Università degli Studi di Ferrara
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Realizzazione di Librerie per il Controllo di un Sistema ad Alta Tensione per Rivelatori di Muoni

Primo Relatore:
Dott. MIRCO ANDREOTTI

Secondo Relatore:
Dott. GIANLUIGI CIBINETTO

Laureando
RUGGERO SINIGAGLIA

Anno Accademico 2005-2006

...alla mia famiglia ed ai miei amici più cari...
...e soprattutto a me stesso

Indice

Introduzione

In questa tesi presentiamo la realizzazione di librerie per l'utilizzo dell'alimentatore ad alta tensione mod. SY546 della CAEN, sviluppiamo un procedura per gestire il condizionamento dei tubi LST in modo automatizzato. Nell'ultima parte viene presentato una procedura automatica, per la misura dei plateau.

Per la realizzazione delle librerie, della procedura di condizionamento e della misura dei plateau è stato utilizzato il software della National Instrument, Labview 8. Tutto il lavoro di tesi è stato svolto nel laboratorio di criogenia ed LST del Dipartimento di Fisica dell'Università degli Studi di Ferrara. In questo laboratorio si svolgono attività di ricerca e sviluppo nel campo della rivelazione delle particelle, in particolare nella rivelazione dei muoni.

Le attività di ricerca sono rivolte allo sviluppo di sempre più efficienti sistemi di rivelazioni di tali particelle in vista di un futuro utilizzo negli esperimenti internazionali di fisica delle particelle di nuova generazione.

In generale quando si utilizzano rivelatori di particelle si devono trattare segnali elettrici molto veloci, quindi inevitabilmente si devono utilizzare sistemi di elettronica in grado di manipolare tali segnali con frequenze molto alte. In conseguenza di questo fatto l'acquisizione, l'elaborazione e la memorizzazione dei dati corrispondenti deve essere affidata ad un calcolatore in grado di gestire con sufficiente rapidità tutte queste operazioni. Il calcolatore dovrà essere interfacciato con il sistema di alimentazione attraverso oppurtuni sistemi di interfaccia e dovrà quindi gestire il settaggio della tensione di lavoro, corrente massima, e altri parametri principali, utilizzando un'applicazione software adeguatamente progettata.

Il progetto prevede la conversione delle librerie esistenti, che sono realizzate in linguaggio C, in modo da permetterne l'uso tramite Labview, il quale utilizza il linguaggio grafico, la realizzazione software di un automatismo per eseguire la procedura di conditioning dei tubi LST e la creazione di un metodo per la misura dei plateau.

La prima fase del lavoro si è concentrata sullo studio delle librerie esistenti al

fine di individuarne le modalità di funzionamento per poi eseguirne la conversione utilizzando il linguaggio di Labview. Al termine della realizzazione delle librerie, esse sono state utilizzate per la realizzazione della procedura di conditioning per i tubi LST e per la realizzazione della parte software che gestisce la costruzione dei diagrammi di plateau.

Le librerie e il software che gestisce le varie procedure sono stati sviluppati sul sistema operativo Linux Fedora Core 3.

Una considerazione del lavoro svolto può essere il fatto che anche un informatico può lavorare nell'industria, realizzando in questo caso procedure utilizzate per eseguire test e per controlli di qualità, che siano il più affidabili e automatizzati possibili durante la produzione di massa dei rivelatori.

Nel capitolo 1 riportiamo una descrizione dell'apparato ad alta tensione SY546 Descrivedone le caratteristiche principali. Viene poi riportato una descrizione generale delle librerie già presenti, scritte in C. Nel capitolo 2 viene introdotto la realizzazione tramite labview delle librerie descritte nel capitolo 1. Vengono ripresi i vari moduli, dandone una descrizione per ognuno di essi. Nel capitolo 3 è descritta la procedura che implementa la procedura di condizionamento dei tubi LST. Partendo da una descrizione generale della procedura, fino ad arrivare all'effettiva implementazione con Labview. Nel capitolo 4 è descritta la parte software che gestisce la creazione dei diagrammi di plateau. Nella prima parte vi è un introduzione teorica all'utilità dei diagrammi. Nella seconda parte viene descritta la parte software che gestisce la procedura.

