連續不斷地出現在螢幕上.
CD　停止計數器 , 數據停止出現.

WC 00 ##
　　第1個#代表"幾多重同時"取的數據; #=0是"單一", #=1是"雙重", #=2是"三重", #=3是"四重".
　　第2個#是0至F的16進位值, 代表有哪幾支接在DAQ卡上的計數器(ch0~ch3)是有效的, 計數器分別是
　　1,2,4,8的值, 因此 例如#=B=11=1+2+8 代表設定ch0, ch1,ch3為有效

以下兩行是要設定時間延遲量**:**
WT 01 00
WT 02 78
　上面的 0078 是 16進位 換成十進位 120 □120cnts x 10 ns/cnt = 1200 ns (建議量測大氣簇射時使用).
　　其中10 ns/cnt是DAQ卡的速度 is the speed of the card (100 MHz □ T = 10 ns)
　　注意**:** 緲子生命期建議使用 50 ns,
　　　量測宇宙射線通量建議使用**40 ns.**
以下兩行是要設定時間窗戶的長度**:**

WC 02 cd
WC 03 ab

例如: 量測大氣簇射需要 2400 ns = 240 cnts x 10 ns/cnt □ F0 , 因此這兩行是

WC 02 F0
WC 03 00

緲子生命期需要 10000 ns = 1000 x 10 ns = 3E8 , 因此這兩行是

WC 02 E8
WC 03 03

宇宙射線通量需要 100ns=10x10ns = 0A, 因此這兩行是

WC 02 0A
WC 03 00

指令總集:
H1 Help One
H2 Help Two
DG Get GPS Data
DS Scalar Counts
DC Get Configuration Information
DT Time Control
BA Get Barometer
TH Get Temperature
TI Get Time
V1 View Registers
V2 View Voltage
RB Reset Board (counter)
CE Enable counter to begin data stream (or use Method #1 above) (**Immediately** after RB command)
(Allow data to stream as long as desired)
CD Disable counter to stop data stream
DG Get GPS Data (immediately after CD)
SA n Save setup, 0=(TMC disable), 1=(TMC enable), 2=(Restore Defaults).
TL c d Threshold Level, signal ch(0-3)(4=setAll), data(0-4095mV.

附錄三之一, adjust.py

~/quarknet/plateau

```
1. from voltage import *
2. from uln2803 import *
3. volt_want=input('voltage(mV)=?')
4. volt_now=1000*read_voltage()
  if(volt_want==0.0):
    print volt_now
  else:
    check=True
    delta = volt_want - volt_now
    if(abs(delta) < 5):check=False
```

```
    while check:
      if(delta > 0.0):steps=2
      if(delta < 0.0):steps=-2
      turn_voltage(10,steps)
      volt_now=1000*read_voltage()
      delta=volt_want-volt_now
      print delta
      if(abs(delta) < 5):check=False
  print 'stop',volt_want, volt_now
```

引入兩個模組,**voltage** 與**uln2803**, 其中**voltage**指的是**voltage.py**, **uln2803**指的是**uln2803.py**, 都與**adjust.py**在同一個目錄區裡. **turn_voltage**是**uln2803.py**裡的一個函數, 負責轉動步進馬達, **read_voltage**是**voltage.py**裡的一個函數, 負責讀控制電壓的值. 利用這個程式就可以將電壓定位. 第**3**行要求輸入想要設定的電壓值**volt_want**, 如果輸入值是**0**, 表示只要取得現在的電壓值. 接下來程式會進行掃描並定位, 在接近**5mV**範圍時停止.
注意: 步進馬動轉一圈要動**515**步.

附錄三之二, voltage.py

~/quarknet/plateau

這個程式基本上是與**SCI.py**的第**8**行至第**18**行是一樣的, 只有這裡的第**21**行至第**27**行是做了一個取**500**個數據後的平均值.

```
1.  import spidev
2.  import time
3.  import os

4.  # Open SPI bus
5.  spi = spidev.SpiDev()
6.  spi.open(0,0)

7.  # Function to read SPI data from MCP3008 chip
8.  # Channel must be an integer 0-7
9.  def ReadChannel(channel):
10.   adc = spi.xfer2([1,(8+channel)<<4,0])
11.   data = ((adc[1]&3) << 8) + adc[2]
12.   return data

13. # Function to convert data to voltage level,
14. # rounded to specified number of decimal places.
15. def ConvertVolts(data,places):
16.   volts = (data * 3.3) / 1023
17.   volts = round(volts,places)
18.   return volts


19. # Define sensor channels
20. light_channel = 0
```

```
21.  def read_voltage():
22.    vav=0.0
23.    for i in range(500):
24.      light_level = ReadChannel(light_channel)
25.      light_volts = ConvertVolts(light_level,3)
26.      vav += light_volts
27.  return vav/500
```

附錄三之三, uln2803.py

~/quarknet/plateau

這個程式控制步進馬達的轉動, 只要確定第**1~4**行的**pin**的數目是與實際**GPIO**的接腳是一致的就好**.**

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

1.    coil_A_1_pin = 4
2.    coil_A_2_pin = 17
3.    coil_B_1_pin = 23
4.    coil_B_2_pin = 24

GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
GPIO.setup(coil_B_1_pin, GPIO.OUT)
GPIO.setup(coil_B_2_pin, GPIO.OUT)

def forward(delay, steps):
  for i in range(0, steps):
    setStep(0, 0, 0, 1)
    time.sleep(delay)
    setStep(0, 0, 1, 1)
    time.sleep(delay)
    setStep(0, 0, 1, 0)
    time.sleep(delay)
    setStep(0, 1, 1, 0)
    time.sleep(delay)
    setStep(0, 1, 0, 0)
    time.sleep(delay)
    setStep(1, 1, 0, 0)
    time.sleep(delay)
    setStep(1, 0, 0, 0)
    time.sleep(delay)
    setStep(1, 0, 0, 1)
    time.sleep(delay)
def backwards(delay, steps):
```

```
  for i in range(0, steps):
    setStep(1, 0, 0, 1)
    time.sleep(delay)
    setStep(1, 0, 0, 0)
    time.sleep(delay)
    setStep(1, 1, 0, 0)
    time.sleep(delay)
    setStep(0, 1, 0, 0)
    time.sleep(delay)
    setStep(0, 1, 1, 0)
    time.sleep(delay)
    setStep(0, 0, 1, 0)
    time.sleep(delay)
    setStep(0, 0, 1, 1)
    time.sleep(delay)
    setStep(0, 0, 0, 1)
    time.sleep(delay)

def  setStep(w1,w2, w3, w4):
  GPIO.output(coil_A_1_pin, w1)
  GPIO.output(coil_A_2_pin, w2)
  GPIO.output(coil_B_1_pin, w3)
  GPIO.output(coil_B_2_pin, w4)

#delay = raw_input("Delay between steps (milliseconds)?")

def turn_voltage(delay,steps):
    if steps>0:
      forward(int(delay) / 1000.0, int(steps))
    else:
      steps=-steps
      backwards(int(delay) / 1000.0, int(steps))
```

附錄三之四，plateau.py

~/quarknet/plateau

```
from serial import Serial
from voltage import *
from uln2803 import *
import sys
import time
import datetime
import matplotlib
matplotlib.use("TkAgg")
import matplotlib.pyplot as plt
import numpy as np
```

引入各種模組，附錄三之二與三之三的程式也在這裡引入.

```
ser = Serial('/dev/ttyUSB0', 115200,
bytesize=8,parity='N',stopbits=1,timeout=None, xonxoff=False)
ser.write('CD'+'\r'+'WC 00 13'+'\r')
time.sleep(1,)
ser.flushInput()
ser.write('ST 0'+'\r')
time.sleep(1)
ser.flushInput()
ser.write('TL 4 '+str(600)+'\r')
ser.readline(ser.inWaiting())
ser.flushInput()

fig = plt.figure()
px =[]
py =[]
pz =[]
pw =[]
ps =[]
pp =[]

ax = fig.add_subplot(211)
#plt.subplots_adjust(hspace = .001)
li1, = ax.plot(px, py,'ro')
li2, = ax.plot(px, pz,'go')
li3, = ax.plot(px, pw,'bo')
#li4, = ax.plot(px, ps,'yo')
plt.grid(True)              #Turn the grid on
plt.ylabel('Hz')
ax.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
# draw and show it
fig.canvas.draw()
plt.show(block=False)

bx = fig.add_subplot(212)
plt.subplots_adjust(hspace = .2)
li5, = bx.plot(px, pp,'ro')
bx.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
plt.grid(True)              #Turn the grid on
plt.ylabel('Hz')
fig.canvas.draw()
plt.show(block=False)

tin=time.time()
ser.write('RB'+'\r')
time.sleep(0.5)
ser.readline(ser.inWaiting())
ser.flushInput()
flux1=0.0
flux2=0.0
flux3=0.0
```

DAQ卡上接了2支計數器，先建立與DAQ卡的通訊，
然後下指令:
CD
WC 00 13
ST 0
TL 4 600
要求雙重同時.
注意有些指令, DAQ卡會有回應, 所以要做對應的處理.

數據繪圖的部分, 要預做設定.
圖有上圖(211)及下圖(212), 上圖是3支計數器的單一通量
下圖是雙重同時的通量.

```
flux4=0.0
flux5=0.0
cond=True
print ' avflux:   ch0     ch1     ch2     ch3    c.c. '
# the main sensor reading and plotting loop
while cond:
 turn_voltage(1,5)
 volt=read_voltage()
 if (volt<0.4) or (volt>1.5):
  cond=False
  name=input('file name and file type=')
  plt.savefig(name)
 else:
  tin=time.time()
  time.sleep(5)
  ser.write('DS'+'\r')
  time.sleep(0.1)
  ser.readline(ser.inWaiting())
  q=ser.readline(ser.inWaiting()).strip("\r\n")
  dt=time.time()-tin
  pt=str(q[0:6])
  if pt != 'DS S0=':
     print i,pt
     ser.flushInput()
  else:
     q1=q[6:14]
     q2=q[18:26]
     q3=q[30:38]
     q4=q[42:50]
     q5=q[54:62]
     z1=float(int(q1,16))/dt
     z2=float(int(q2,16))/dt
     z3=float(int(q3,16))/dt
     z4=float(int(q4,16))/dt
     z5=float(int(q5,16))/dt
     ser.write('RB'+'\r')
     time.sleep(0.5)
     ser.readline(ser.inWaiting())
    print volt,' ',z1,' ',z2,' ',z3,' ',z5
    px.append(volt)
    py.append(z1)
    pz.append(z2)
    pw.append(z3)
#   ps.append(z4)
    pp.append(z5)

    li1.set_data(px,py)
    li2.set_data(px,pz)
    li3.set_data(px,pw)
#li4.set_data(px,ps)
```

控制電壓限制在400mV與1500mV之間，
每次取通量值後轉動5步. 程式停止執行後，
輸入要存檔的檔名，輸入的格式是 “filename.png”
，
要以引號包括之.

這裡輸入多久取通量值一次，一般60sec或300sec.
如果只是做測試, 用5sec也可以.

將收取到的數串q做分割，使之對應到各計數器的
單一與雙重同時, 並且計算得到通量
z1,z2,z3,z4,z5.

將數據點繪圖

```
    plt.subplot(211)
    plt.gca().relim()
    plt.gca().autoscale_view()
    fig.canvas.draw()

    li5.set_data(px,pp)
    plt.subplot(212)
    plt.gca().relim()
    plt.gca().autoscale_view()
    fig.canvas.draw()

ser.close()
```

執行完畢, 將serial裝置關閉

附錄四, threshold.py

~/quarknet/threshold

```
from serial import Serial
import sys
import time
import datetime
from pylab import plot,show

ser = Serial('/dev/ttyUSB0', 115200,
bytesize=8,parity='N',stopbits=1,timeout=None, xonxoff=False)
ser.write('CD'+'\r'+'WC 00 27'+'\r')
time.sleep(1,)
ser.flushInput()
ser.write('ST 0'+'\r')
time.sleep(1)
ser.flushInput()
mintl=100
CHUNK = 46
px=[]
py=[]
pz=[]
pw=[]
ps=[]

tin=time.time()
ser.write('RB'+'\r')
time.sleep(0.5)
ser.readline(ser.inWaiting())


for i in range(CHUNK):
        vtl=mintl+i*20
        ser.write('TL 4 '+str(vtl)+'\r')
        time.sleep(0.5)
```

閥值TL 由100mV每20mV掃描一次,
一共掃描46次.