Capitolo 1

Apparato Sperimentale

Nel seguente capitolo verranno descritti le principali caratteristiche del sistema di alimentazione ad alta tensione. Questo apparato viene utilizzato nel laboratorio di criogenia e LST del dipartimento di Fisica dell'Universitá degli Studi di Ferrara, per l'alimentazione dei rilevatori di muoni[1](LST), i quali sono descritti in appendice.

1.1 C.A.E.N. mod.SY546

L'apparato ad alta tensione provvede all'alimentazione dei rilevatori di muoni. Questo alimentatore può contenere fino ad otto elementi (board) ciascuno con 12 canali. Lo stato di ogni canale può essere monitorato. È presente una protezione del circuito che automaticamente limita la corrente e previene il danneggiamento dei tubi. Tutte le uscite di una singola *board* sono impostate allo stesso valore di tensione, mentre la corrente può assumere valori distinti e può essere monitorata separatamente per ogni canale. L'apparato deve provvedere a regolare la tensione fino a 6 kV, ed a proteggere da sovracorrenti. L'alimentatore è provvisto di un alloggiamento per un controller (mod. A547), il quale permette il controllo da remoto del sistema per mezzo di un terminale[2] connesso attraverso la porta RS232C oppure attraverso una scheda caenet che descriveremo in dettaglio.

Ci sono vari sistemi d'allarme/protezione, come overcurrent (ovc), undervoltage (udv). Tutti i parametri più rilevanti sono mantenuti in una memoria non

[1]I muoni sono particelle cariche della famiglia dei leptoni. I muoni sono molto simili agli elettroni, anch'essi leptoni, in particolare si distinguono da questi per la massa che è circa 200 volte più grande. I muoni studiati in questo laboratorio sono prodotti nell'alta atmosfera terrestre a seguito di interazioni nucleari dei raggi cosmici con i nuclei dei gas dell'atmosfera stessa.

[2]compatibile ANSI VT100

volatile in modo che le informazioni non si perdano con lo spegnimento della macchina. Il sistema può essere impostato in modo che all'accensione oppure ad un riavvio della macchina, vengano portati tutti i canali dal valore zero al valore settato, senza che vi sia bisogno dell'intervento di un operatore. Automaticamente viene ripristinato lo stato in cui si trovava prima che l'alimentazione fosse interrotta.

Per il nostro progetto l'alimentatore verrà utilizzato per eseguire procedure automatiche per il condizionamento dei tubi LST e per la misura dei plateau.

Controllo e Monitoraggio

Una interruttore situato sulla parte destra dell'apparato provvede ad avviare il sistema. Un interruttore HV EN posto sul pannello frontale gestisce l'abilitazione/disabilitazione delle uscite.

Caratteristiche generali:

- 8 board disponibili per ogni crate;
- Controllo da remoto: RS232 o con scheda Caenet;
- valori impostabili da remoto: Tensione - Corrente - Rump Up - Rump Down - Trip;
- valori monitorabili da remoto: Tensione - Corrente - Channels status - General Status;
- Protezione da password per ogni canale o gruppo;
- Vmax: 6 kV
- Imax: 1 nA
- Corrente: 5 μA ed 1 nA di sensibilità
- Allarmi: Sovracorrente (ovc), sottotensione (uvv) e sovratensione (ovv).
- Alimentazione: 220/110 V 50/60 Hz 1/2 A

Vset	valore di tensione impostato
Iset	valore di corrente massimo ammesso
Ramp-Up	velocità di salita massima in V/s
Ramp-Down	velocità di discesa massima in V/s
Vmon	Valore di tensione monitorata
Imon	Valore di corrente monitorata
Trip	Tempo massimo che è permesso ad una sovracorrente.

Tabella 1.1: parametri principali dell'apparato

UP	Tensione in salita
DOWN	Tensione in discesa
OVV	OVerVoltage
UNV	UNderVoltage
OVC	OVerCurrent
TRIPPED	raggiunto questo stato viene spento
VMAX	Il canale alla massima tensione

Tabella 1.2: stato dei canali

Nella parte seguente del capitolo vengono descritte le principali caratteristiche del software usato per controllare l'apparato ad alta tensione. Le librerie attualmente in uso vengono usate dal terminale Linux. Come prima parte viene introdotta la scheda di controllo PCI caenet A1303, descrivendone le caratteristiche principali e le funzioni usate per comandare l'apparato ad alta tensione. Di seguito viene proposto uno schema, che descrive la situazione presente nel nostro laboratorio