```
        ser.flushInput()
        time.sleep(60)
        ser.write('DS'+'\r')
        time.sleep(0.5)
        ser.readline(ser.inWaiting())
        q=ser.readline(ser.inWaiting()).strip("\r\n")
        pt=str(q[0:6])
        if pt != 'DS S0=':
          print i,pt
          ser.flushInput()
        else:
          dt=time.time()-tin
          q1=q[6:14]
          q2=q[18:26]
          q3=q[30:38]
          q4=q[54:62]
          z1=float(int(q1,16))/dt
          z2=float(int(q2,16))/dt
          z3=float(int(q3,16))/dt
#         z4=float(int(q4,16))/dt
#         print i, str(q)
          print  vtl,' ',z1,' ',z2,' ',z3
          ser.write('RB'+'\r')
          time.sleep(0.5)
          ser.readline(ser.inWaiting())
          tin=time.time()
          px.append(vtl)
          py.append(z1)
          pz.append(z2)
          pw.append(z3)
#         ps.append(z4)
#plot(px,py,'r',px,pz,'g',px,pw,'b',px,ps,'y')
plot(px,py,'r',px,pz,'g',px,pw,'b')
show()
name=input('file name and file type=')
plt.savefig(name)

ser.close()
```

程式停止執行後,
輸入要存檔的檔名，輸入的格式是 “filename.png”
,
要以引號包括之.

附錄五, **flux.py**
~/quarknet/flux

```
from serial import Serial
import sys
import time
import datetime
import matplotlib.pyplot as plt
import numpy as np
```

```
ser = Serial('/dev/ttyUSB0', 115200,
bytesize=8,parity='N',stopbits=1,timeout=None,xonxoff=False)
ser.write('CD'+'\r'+'WC 00 27'+'\r')
time.sleep(1,)
ser.flushInput()
ser.write('ST 0'+'\r')
time.sleep(1)
ser.flushInput()
ser.write('TL 4 '+str(600)+'\r')
ser.readline(ser.inWaiting())
ser.flushInput()

# init plot data
px =[]
py =[]
pz =[]
pw =[]

ps =[]
pp =[]
pu =[]

fig = plt.figure()
ax = fig.add_subplot(211)
plt.subplots_adjust(hspace = .2)
li1, = ax.plot(px, py,'ro')
li2, = ax.plot(px, pz,'go')
li3, = ax.plot(px, pw,'bo')
li4, = ax.plot(px, ps,'yo')
plt.grid(True)                #Turn the grid on
plt.ylabel('Hz')
ax.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
# draw and show it
fig.canvas.draw()
plt.show(block=False)


bx = fig.add_subplot(212)
plt.subplots_adjust(hspace = .2)
li5, = bx.plot(px, pp,'ro')
li6, = bx.plot(px, pu,'g--')
plt.grid(True)                #Turn the grid on
plt.ylabel('Hz')
ax.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
# draw and show it
fig.canvas.draw()
plt.show(block=False)

CHUNK = 5
```

```
tin=time.time()
ser.write('RB'+'\r')
time.sleep(0.5)
ser.readline(ser.inWaiting())
ser.flushInput()
runtime=0.0
flux=0.0
iflux=0
print ' runtime=     ch0=      ch1=      ch2=     ch3=    c.c. ='

# the main sensor reading and plotting loop
try:
  while True:
    for i in range(CHUNK):
       time.sleep(5)
       ser.write('DS'+'\r')
       time.sleep(0.1)
       ser.readline(ser.inWaiting())
       q=ser.readline(ser.inWaiting()).strip("\r\n")
       pt=str(q[0:6])
       if pt != 'DS S0=':
          print i,pt
          ser.flushInput()
       else:
          dt=time.time()-tin
          runtime += dt
          q1=q[6:14]
          q2=q[18:26]
          q3=q[30:38]
          q4=q[42:50]
          q5=q[54:62]
          z1=float(int(q1,16))/dt
          z2=float(int(q2,16))/dt
          z3=float(int(q3,16))/dt
          z4=float(int(q4,16))/dt
          z5=float(int(q5,16))/dt
          flux=flux+z5
          iflux=iflux+1
          avflux=flux/iflux
          print i, str(q)
          print ' ', runtime,'  ',z1,'  ',z2,' ',z3,' ',z4,' ',z5
          print ' average trigered flux', avflux
          ser.write('RB'+'\r')
          tin=time.time()
          time.sleep(0.5)
          ser.readline(ser.inWaiting())

          px.append(runtime)
          py.append(z1)
          pz.append(z2)
```

```
            pw.append(z3)
            ps.append(z4)
            pp.append(z5)
            pu.append(avflux)


        li1.set_data(px,py)
        li2.set_data(px,pz)
        li3.set_data(px,pw)
        li4.set_data(px,ps)
        plt.subplot(211)
        plt.gca().relim()
        plt.gca().autoscale_view()
        fig.canvas.draw()
        li5.set_data(px,pp)
        li6.set_data(px,pu)
        plt.subplot(212)
        plt.gca().relim()
        plt.gca().autoscale_view()
        fig.canvas.draw()

except KeyboardInterrupt:
  name=input('file nam and file typee=')
  plt.savefig(name)
  ser.close()
```

附錄六, upload.py

```
from serial import Serial
import sys
import time
import datetime
from sys import argv
script,filename=argv
print "We're going to erase %r." % filename
print "If you don't want that, hit CTRL-C (^C)."
print "If you do want that, hit RETURN."

raw_input("?")

print "Opening the file..."
target = open(filename, 'wr')

print "cleanning the file !"
target.truncate()

ser = Serial('/dev/ttyUSB0', 115200,
bytesize=8,parity='N',stopbits=1,timeout=None,xonxoff=False)
ser.write('CD'+'\r'+'WC 00 07'+'\r')
```

準備要寫入數據的檔案

```
time.sleep(1,)
ser.flushInput()
ser.write('ST 0'+'\r')
time.sleep(1)
ser.flushInput()
ser.write('TL 4 '+str(600)+'\r')
ser.readline(ser.inWaiting())
ser.flushInput()
tin=time.time()
ser.write('RB'+'\r')
time.sleep(0.5)
ser.readline(ser.inWaiting())
ser.flushInput()

i=0
# the main sensor reading and plotting loop
ser.write('CE'+'\r')
time.sleep(0.5)
ser.readline(ser.inWaiting())
ser.flushInput()

try:
  while True:
       time.sleep(0.01)
       q=ser.readline(ser.inWaiting()).strip("\r\n")
       checksum=len(q)
       if checksum == 72:
          target.write(q)
          target.write('\r\n')
          print q
#          print 'checkm=', checksum-72
       else:
          ser.flushInput()
          print 'may not be right #',i,q
          i += 1
except KeyboardInterrupt:
  taget.close()
  ser.close()
```

與DAQ卡溝通

處理讀入的數據

附錄七之一, monitor.py

~/quarknet/monitor

```
from serial import Serial
import sys
import time
import datetime
import event
```

引入各種模組，import event 是引入附錄七之二的event.py

```
import matplotlib
matplotlib.use("TkAgg")
import matplotlib.pyplot as plt
import numpy as np

ser = Serial('/dev/ttyUSB0', 115200,
          bytesize=8,parity='N',stopbits=1,timeout=None,xonxoff=False)
ser.write('CD'+'\r'+'WC 00 27'+'\r')
time.sleep(1,)
ser.flushInput()
ser.write('ST 0'+'\r')
time.sleep(1)
ser.flushInput()
ser.write('TL 4 '+str(600)+'\r')
ser.readline(ser.inWaiting())
time.sleep(1)
ser.flushInput()

# init plot data
px=np.zeros(4)
py=np.zeros(4)
pz=np.zeros(4)
ps=np.zeros(4)
pw=np.zeros(4)
pp=np.zeros(4)
py[0]=0
py[1]=1
py[2]=1
py[3]=0
pw[0]=0
pw[1]=1
pw[2]=1
pw[3]=0
pp[0]=0
pp[1]=1
pp[2]=1
pp[3]=0
fig = plt.figure()

ax = fig.add_subplot(311)
plt.subplots_adjust(hspace = .2)
li1, = ax.plot(px, py,'r-')
plt.grid(True)                #Turn the grid on
plt.xlabel('event time(ns)')
plt.xlim([1,100])
plt.ylim([0,2])
ax.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
fig.canvas.draw()
plt.show(block=False)
```

與DAQ卡溝通

設定繪圖的項目, 預定的脈沖高度為1, 寬度是取決于event.py 攫取到訊號的
上升邊(rising edge, RE)及下降邊(falling edge, FE)的時間標簽.

```
bx = fig.add_subplot(312)
plt.subplots_adjust(hspace = .2)
li2, = bx.plot(pz, pw,'g-')
plt.grid(True)                #Turn the grid on
plt.xlabel('event time(ns)')
plt.xlim([1,100])
plt.ylim([0,2])
bx.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
# draw and show it
fig.canvas.draw()
plt.show(block=False)

cx = fig.add_subplot(313)
plt.subplots_adjust(hspace = .2)
li3, = cx.plot(ps, pp,'b-')
plt.grid(True)                #Turn the grid on
plt.xlabel('event time(ns)')
plt.xlim([1,100])
plt.ylim([0,2])
cx.axes.get_xaxis().set_visible(True)    #6/16 hsiao xx-> ax
# draw and show it
fig.canvas.draw()
plt.show(block=False)

ser.write('CE'+'\r')
time.sleep(1)
ser.readline(ser.inWaiting())
ser.flushInput()
cond=True
try:
  while cond:
       RE,FE=event.rise_fall(ser)
       px[0]=RE[0]
      px[1]=RE[0]
       px[2]=FE[0]
       px[3]=FE[0]
       pz[0]=RE[1]
          pz[1]=RE[1]
          pz[2]=FE[1]
          pz[3]=FE[1]
          ps[0]=RE[2]
          ps[1]=RE[2]
          ps[2]=FE[2]
          ps[3]=FE[2]

          li1.set_data(px,py)
          plt.subplot(311)
#         plt.gca().relim()
#         plt.gca().autoscale_view()
          fig.canvas.draw()
```

呼叫 event.rise_fall的到每個event的細節

```
            li2.set_data(pz,pw)
            plt.subplot(312)
#           plt.gca().relim()
#           plt.gca().autoscale_view()
            fig.canvas.draw()

            li3.set_data(ps,pp)
            plt.subplot(313)
#           plt.gca().relim()
#           plt.gca().autoscale_view()
            fig.canvas.draw()


except KeyboardInterrupt:
  name=input('file nam and file typee=')
  plt.savefig(name)
  ser.close()
```

停止程式, CNTL-C, 輸入存圖的檔名,
要以" " 包起.

附錄七之二, event.py

~/quarknet/monitor