1.2 Controllo e monitoraggio tramite minicom

Per utilizzare l'alimentatore ad alta tensione c'è la possibilità di controllarlo via porta seriale, tramite l'uso di minicom. Per utilizzare questo emulatore di terminale sono state eseguite le seguenti configurazioni:

[utente@localhost]$ minicom -s

Si deve scegliere l'impostazione *Serial Device*, che è la più importante ossia la porta seriale a cui risulta essere connesso l'hardware. Solitamente in */dev/ttyS0* Le impostazioni principali utilizzate sono:

velocitá	9600 bps
bit di stop	1
paritá	nessuna
hardware flow control	no
Software flow control	no
Serial device	/dev/ttyS0
LockFile location	/var/lock

Nota bene: Ogni utente che utilizza */dev/ttyS0* deve avere i permessi di accesso, quindi bisogna assicurarsi che si abbia l'autorizzazione necessaria, altrimenti da utente root si deve eseguire il seguente comando:

[utente@localhost]$ mchmod 666 ttyS0

1.3 Situazione presente in laboratorio

Figura 1.1: schema situazione laboratorio.

In Fig.1.1 viene proposto lo schema della configurazione hardware presente nel laboratorio. Si vede che: Nel calcolatore è presente la scheda PCI caenet A1303, la quale comanda l'alimentarore SY546 tramite il controller A547. A sua volta l'alimentatore è collegato ai tubi LST.

1.4 Presente software di controllo della scheda CAENET

L'alimentatore viene pilotato tramite un controller PCI CAENET mod. A1303, che permette di controllare l'alimentatore tramite un normale personal computer. Questa scheda è un controller PCI a 32 bit. La linea di comunicazione usa un cavo coassiale a 50Ω. Le librerie già disponibili sono scritte in linguaggio C, come componenti principali, utilizzano i driver forniti dalla CAEN. Le funzioni che fanno uso della scheda caenet utilizzate nelle librerie sono le seguenti:

HSCAENETCardInit	inizializza il controller
HSCAENETSendCommand	invia il comando al modulo
HSCAENETReadResponse	aspetta e legge la risposta dal modulo
HSCAENETCardReset	resetta il controller A1303
HSCAENETCardEnd	resetta le strutture dipendenti dal sistema operativo

HSCAENETCardInit Questa funzione deve essere chiamata prima di ogni utilizzo delle routine. Inizializza le strutture oppurtune del sistema operativo.

HSCAENETSendCommand Spedisce il comando al modulo Caenet.

HSCAENETReadResponse Aspetta una risposta dal modulo caenet, e la memorizza in un buffer di destinazione.

HSCAENETCardReset Procede con il reset della scheda.

HSCAENETCardEnd Questa funzione deve essere chiamato dopo ogni utilizzo delle routines. Si fa carico del reset delle strutture inizializzate in precedenza.

Le librerie C si compongono di cinque moduli:

1. Read System
2. Channels Monitor
3. Set Channels
4. Switch Channels

5. kill All Channels

I principali parametri di input degli eseguibili C sono i seguenti:

- caenetNum: numero della scheda caenet presente nel sistema, in quanto potrebbero essere presenti anche pi schede PCI.
- crateNum: numero del crate, si trova impostato sul crate
- board: numero della board che si vuole spegnere o accendere.
- ON/OFF: è lo stato a cui si vuole impostare una determinata board
- VSET: tensione a cui si vuole portare una board
- ISET: massima corrente sopra il cui valore si passa ad uno stato di overcurrent
- Vmax: massimo valore di tensione impostabile.
- RUP: velocità di salita della tensione in V/s.
- RDWN: velocità di discesa della tensione in V/s.
- TRIP: è il tempo massimo per cui si puó rimanere in uno stato di overcurrent, dopo il quale la *board* viene spenta.

Read System

Il programma *Read System*, effettua un controllo sullo stato della macchina, ottenendo come output uno dei seguenti messaggi:

- OFFLINE: l'apparato è spento.
- SY546 V1.04: il sistema è stato rilevato correttamente.
- un qualunque messaggio di errore.

Principalmente può essere usato per effettuare una prova preliminare, atta a verificarne lo stato di accensione. Questo può risultare utile prima di andare effettivamente a modificarne vari parametri.

In Fig.1.2 viene riportato un esempio di output che il programma produce dopo essere stato avviato:

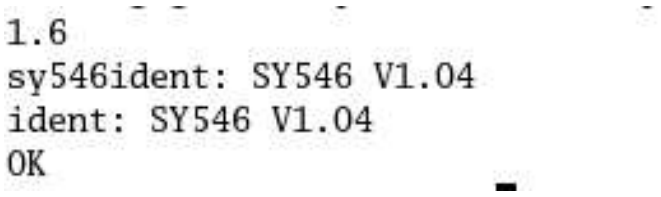

Figura 1.2: output a console del modulo readSystem.

Il comando da digitare sul terminale per eseguire il programma è:
[guest@host]$ ./readSystem caenetNum crateNum[1-8]
Un esempio è: *[guest@host]$ ./readSystem 0 1*
il quale produce un output come mostrato in Fig.1.2

Channels Monitor

Questo modulo, effettua una scansione dello stato dei canali di ogni singola board, legge i parametri di ogni canale.
Ritorna uno dei seguenti messaggi:

- informazioni sullo stato e sui parametri dei canali.
- un qualunque messaggio di errore.

La funzione principale utilizzata per inviare il comando per monitorare l'apparato è:

```
1response=HSCAENETComm(HSCAENETDev, code, crateNum1, (NULL),
                    (0), (NULL))) != TUTTOK
2#define   READ_BOARDS_INFO      0x3
3code = READ_BOARDS_INFO;
```

Il comando inviato alla funzione *HSCAENETComm()* legge le informazione provenienti dalla board. Di seguito viene riportato un esempio di output del programma chMonitor:

I parametri principali mostrati sono:

a Numero del canale

b Tensione Monitorata

c Corrente Monitorata

a	b	c	d	e	f	g	h	i
0:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
1:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
2:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
3:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
4:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
..	...	...	...	...	...	...	...	...
9:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
10:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s
11:	1.50V	0.00nA	Present	600V	1nA	5700V	200V/s	500V/s

Tabella 1.3: Output di Channels Monitor

d Presenza o meno del canale

e Tensione settata dall'utente

f Corrente settata dall'utente

g Tensione Massima

h/i Parametri: Ramp-up e Ramp-down

Nota bene La corrente monitorata varia per ogni singolo canale, mentre la tensione è uguale su tutti i canali di una board.

Il comando da digitare sulla shell per eseguire il programma è:
[guest@host]$./chMonitor caenetNum crateNum[1-8] board[0-7] log_slotInfo[0-1]

un esempio:
[guest@host]$./chMonitor 0 1 1 0

Set Channels

setchannels, imposta i parametri per tutti i canali.
Ritorna uno dei seguenti messaggi:

- OK.
- un qualunque messaggio di errore.

La funzione principale utilizzata per inviare il comando per monitorare l'apparato è:

```
response=HSCAENETComm(HSCAENETDev, code, crateNum1, (NULL),
                    (0), (NULL))) != TUTTOK
```

Il comando da digitare sulla shell per eseguire il programma è:

[guest@host]$./setChannel caenetNum crateNum[1-8] board[0-7]
VSET[V] ISET[nA] Vmax[V] RUP[V/s] RDWN[V/s] TRIP
un esempio:
[guest@host]$./setChannels 0 1 1 500 1000 5700 200 500 60

Switch Channels

SwitchChannels, accende o spegne le board presenti nell'apparato. Ottiene uno dei seguenti messaggi in uscita:

- OK: l'operazione eseguita con successo.
- un qualunque messaggio di errore.

Di seguito viene mostrato la parte del codice sorgente che si occupa di inviare il comando all'apparato.

```
1   if (turn == 1)
2 {
3 if ((response=HSCAENETComm(HSCAENETDev, code, crateNum)) !=
                    TUTTOK)
4   {
5    printf("HSCAENETComm switch On failed. %s\n",
                      DecodeCAENETResp(response));
6    endCard();
7    exit(-1);
8   }
9 }
10 else if (turn == 0)
11 {
12 if ((response=HSCAENETComm(HSCAENETDev, code, crateNum)) !=
                    TUTTOK)
13    {
14     printf("HSCAENETComm switch Off failed. %s\n",
                       DecodeCAENETResp(response));
```

```
15     endCard();
16     exit(-1);
17    }
18 }
```

Si puó notare che a seconda del parametro passato, ON o OFF, si invia alla scheda caenet il comando relativo all'accensione (riga 3) o allo spegnimento (riga 12)

Il comando da digitare sulla shell per eseguire il programma è:
[guest@host]$./switchChannels caenetNum crateNum[1-8] board[0-7] ON/OFF[1-0]

Un esempio viene qui riportato:

[guest@host]$./switchChannel 0 1 1 0

Kill All Channels

Kill All Channels, imposta a zero tutti i canali, letteralmente: *uccisione* dei canali. Nel caso si venga a verificare un'anomalia o un errore per il quale si debba avere un immediato spegnimento dei canali dell'alimentatore, si può eseguire questo programma il quale scaricare a massa con il risultato di un immediato spegnimento dei canali.