```
from serial import Serial

import sys
import time
import datetime
import numpy as np

def rise_fall(ser):
  cond=True
  while cond:
    time.sleep(0.01)
    q=ser.readline(ser.inWaiting()).strip("\r\n")
    if len(q)==72:
      p=q.split(' ')
#      print 'check', q
      q1=p[0]
      q2=p[1]
      q3=p[2]
      q4=p[3]
      q5=p[4]
      q6=p[5]
      q7=p[6]
      q8=p[7]
      q9=p[8]
      qa=p[9]
```

由DAQ卡讀進一行數據, 分成16筆資料,

```
        qb=p[10]
        qc=p[11]
        qd=p[12]
        qe=p[13]
        qf=p[14]
        qg=p[15]

        p1=int(q1,16)
        p2='{0:08b}'.format(int(q2,16))
        r=str(p2)[0]
#       print q2,p2,r
        if r=='1':
          cond=False
    delt=1.25
    Delt=40.0
    tin=p1
    t1=0.0
    cond=True
    RE=np.zeros(4)
    FE=np.zeros(4)
    while cond:
        p1=int(q1,16)
        p2='{0:08b}'.format(int(q2,16))
        p3='{0:08b}'.format(int(q3,16))
        p4='{0:08b}'.format(int(q4,16))
        p5='{0:08b}'.format(int(q5,16))
        p6='{0:08b}'.format(int(q6,16))
        p7='{0:08b}'.format(int(q7,16))
        p8='{0:08b}'.format(int(q8,16))
        p9='{0:08b}'.format(int(q9,16))
        t1 +=Delt*(p1-tin)
        index=0
        print  q
        for b in (p2,p4,p6,p8):
          if b[2]=='1':
            RE[index] += t1+delt*int(b[3:8],2)
          index +=1
        index=0
        for b in (p3,p5,p7,p9):
          if b[2]=='1':
            FE[index] += t1+delt*int(b[3:8],2)
          index += 1
        time.sleep(0.01)

        q=ser.readline(ser.inWaiting()).strip("\r\n")

        p=q.split(' ')
#       print 'check2 ',p
        if len(q)==72:
          q1=p[0]
```

如果標示是一個事件的開始, 就跳出去, 繼續處理這個事件

1tick=40ns, 1click=40ns/32=1.25ns
有兩個時間間隔: delt, Delt.
是將每Delt時間內分隔成32個delt時間間隔.

16進位的數換成8位元的2進位數
p2,p3是ch0的RE,FE
p4,p5是ch1的RE,FE
p6,p7是ch2的RE,FE
p8,p9是ch3的RE,FE

這個事件裡, t1是每行數據與前一行數據的觸發時間的間

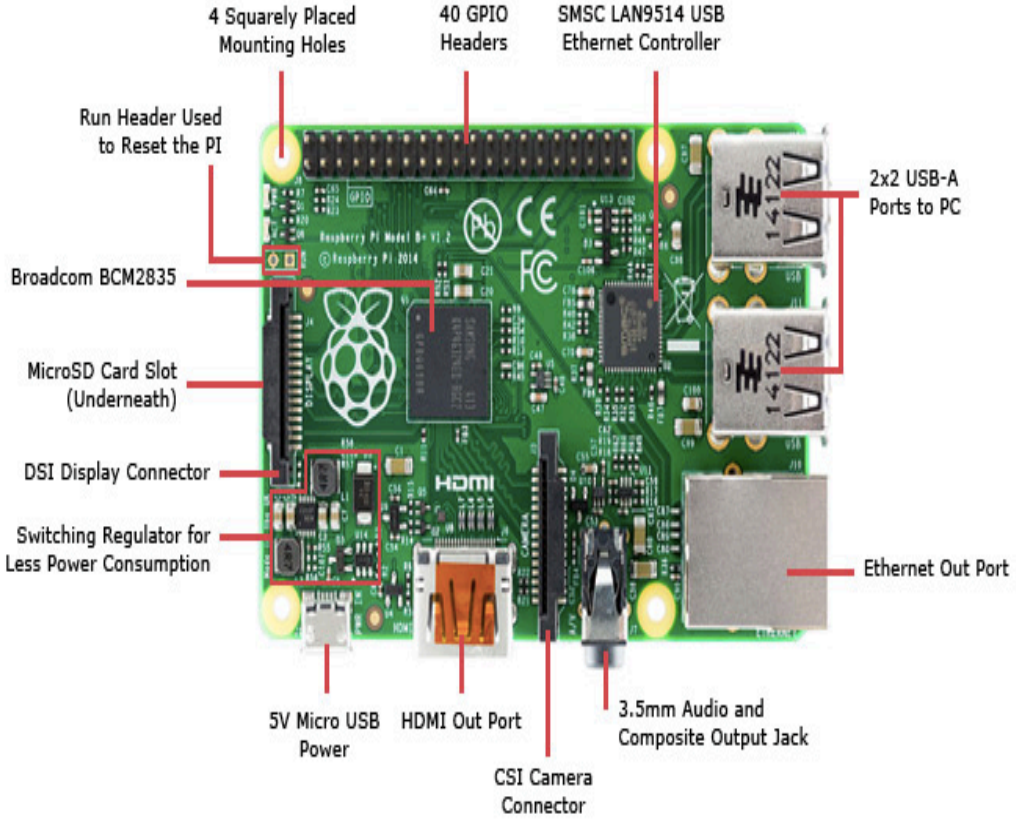
記錄每個計數器的RE與FE.
注意: 時間是要累積的

```
        q2=p[1]
        q3=p[2]
        q4=p[3]
        q5=p[4]
        q6=p[5]
        q7=p[6]
        q8=p[7]
        q9=p[8]
        qa=p[9]
        qb=p[10]
        qc=p[11]
        qd=p[12]
        qe=p[13]
        qf=p[14]
        qg=p[15]
      else:
        ser.flushInput()
        cond=False

      if str('{0:08b}'.format(int(q2,16)))[0]=='1': cond=False
      tin=p1
print FE[0]-RE[0],' ',FE[1]-RE[1],' ',FE[2]-RE[2],' ',FE[3]-RE[3]
return RE,FE
```

繼續讀入數據, 如果數據長度不到72, 就結束這個事件

如果數據顯示是新的事件的開始, 結束這個事件
注意: 下次再進這個程式, 是否就損失一個事件, 這部分要再考慮
???

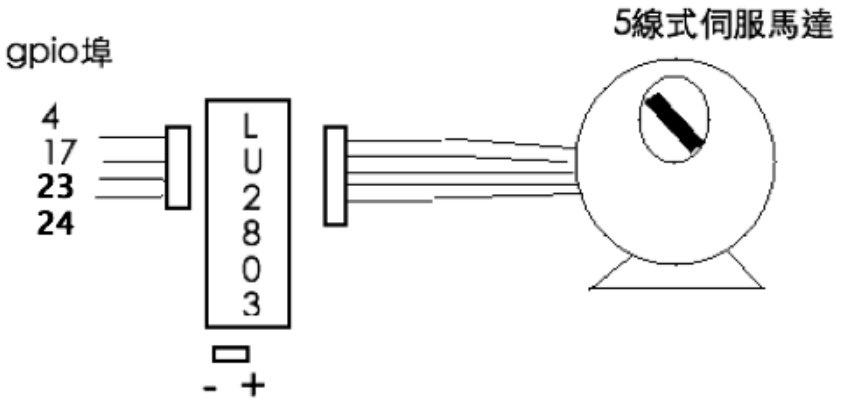
GPIO Pinout Diagram



圖一, 樹梅派電腦結構圖. GPIO埠位置編號與BCM編號圖.

wiringPi Pin	BCM GPIO	Name	Header	Name	BCM GPIO	wiringPi Pin
–	–	3.3v	1 | 2	5v	–	–
8	R1:0/R2:2	SDA0	3 | 4	DNC	–	–
9	R1:1/R2:3	SCL0	5 | 6	0v	–	–
7	4	GPIO7	7 | 8	TxD	14	15
–	–	DNC	9 | 10	RxD	15	16
0	17	GPIO0	11 | 12	GPIO1	18	1
2	R1:21/R2:27	GPIO2	13 | 14	DNC	–	–
3	22	GPIO3	15 | 16	GPIO4	23	4
–	–	DNC	17 | 18	GPIO5	24	5
12	10	MOSI	19 | 20	DNC	–	–
13	9	MISO	21 | 22	GPIO6	25	6
14	11	SCLK	23 | 24	CE0	8	10
–	–	DNC	25 | 26	CE1	7	11

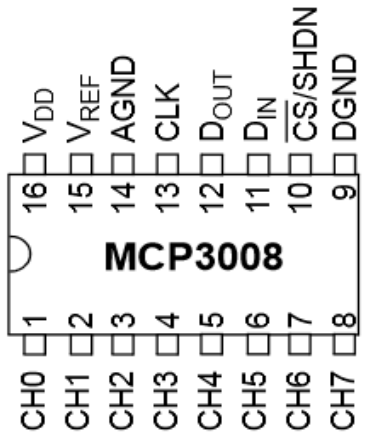
圖二, GPIO放大圖, 包括位置編號, BCM編號 與wiring pi編號



圖三, 步進馬達接線圖, 對應到馬達驅動器上編號 IN1, IN2, IN3, IN4的接線是G4(7), G17(11), G23(16), G24(18). 這裡的編號方式是 “BCM編號(位置編號)”.

The following list shows how the MCP3008 can be connected. It requires 4 GPIO pins on the Pi P1 Header.

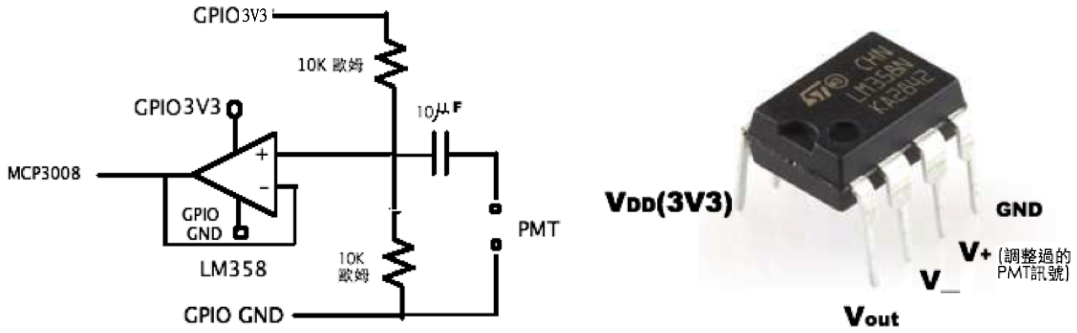
```
VDD   3.3V
VREF  3.3V
AGND  GROUND
CLK   GPIO11 (P1-23)
DOUT  GPIO9  (P1-21)
DIN   GPIO10 (P1-19)
CS    GPIO8  (P1-24)
DGND  GROUND
```



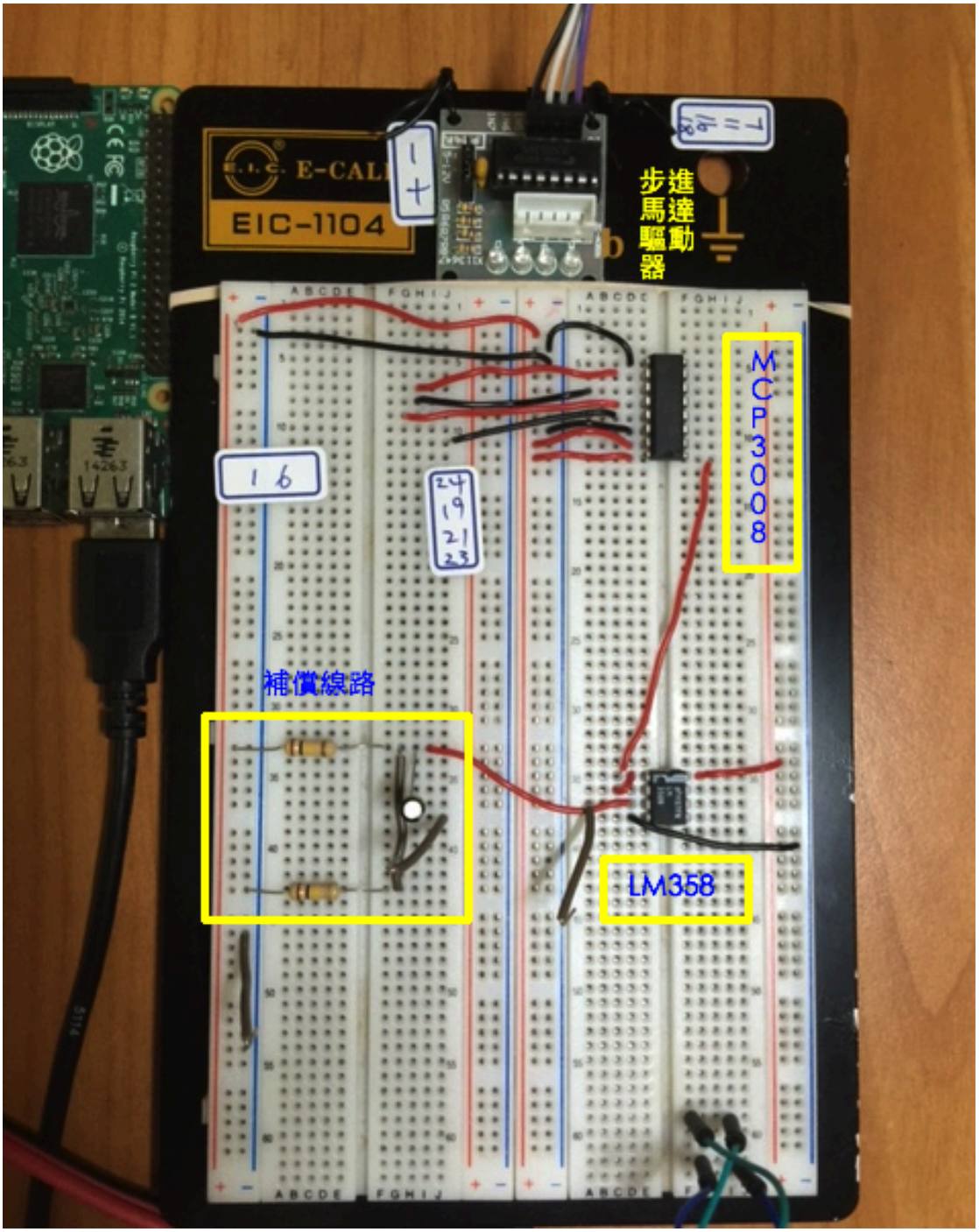
The CH0-CH7 pins are the 8 analogue inputs.

圖四, ADC(MCP3008)接線圖.對應到MCP3008右側編號 V_{DD}, V_{REF}, AGND, CLK, D_{OUT}, D_{IN}, CS, GND 的接線是 3V3(1) , 3V3(1), GND(7), G11(23), G9(21), G10(19), G8(24), GND(7)

圖五, 量電源盒訊號, LM358接線圖



圖六, 量PMT訊號, LM358與補償線路的接線圖



圖七，麵包板上的佈線圖

LINUX常用指令

cat
功能:印出檔案內容至銀幕(標準輸出)或合併多檔。
cat file1 file2 > file3可將檔案file1、file2之內容依順序合併, 並將結果存至file3。
語法:cat file

cd
功能:改變目前工作目錄之位置至directory, 若[directory]略之, 則返回至使用者之簽入時之目錄(Home Directory)。
語法:cd [目的目錄名稱]

clear
功能:清除目前銀幕顯示。
語法:clear
補充說明:當銀幕因某些原因無法正常顯示時, 可用指令"reset"重設銀幕。

cp
功能:檔案複製。若target為檔案名稱, 則file只能指定一個檔案, 若target為一個目錄名稱, 則會將file1、file2..分別複製至此一目錄之下。
語法:cp [選項] file1 file2
例子:
1. 將 file1 複製到 file2
   cp file1 file2
2. 將 file1 file2 fuile3 一起複製到後面的目錄裡
   cp file1 file2 file3 directory
3. 將目錄 directory1 複製到 目錄 directory2
   cp –R directoty1 directory2

df
功能:顯示磁碟相關資訊。
語法:df [選項]
Example:

```
pi@ASRP0114 ~ $ df
Filesystem     1K-blocks    Used Available Use% Mounted on
rootfs          14176312 4627564   8805580  35% /
/dev/root       14176312 4627564   8805580  35% /
devtmpfs          219744       0    219744   0% /dev
tmpfs              44784     280     44504   1% /run
tmpfs              5120       0      5120   0% /run/lock
pi@ASRP0114 ~ $
```

exit
功能:退出系統(=logout)。

grep
功能:找尋並印出files中含有字串pattern 所在行內容
語法:grep pattern filename
Example1: 找出/etc/passwd 中含有woody字串的行
[woody@dragon1 woody]$ grep woody /etc/passwd
woody:x:3694:710:Woody WU:/users/staff/woody:/bin/bash
woody1:x:5361:5361::/home/woody1:/bin/tcsh
woody2:x:5362:5362::/home/woody2:/bin/tcsh

Example2: 找出目前系統中, 所有以woody名義執行的process訊息
[woody@dragon1 woody]$ ps aux | grep woody
root 10554 0.0 0.8 2236 1136 pts/0 S 00:14 0:00 login -- woody
woody 10555 0.0 0.7 1700 944 pts/0 S 00:15 0:00 -bash
woody 10569 0.0 0.6 2516 876 pts/0 R 00:17 0:00 ps aux
woody 10570 0.0 0.4 2124 520 pts/0 S 00:17 0:00 grep woody

gunzip (gnu unzip)
功能:解開壓縮檔(.gz)
語法:gunzip file_name
Example1: 解開檔名為a.gz的壓縮檔, 並刪除a.gz
[woody@dragon1 woody]$ gunzip a.gz

Example2: 解開檔名為a.gz的壓縮檔, 保留a.gz, 並將結果輸出至"b"
[woody@dragon1 woody]$ gunzip a.gz > b

gzip
功能:檔案壓縮
語法:gzip file_name
如果用 gzip -d 效果就像 gunzip

last
功能:列出目前與過去登入系統的使用者相關資訊
語法:last [帳號名稱]
Example1:
[woody@dragon1 woody]$ last woody
woody pts/0 woody Thu Mar 9 00:15 still logged in
woody pts/0 woody Wed Mar 8 03:48 - 05:29 (01:41)
woody pts/0 liu Mon Mar 6 21:38 - 23:28 (01:50)

woody pts/1 woody Fri Mar 3 21:44 - 21:49 (00:05)
woody pts/0 woody Fri Mar 3 21:39 - 22:54 (01:14)
woody pts/0 woody Fri Mar 3 05:15 - 05:22 (00:06)

ls
功能:列出所在或指定目錄的內容or檔案之相關特性
語法:ls [-選項] [目錄or檔案名稱]

常用選項及其代表之意義
-a : 列出所有檔案名稱, 包括以『.』開頭之隱藏檔, 如.profile、.login等。
-d : 若參數的為一目錄名稱, 則只列出檔案名稱而非目錄之內容。
-l : 列出檔案的詳細資料, 包括檔案形態、存取權限、連結數目、擁有 者名稱、群組名稱等。

man
功能:線上求助程式
語法:man <指令名稱>
Example:
$man ls

mkdir
功能:建立目錄
語法:mkdir <目錄名稱>

more
功能:以每次一頁方式顯示一個檔案
語法:more file_name

passwd
功能:更改使用者密碼
語法:passwd [username]
補充說明:
1.當username不提供時, 為更改目前login者密碼
2.更改本身密碼時, 必須先輸入舊密碼, 確認無誤後, 才可更改。
3.不具備MD5功能的系統, 密碼最大長度為8字元

w (who)
功能:列出目前線上使用者資訊
語法:w (who)

2018