La parte principale del codice che si occupa di effettuare questa operazione è la seguente:

```
1  if ((response=HSCAENETComm(HSCAENETDev, code, crateNum))!=
                    TUTTOK) {
2          printf("HSCAENETComm channels kill failed. %s\n",
                              DecodeCAENETResp(response));
3          endCard();
4          exit(-1);
5   } else {
6      code = CONFIRM_KILL_CHANNELS;
7      if ((response=HSCAENETComm(HSCAENETDev, code, crateNum
                         ))!=TUTTOK) {
8          printf("HSCAENETComm channels kill failed. %s\n",
                              DecodeCAENETResp(response));
9          endCard();
10         exit(-1);
11     }
12  }
```

Si nota che per effettuare il processo, vengono inviati due comandi, il primo per indicare all'apparato che si vuole procedere con l'operazione, mentre il secondo

consiste in una conferma del comando, dopo il quale si effettua l'operazione di spegnimento.
Il comando da digitare sulla shell per eseguire il programma è:

[guest@host]$./killAllChannels caenetNum crateNum[1-8]
Un esempio è il seguente:
[guest@host]$./killAllChannel 0 1

1.5 Diagramma a blocchi

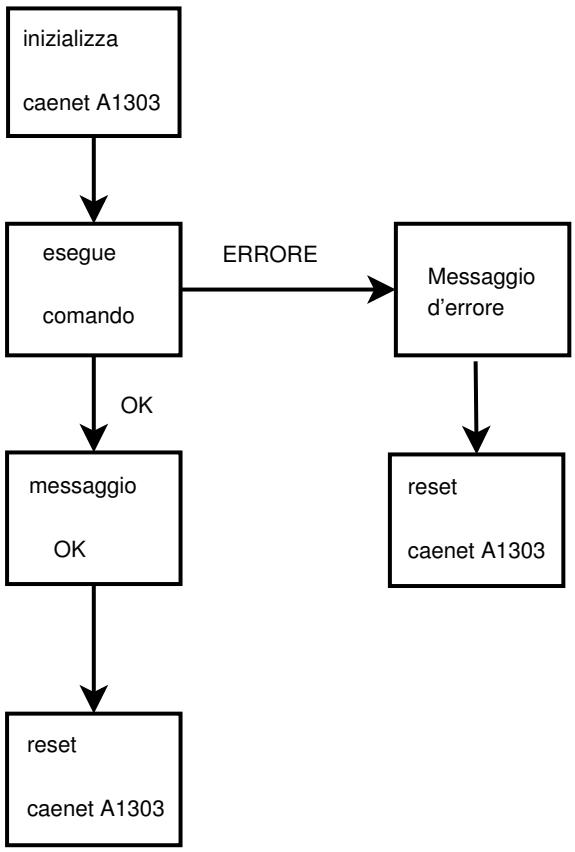

Figura 1.3: Diagramma a blocchi del funzionamento principale delle librerie C.

In Fig.1.3 viene riportato il diagramma che illustra il funzionamento delle librerie. Queste librerie descritte, venivano utilizzate in programmi realizzati con Labview, ma con uso che risultava poco pratico per il nostro scopo cioè realizzare una procedura automatica per il condizionamento dei tubi LST e la realizzazione

di una procedura per la misura dei Plateau. Risultano invece molto pratiche se utilizzate da shell. Nel prossimo capitolo si procederà alla descrizione di come sono state realizzate utilizzando Labview.

Capitolo 2

Librerie realizzate con Labview

Nel seguente capitolo si dà un introduzione generale al software della National Instrument Labview, poi viene descritto come sono state realizzate le librerie descritte nel capitolo precedente utilizzando il programma Labview. Vengono presentati anche i rispettivi pannelli frontali.

2.1 Introduzione a Labview

Labiew[1] è l'ambiente di sviluppo integrato per il linguaggio di programmazione visuale di National Instrument. Tale linguaggio grafico viene chiamato: Linguaggio G. Originalmente realizzato da Apple nel 1986, Labview utilizzato principalmente per acquisizionee analisi dati, controllo dei processi, generazione di rapporti, o più in generale tutto ciò che concerne l'automazione industriale.

Programmazione G

Il linguaggio usato in Labview è grafico. Un programma o un sottoprogramma viene denominato VI[2], non esiste sotto forma di testo, ma può essere salvato solo come file visualizzabile e compilabile solamente da Labview. La definizione di strutture dati ed algoritmi avviene con icone e altri oggetti grafici, ognuno dei quali incapsula funzioni diverse, uniti da linee di collegamento[3], in modo da formare una sorta di diagramma di flusso.

[1] abbreviazione di LABoratory Virtual Instrument Engineering Workbench
[2] Virtual Instrument
[3] wire

Dettagli dei VI

Nell'ambiente di sviluppo, i VI constano di tre componenti principali:

- Il pannello frontale
- Diagramma a blocchi
- Il riquadro connettori

Pannello Frontale

Il pannello frontale è l'interfaccia utente del VI. Si realizza con controlli ed indicatori, che costituiscono i terminali interattivi di ingresso ed uscita. I controlli sono: matrici, manopole, potenziometri, pulsanti, ecc; simulano i dispositivi d'ingresso degli strumenti e forniscono dati allo schema a blocchi del VI. Gli indicatori sono: grafici, tabelle, led, termometri, ecc; simulano i dispositivi d'uscita degli strumenti e visualizzano i dati che lo schema a blocchi acquisisce o genera.

Schema a blocchi

Lo schema a blocchi è il diagramma di flusso che rappresenta il codice sorgente in formato grafico. Gli oggetti del pannello frontale appaiono come terminali di ingresso o uscita nello schema a blocchi. Gli oggetti dello schema a blocchi comprendono: terminali, funzioni, costanti, strutture, chiamate ad altri VI, fili di collegamento, commenti testuali.
Le funzioni sono chiamate esse stesse VI. Possono avere un numero indefinito di ingressi e di uscite come ogni VI. Le strutture eseguono il controllo di flusso di base. Al esempio un ciclo for è rappresentato da un contenitore quadrato, che ripete N volte la porzione di schema a blocchi presente al suo interno. I fili di collegamente possono trasportare teoricamente qualunque mole di dati di qualunque tipo, anche aggregati definiti dal programmatore. Il colore e lo spessore del filo cambia a seconda del tipo di dato. Ad esempio gli interi scorrono su fili blu. Lo schema a blocchi può essere reso visibile anche durante l'esecuzione, cosa molto utile in fase di debug, in quanto è possibile visualizzare un animazione al rallentatore del movimente dei dati lungo i fili presenti nello schema.

Riquadro connettori

Ogni VI può essere a sua volta utilizzato come subVI[4] e comparire all'interno dello schema a blocchi di altri VI, proprio come una qualsiasi funzione, e come tale può avere ingressi ed uscite a cui collegare le linee di flusso.

Eseguibili A partire dai VI si possono anche creare eseguibili a sè stanti e librerie condivise, perché Labview è un vero compilatore a 32bit. Per usare tali eseguibili non occorre un installazione di Labview, ma è necessario che sul computer di destinazione sia installato il run-time engine di Labview.