Organized by: Macao Science Center,
Inst. Phys., Academia Sinica, and TW-QuarkNet

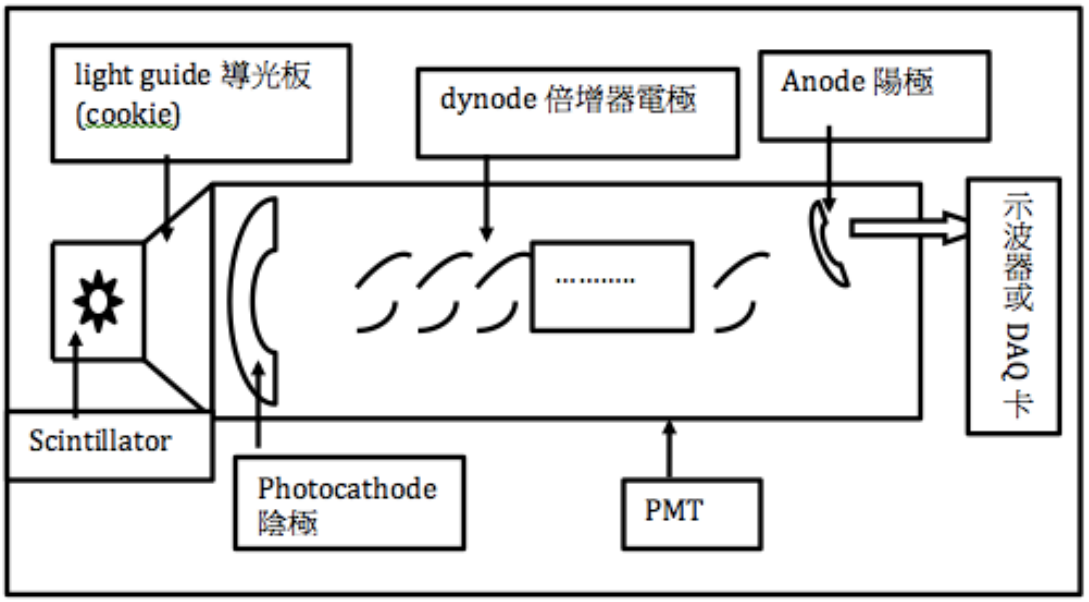
12/1	Saturday	12/2	Sunday
9:30-10:00	Registration	9:30 | 10:00	Registration
10:00 | 11:00	**工作坊演講講** Introduction of particle detectors in high-energy physics experiments (高能物理實驗中粒子探測器簡介) **章文箴**	10:00 | 11:00	科普講座 Exploration of the subatomic world (次原子世界的探索) 章文箴
11:10 | 11:40	The ClassRoom Cosmic Ray Detector 蕭先雄	11:10 | 12:30	Measuring Cosmic Rays and Discussion Session 蕭先雄
11:40 | 12:30	The Cosmic Ray e-Lab 蕭先雄		
12:30 | 1:30	Lunch		
1:30 | 2:10	Breakout Session Of Counter Assembly 蔡佩容		
2:10 | 4:10	Breakout Session Of Counter Assembly 蔡佩容		
4:10 | 4:30	Break		
4:30 | 5:30	Detector Assembly, Detector Testing and Discussion Session 蔡佩容, 蕭先雄		

地點**:**
自帶物品**:** 筆電**.**

Quarknet閃爍體計數器(Scintillator Counter)
檢測, 組裝與量測

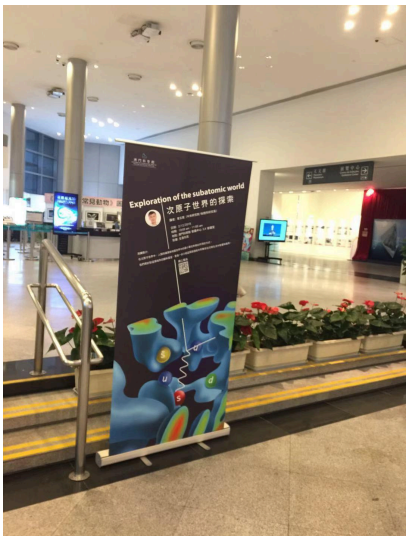
閃爍體計數器工作原理

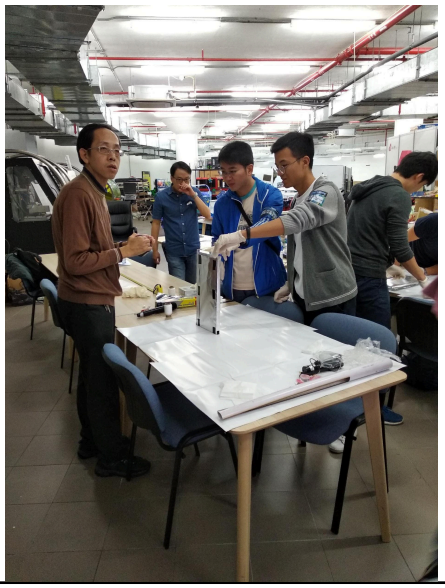
閃爍體計數器包含光電倍增管(PMT)閃爍板(scintillator)兩部分. 當宇宙線撞擊到閃爍板, 使閃爍板發光, 這個光會在閃爍板中散射, 最後經由導光板(light guide)進入光電倍增管中，在陰極(photocathode)經由塗料的光電效應性質將光轉變成電流訊號，再透過倍增器電極(dynode)將訊號放大百萬倍以上，在陽極(anode)處收集，利用示波器或DAQ卡讀取、分析或記錄。

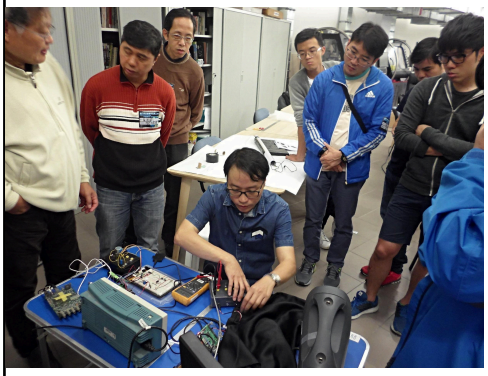


Quarknet用的PMT是Sens-Tech的設計(型號P30CW5), tube base裡加上小型HV power supply ,
http://www.sens-tech.com/index.php/products/photomultiplier-modules

目錄：







澳門科學館
CENTRO DE CIÊNCIA DE MACAU
MACAO SCIENCE CENTER

修業證書
Certificate of completion

茲證明
This is to certify

張子健

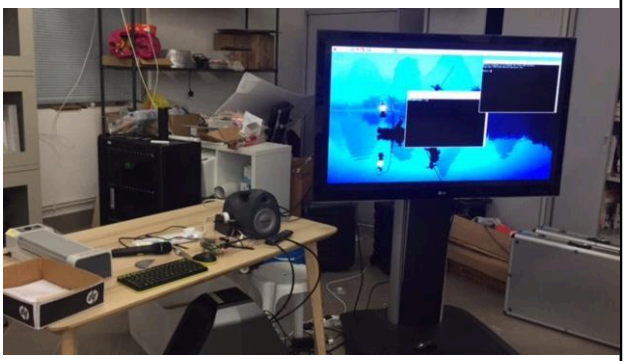
完成由澳門科學館舉辦，中央研究院物理研究所以及台灣夸克網協辦
為期十二小時之澳門夸克網工作坊課程

has completed a 12-hour Macao-QuarkNet Workshop 2018 Course organized by
Macao Science Center, Inst. Phys., Academia Sinica, and TW-QuarkNet.

由 From : 1/12/2018 至 To : 2/12/2018

梁思聰先生
教育及展品總監
Mr. LEONG, Si Chong Oscar
Education and Exhibits Controller

簽發日期
Date of Issue :



(一)檢測光電倍增管(PMT)

檢測流程:

一、製作LED光源套筒以測試PMT之輸出電壓

二、測試PMT之電壓值變化

附錄 1,2,3.

一、製作LED光源套筒以測試PMT之輸出電壓

材料:

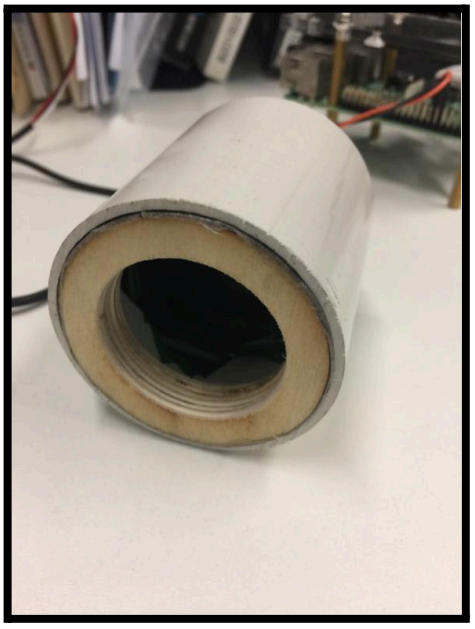
1. PVC水管(直徑56mm x 高度60mm) x 1
2.圓型木板 x 2
3.電路測試板
4.O型木板 x 1
5. LED燈泡 x 1
6.玻璃紙 7片 (50mm x 50mm)

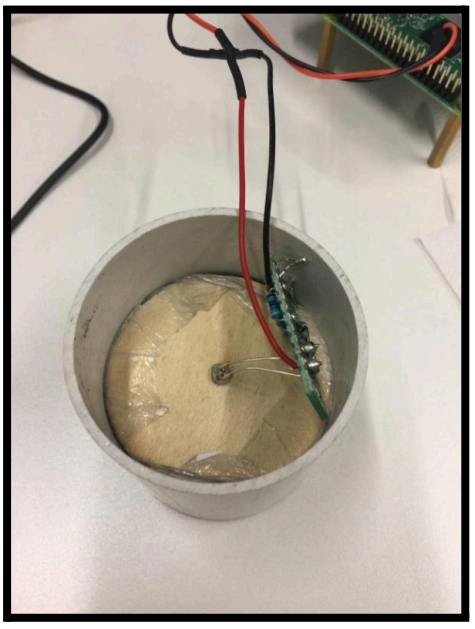
製作目的：

製作LED光源套筒以模擬宇宙射線在閃爍體中所激發的微弱光訊號。