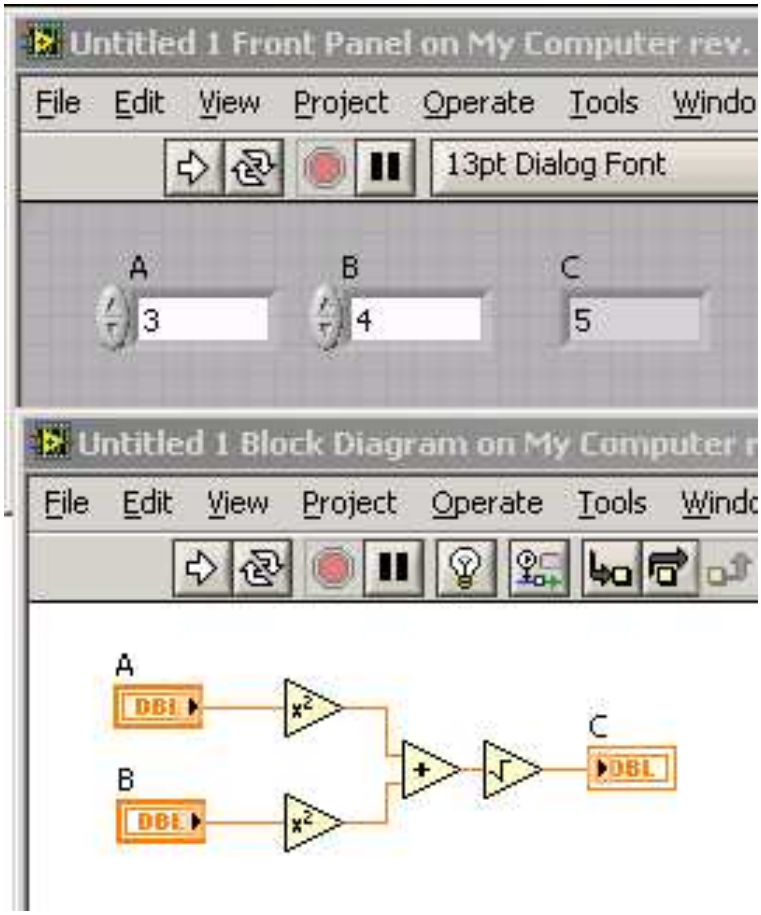

Figura 2.1: Esempio di pannello e schema a blocchi Labview.

[4] sotto VI

2.2 Realizzazione delle Librerie

Per la realizzazione delle librerie con Labview, principalmente è stato utilizzato una funzione chiamata Code Interface Node, presente in Labview. Questo oggetto permette di richiamare codice sorgente scritto in C da Labview.

Modulo Code interface Node (CIN)

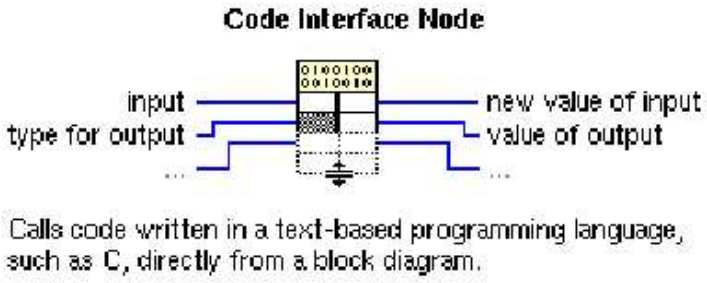

Figura 2.2: Code Interface Node.

Alla funzione CIN può essere passato un qualunque numero di parametri, sia in ingresso sia in uscita. I quali possono essere elaborati dal codice C che si vuole richiamare. L'utilizzo del modulo CIN, prevede i seguenti passi:

1. Crea il CIN connettendo in ingresso ed uscita i parametri necessari;
2. Salva il VI;
3. Crea il file .c per il modulo CIN usato.

Una volta creato è il .c che in generale si presenta in questo modo:

```
1#include "extcode.h"
2
3CIN MgErr CINRun (tipo param1, tipo param2, ... );
4
5CIN MgErr CINRun (tipo param1, tipo param2, ... )
6{
7
8  ...Inserire qui il proprio codice...
9
10  return noErr;
11}
```

Si nota dal codice proposto sopra, che la funzione CINRun() risulta essere il main che il code interface node utilizza. Infatti è all'interno di essa che deve essere inserito il proprio codice, il quale verrá utilizzato per la risoluzione di un determinato problema. I parametri presenti nella funzione CINRun() sono effettivamente controlli e indicatori presenti nel pannello Labview.

Procedura per la generazione del file .lsb Per far sì che il modulo utilizzi il codice dell'utente, si deve generare un file .lsb a partire dal sorgente C. Per far ciò si deve utilizzare un tool fornito dalla National Instrument, *lvmkmf*. Questo tool genera un Makefile, da utilizzare per compilare il sorgente realizzato. Al termine della compilazione, verrà generato il file .lsb che potrà essere caricato dal modulo CIN.