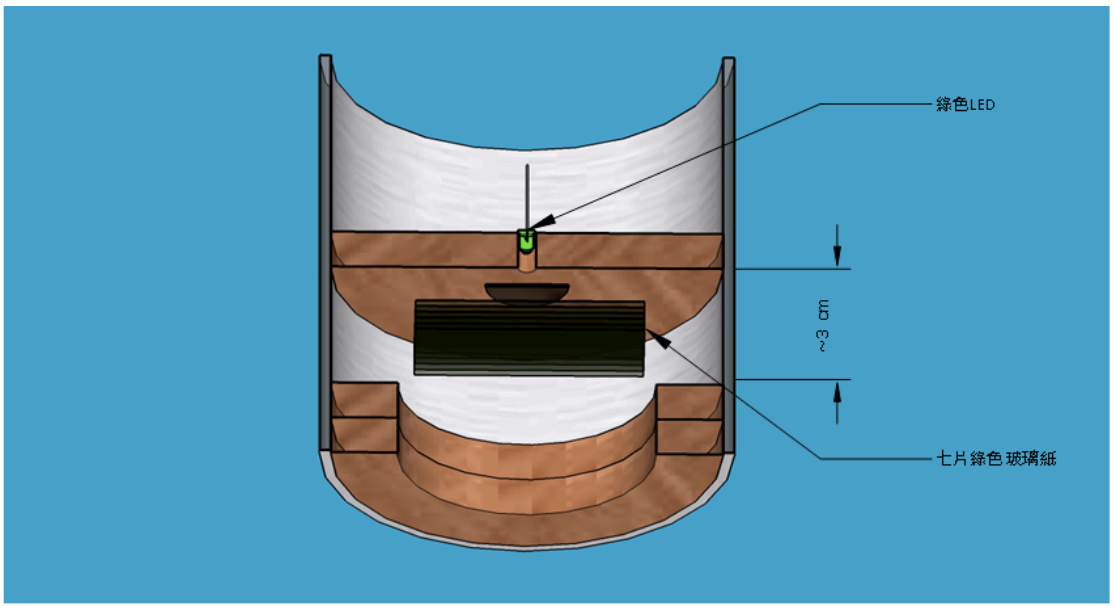
製作過程:

可以利用Laser Cutter或線鋸床, 把5mm厚的白楊木片切割出如下圖所示的形狀, 共兩片O型木片及一片圓形木片, O型木片之外徑與PVC水管內徑一致, O型片內徑跟PMT外徑一致(可預留0.5mm之寬度, 方便套進PMT), 可用雙面膠帶使之與PVC管黏合(或可用AB膠黏合)。圓型木片可在中央位置鑽一2.5mm孔徑之小孔以便放入LED燈泡, 孔徑大小視乎LED燈泡的實際大小為準, 燈泡可用雙面膠帶與圓形木片黏合。

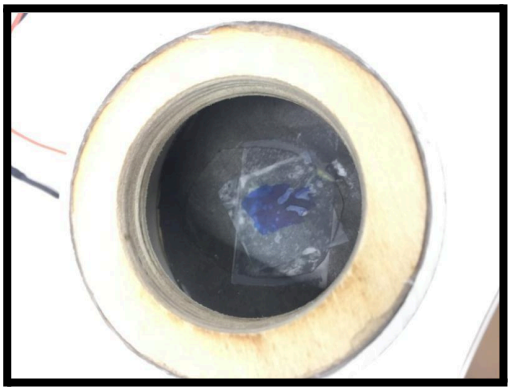




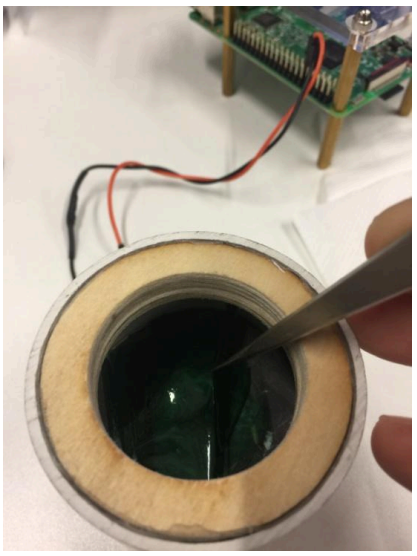
O型木片與圓形木片間距約3cm, 詳見下圖說明:



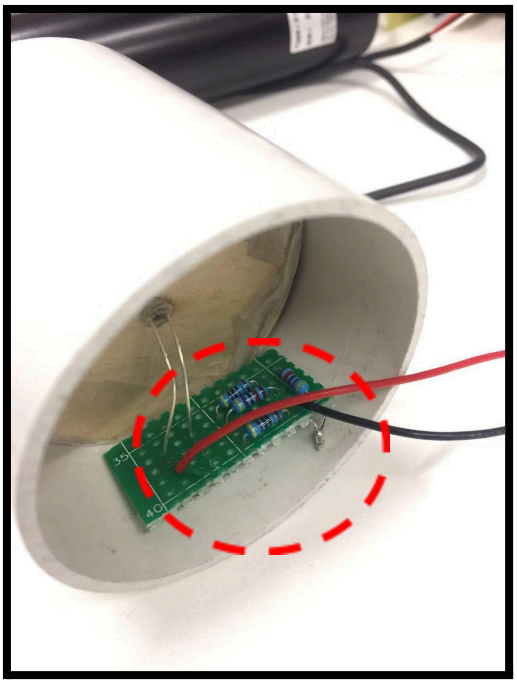
可在LED小孔上預先貼上薄圓型卡紙並用針扎一小孔，小孔越小越好，再用3M半透明膠帶貼在上方，並用藍色簽字筆在小孔上塗上顏色。



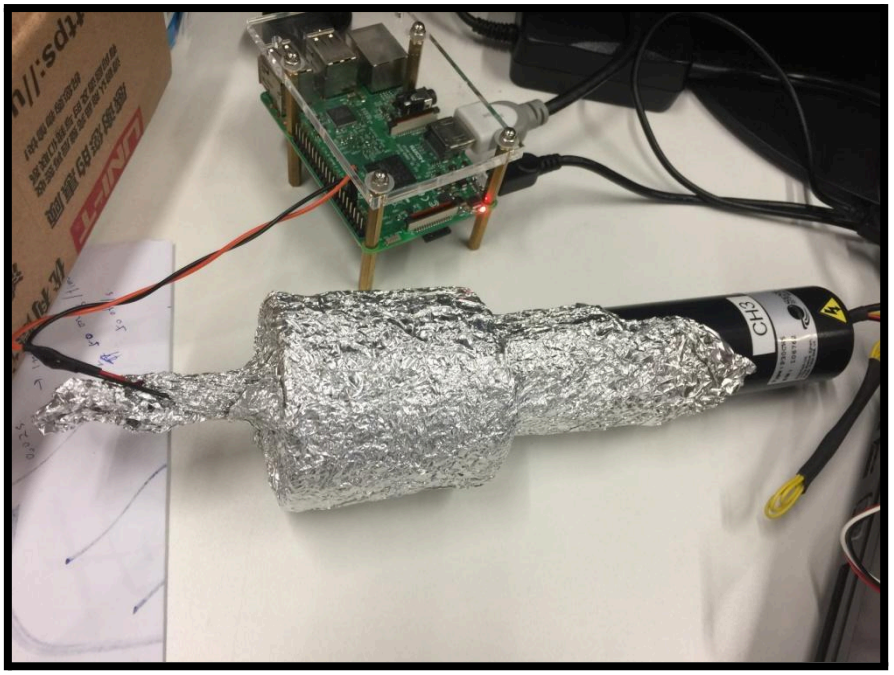
預先準備好七片綠色玻璃紙，裁成正方型其大小以僅可放入筒內便可，再用鑷子一片一片疊在LED小孔上，目的是減弱*LED*的光線，完成後可嘗試用肉眼測試光源的亮度，基本上，當LED的光源調至最亮時(Brightness = 100)，眼睛即使探進套筒也很難區別有光源發出，即便是如此漆黑，光電倍增管仍能偵測出微弱光線！



本測試所使用之LED之總電阻值為55k歐姆其目的使LED亮度降至更低:



光電倍增管(PMT)插入LED光源套筒後再用鋁箔紙包裹, 儘量讓鋁箔紙緊貼套筒、電源線以及PMT表面，以防止光線進入PMT，PMT可先往內推進LED小孔表面，再用鋁箔紙固定住以減少外界光線進入，**由於PMT可將微弱電流放大數百萬倍，若有漏光情況可能會損害PMT，故製作時要多加注意**。



二、測試PMT之電壓值變化

材料:

1.電源分配盒(Power distribution box: PDU) x 1 2.示波器(Oscilloscope)x1 3.萬用電表(Multimeter)x1 4.樹莓派Raspberry Pi 3 Model B x 1 5.HDMI 顯示器,	6.無線滑鼠x1、無線鍵盤x2 7.USB電源線 8.BNC連接線(PMT 接 視波器)x1 9.PMT電源線(單聲道音源線) x 1 10.LED光源套筒 x1 11.鋁薄紙 x1卷

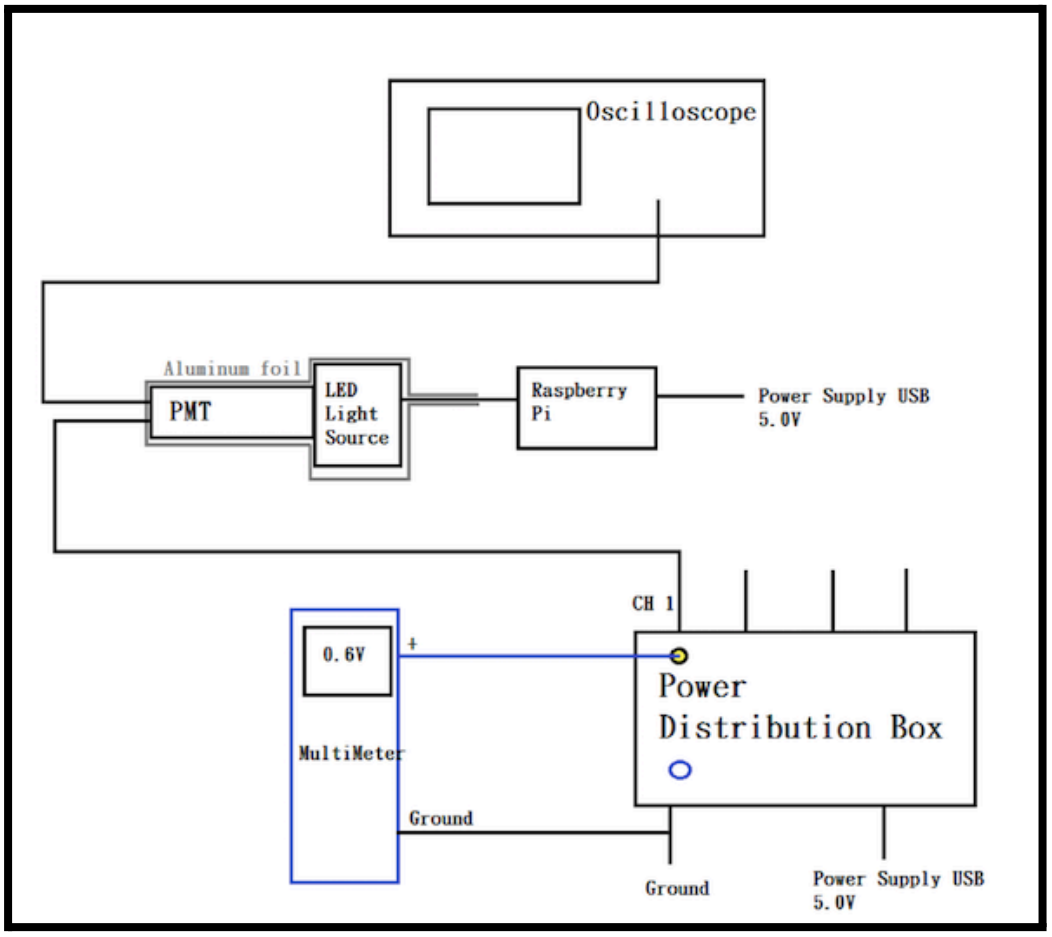
測試目的:

利用LED所產生的點光源並改變電源分配盒上的電阻值, 以觀察並記錄PMT信號的變化, 了解PMT的工作電壓值範圍。

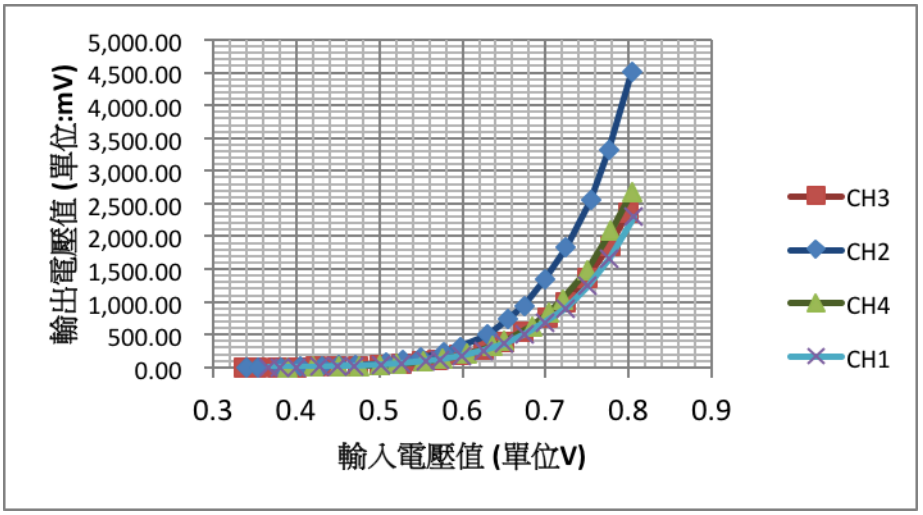
測試流程:

開啟Raspberry Pi 3, USB電源線接電後Raspberry Pi3會自動打開, 待開啟完成後, 在顯示器上會看見Raspberry Pi的主畫面, 選取terminal 終端機圖示並打開, 可執行已製作好的python程式:pwm.py, 該程式能讓GPIB端口送出一方波信號, 本實驗使用了GPIO的第6和11針腳與LED連接, 執行pwm.py程式並輸入亮度值(brightness=?), LED便會發亮。

把PMT放進LED套筒內並用鋁薄紙包裹好, 切勿讓外界光線進入PMT套筒內部, PMT的輸出信號線可以跟示波器連接上, PMT之電源是來自電源分配盒所提供, 關於電源分配盒之製作(詳見附件說明)接線圖可參考以下圖示:

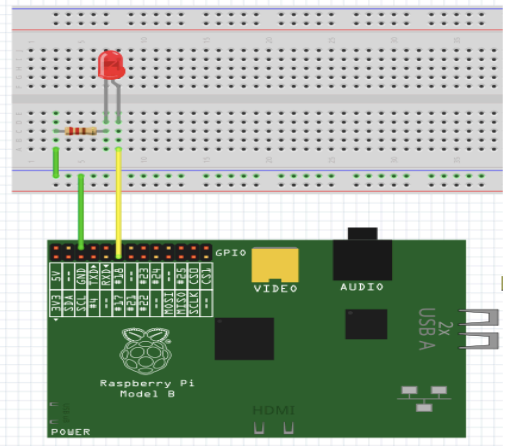


PMT之輸入電壓值由可變電阻獲得，先讓電阻調至最大，然後慢慢調至0.6V再開始測量，勿讓輸入電壓值調至太大以免傷害PMT（*PMT之輸出電壓值可能會增至十幾伏之高，故操作時務必注意！*），觀察萬用電表之電壓值，每0.025V記錄一次示波器上波形之電壓值。LED亮度可設定較微弱光源（Brightness < 30），可獲得如下圖測量四支不同PMT之輸出電壓值變化：

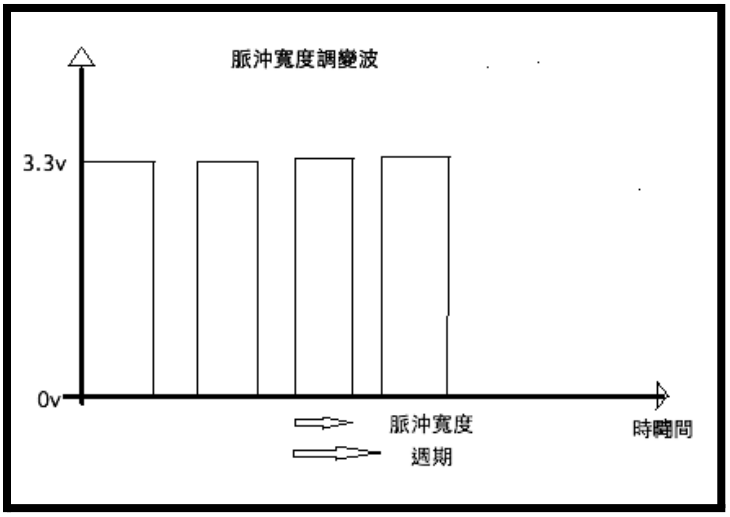


附錄:

1. **raspberry pi** 接LED與電阻的線路, 此例是以第6隻針腳(GND)接電阻, 第11隻針腳(GPIO 17)接LED的正極端(長針腳).



2. PWM 波形, 脈沖寬度的百分比例由程式pwm.py裡的x值決定.



3. 程式pwm.py送出高度3.3V週期性的方波(100Hz), 程式裡的x值對應到每個週期裡3.3V方波所
占的百分比(參考上圖), 下指令sudo python pwm.py 就可以從GPIO17送出 這個可以調方波百分
比的波形(PWM), 點亮LED燈.

```
# pwm.py
import RPi.GPIO as GPIO
#from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setup(17,GPIO.OUT)
p1 = GPIO.PWM(17,100)
try:
        while True:
                x= float(raw_input("(0~100)brightness= ? "))
                p1.start(x)
except KeyboardInterrupt:
```

```
    p1.stop()
GPIO.cleanup()
```

(二)組裝閃爍體計數器(Scintillator Counter)

準備工作:

1. 將桌面鋪上乾淨紙張
2. 將閃爍體取出置於紙上

注意: handle閃爍體前, 請先戴上手套!!!

零件與材料：

閃爍體+光電倍增管+轉接環(cookie)+光學油+迴紋針PVC水管+鋁箔紙+刀片+ 剪刀 + 黑膠帶 +
一般透明膠帶 +水龍頭封鐵氟龍白膠布 + 100cm直尺 + 手套 + 酒精 + 拭鏡紙 .

組裝流程：

一、閃爍體包裝
二、轉接環(cookie)組裝
三、光電管(PMT)組裝
四、水管治具(jig)組裝
五、測試結果

一、閃爍體包裝

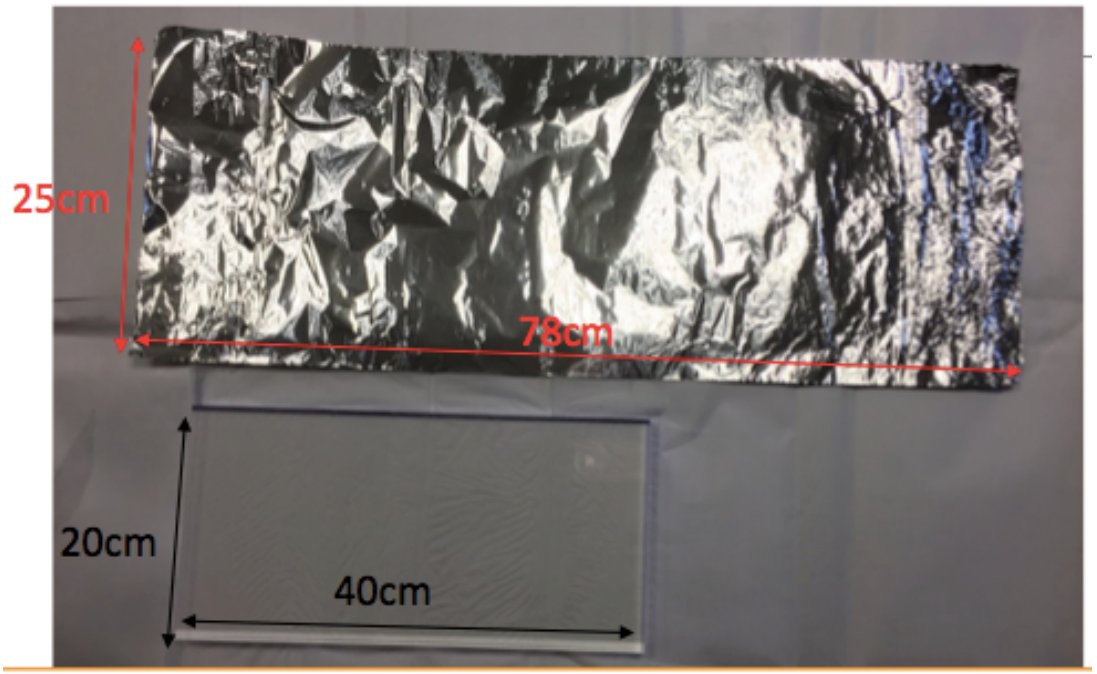
11. 戴上白手套將閃爍板取出, 先放置在乾淨的表面上, 測量閃爍板的常寬

高(40cmx20cmx2cm)，並記錄之。

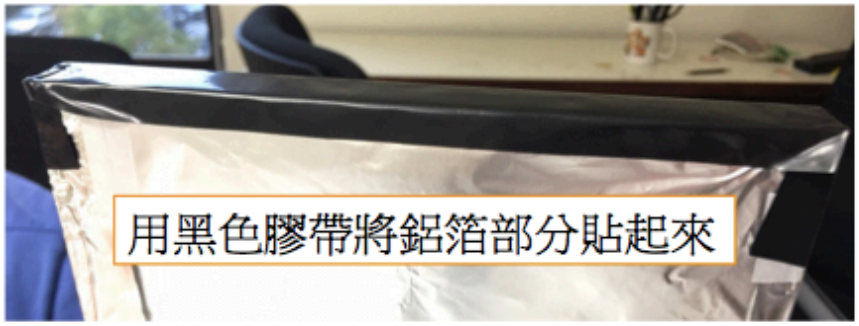
注意：測量時戴上白手套，以免手上油污沾在閃爍板上，影響正式測量時的光線反射。

12. 裁切鋁箔紙(25cm x 78cm)，亮面向上，將閃爍板放置在鋁箔紙離邊緣約兩公分(預留cookie空間)。
13. 先從閃爍板的長邊包起，然後再其他邊，鋁箔紙不需要特別平整，但是不能撕破及小洞，但是可以稍微修補。角落包裝時像禮物方式包裝，打開的折邊先用透明膠帶貼好固定.

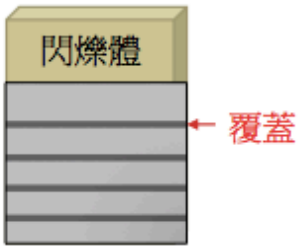
4. 用黑色膠帶將鋁箔部分貼起來，貼的時候，將所有縫隙貼死且每塊膠帶需要覆蓋(避免漏光).



打開的折邊用透明膠帶貼好固定

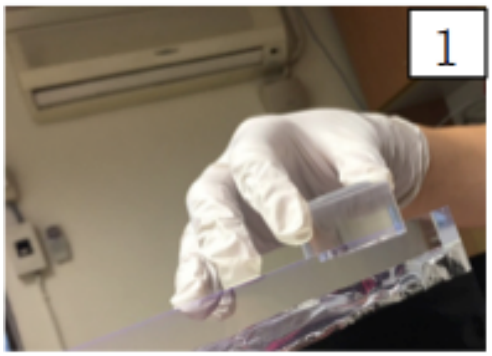


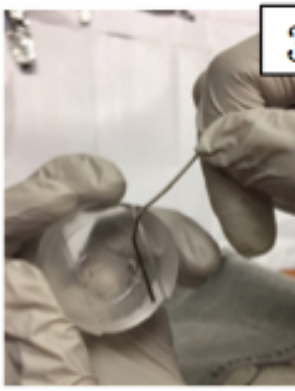
Tips:
貼的時候，將所有縫隙貼死且每塊膠帶需要覆蓋(避免漏光)

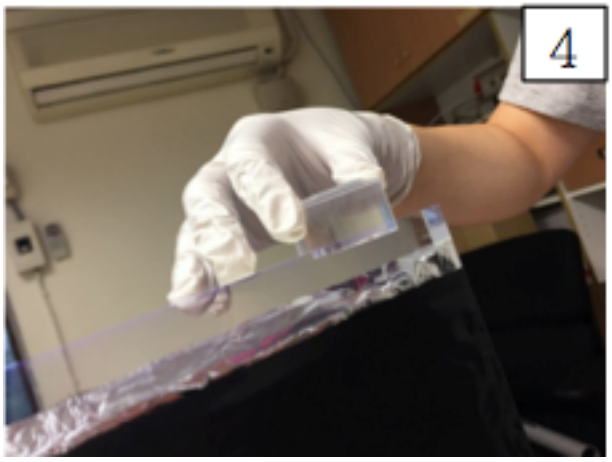


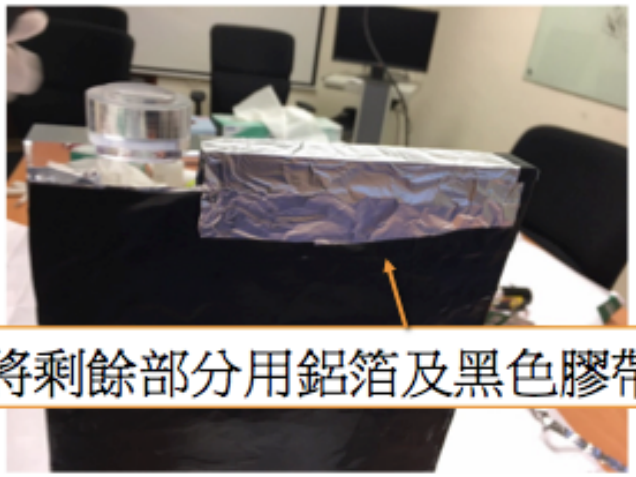
二、轉接環(cookie)組裝

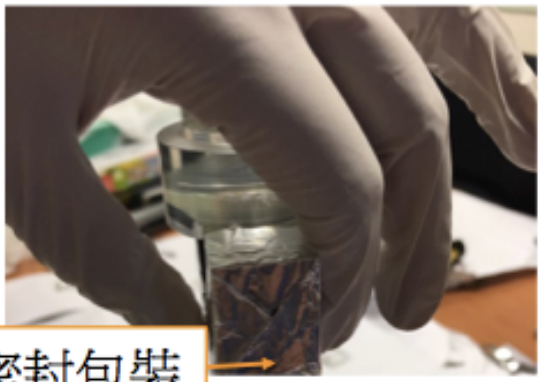
1. 先將cookie放上閃爍體(觀察透光性).
2. 使用迴紋針沾少量光學油(約一米粒量).
3. Cookie塗上光學油.
4. 將cookie輕輕放上閃爍體, 放置位置離邊約**3**公分, 稍微輕壓並左右移動, 讓cookie和閃爍體緊密結合, 並將光學油中的氣泡排出。
5. 再次觀察透光性。
6. 將剩餘部分用鋁箔及黑色膠帶密封包裝.









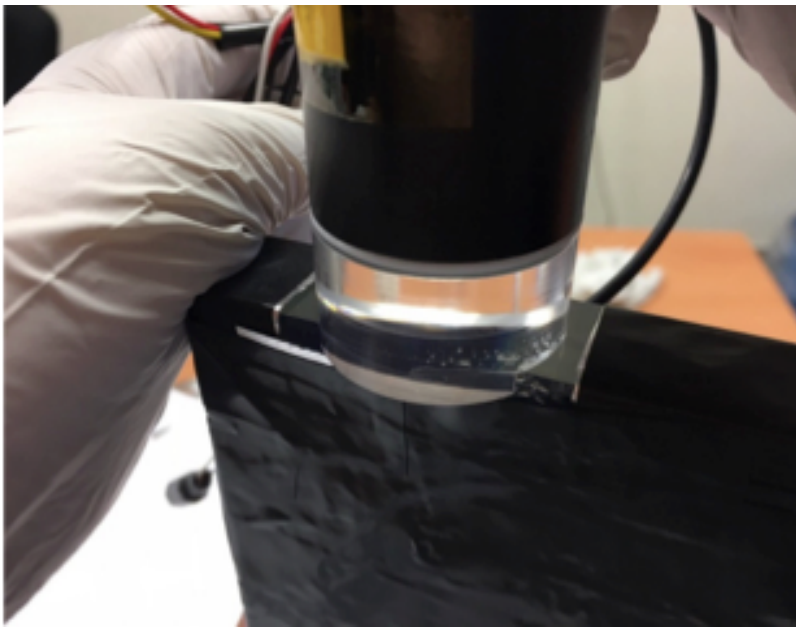


再將剩餘部分用鋁箔及黑色膠帶密封包裝

三、光電管(PMT)組裝

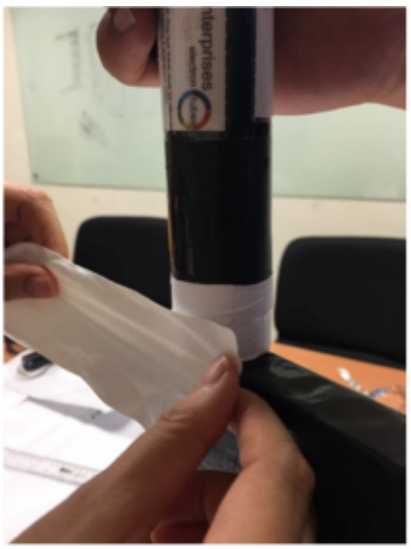
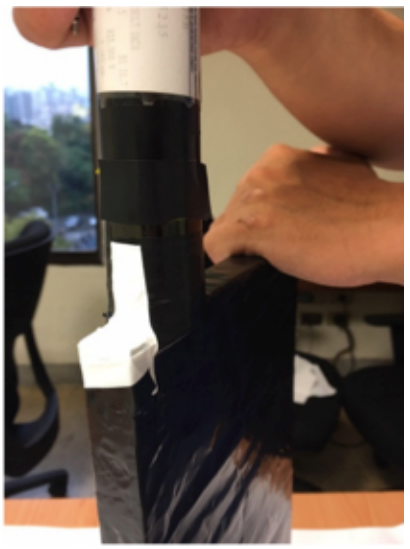
1. 將閃爍體站立, PMT（光電管）前端塗上光學油. 將PMT輕輕地和cookie對正結合，並稍

微輕壓及轉動，讓PMT和閃爍體緊密結合，並將光學油中的氣泡排出.



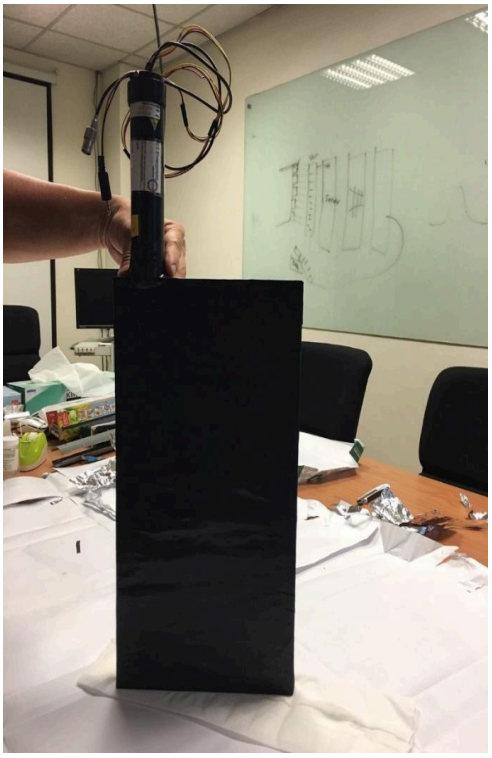
2. 白色鐵氟龍膠帶繞cookie 2~3圈，側邊下邊都要包到. 用黑膠帶將PMT和cookie接合處

黏貼，並再檢查位置沒有變歪，不會漏光.

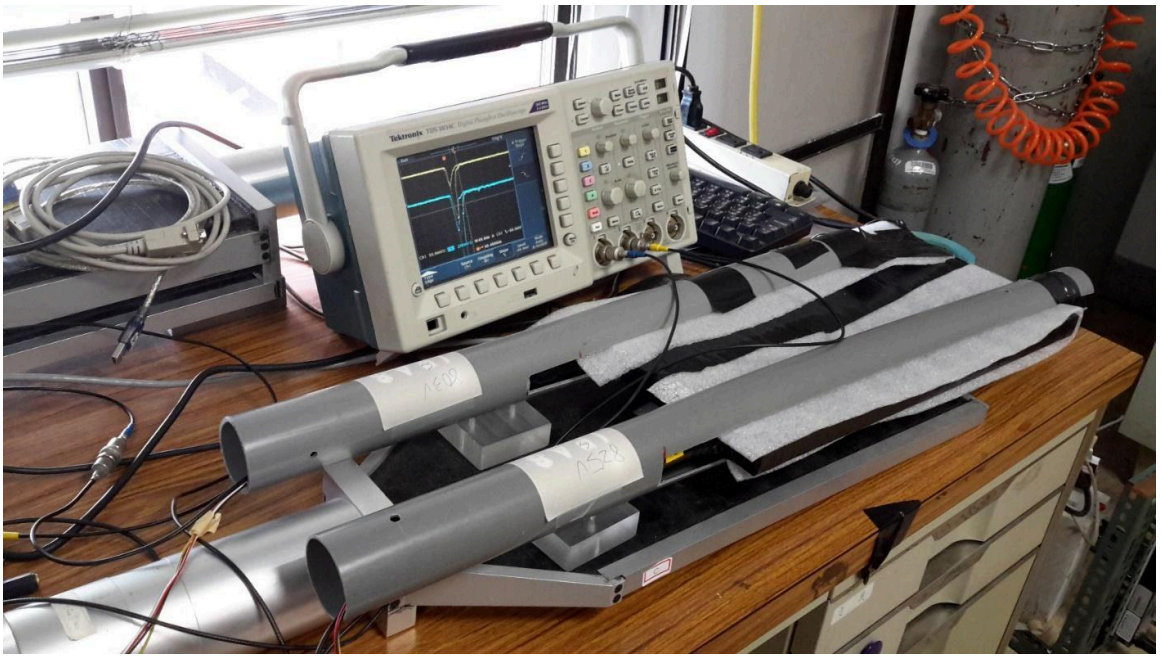
四、水管治具(jig)組裝

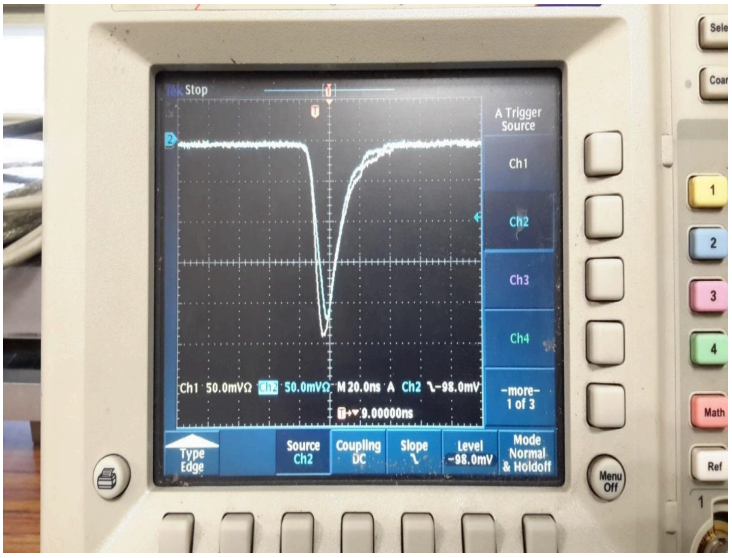
水管的開口沿PMT方向插入，中間可使用泡棉當緩衝，將閃爍板夾住後使用膠帶固定水管。（注意不能卡住PMT的接線）




組裝完成

五、測試結果




Stop
A Trigger Source
Ch1
Ch2
Ch3
Ch4
Ch1 50.0mVΩ BW 50.0mVΩ M 20.0ns A Ch2 -98.0mV
9.00000ns
-more-
1 of 3
Type Edge
Source Ch2
Coupling DC
Slope
Level -98.0mV
Mode Normal & Holdoff
Menu Off
Math
Ref

QuarkNet Research Workshop 2020, Macau Science Center

Date: 14/2/2020 :
Time: 15:00 – 21:30
課程**:**
15:00-16:00教師培訓:夸克網e-Lab教學1:
樹莓派與數據擷取卡的溝通, 量測計數器宇宙射線通量的單符合與雙符合,求取平坦區(plateau), 得到計數器的工作電壓與閥值.
16:00-16:30 茶點小休
16:30-17:30教師培訓:夸克網e-Lab 教學2:
存取數據並上傳到e-Lab, 使用e-Lab的分析工具
17:30-18:00 教師培訓: 樹莓派操作QuarkNet實驗的軟硬體教學 1:
介紹類比轉數位的晶片MCP3008, 運算放大器晶片LM358 與步進馬達uln2803.
18:00-19:00教師培訓:樹莓派操作QuarkNet實驗的軟硬體教學2:
介紹執行夸克網eLab教學1與2 各類實驗的Python 的程式
19:00-21:30 工作晚膳。

Schedule:
15:00-16:00 Cosmic Ray e-Lab, part I:
communicate DAQ card with Raspberry Pi, measure singles flux rates and double coincidence flux rates for plateau, find the working voltages and threshold levels.
16:00-16:30 Break
16:30-17:30 Cosmic Ray e-Lab, part II:
save and upload data to e-Lab and use the analyzing tools of cosmic ray e-Lab
17:30-18:00 Raspberry pi and python programs, part I:
introduce hardwares including a ADC chip MCP3008, an Op Amp chip LM358 and a stepper motor uln2803.
18:00-19:00 Raspberry pi and python programs, part II:
various python programs to perform above experiments in Cosmic Ray e-Lab part I and part II.
19:00-21:30 Dinner。