Esempio Makefile generato con tool *lvmkmf*

```
# Questo make file viene generato in automatico da lvmkmf.
#
CC=gcc
LD=gcc
LDFLAGS=-shared
XFLAGS=-fPIC -O
CINDIR=/usr/local/lv80/cintools
CFLAGS=-I$(CINDIR) $(XFLAGS)
CINLIB=$(CINDIR)/libcin.a
MAKEGLUE=$(CINDIR)/makeglueLinux.awk
AS=gcc -fPIC -c
.SUFFIXES: .lsb .lsb~ $(SUFFIXES)
# Default rule to create an lsb from a C source file
.c.lsb: ; make $*.o
        $(LD) $(LDFLAGS) -o $*.tmp \
                $(CINDIR)/cin.o $(XLDFLAGS) $*.o $(CINLIB)
        $(CINDIR)/lvsbutil -c $* -t CIN -d "‘pwd‘"
        @rm -f $*.tmp
prova.lsb:      prova.o
        $(LD) $(LDFLAGS) -o prova.tmp \
        $(CINDIR)/cin.o $(XLDFLAGS) prova.o  $(CINLIB)
        $(CINDIR)/lvsbutil -c prova -t CIN -d "‘pwd‘"
        @rm -f prova.tmp
clean:
        $(RM) -f prova.o prova.tmp
spotless: clean
        $(RM) -f prova.lsb
```

Il Makefile è generato a partire dal sorgente prova.c, col comando:
[root@domain]$ lvmkmf prova.
Nel nostro caso, il Makefile è stato modificato in modo che quando si compila il sorgente si vengano ad utilizzare le librerie CAEN. In particolare sono state fatte le seguenti modifiche:

modifica uno aggiunta riga: *CAENFLAGS=-lhscaenet*

modifica due riga 20: modificato da:

$(LD) $(LDFLAGS) -o readsystem.tmp

in:

$(LD) $(LDFLAGS) $(CAENFLAGS) -o readsystem.tmp

questa modifica fa si che alla compilazione usando il Makefile, il compilatore utilizzi le librerie CAEN.

schema a blocchi per la creazione file lsb

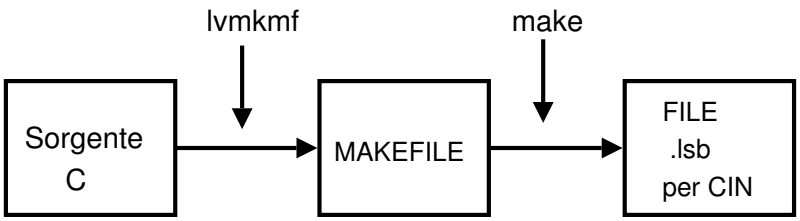

Figura 2.3: fasi per la creazione del file lsb per il CIN

2.3 Read System

Il modulo Read System è stato realizzato utilizzando come unità principale il modulo CIN disponibile in Labview. In sostanza la realizzazione software schematizzata in nel capitolo 1 in Fig.1.3 , viene eseguita dal modulo CIN. Come si può vedere in Fig.2.4 il code inteface node ha tre parametri. I parametri sono:

- Numero Caenet[5]
- Numero Crate[6]
- codice di output

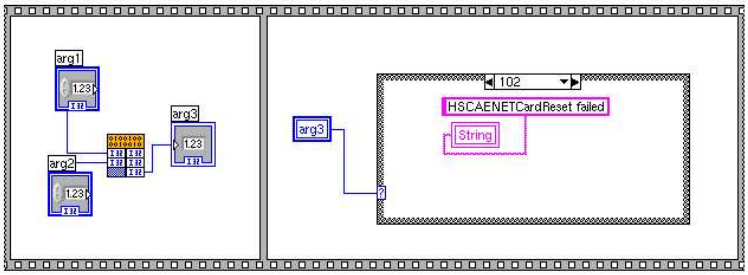

Figura 2.4: Diagramma a blocchi

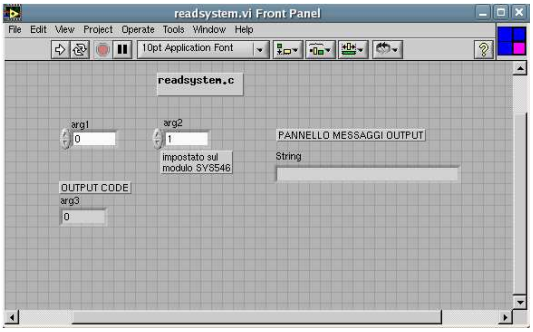

Figura 2.5: Pannello Frontale

In base al codice che viene generato come output del modulo CIN, viene restituito all'utente un messaggio di OK, oppure un qualsiasi messaggio d'errore.

I messaggi sono i seguenti:

- SYS 546 v.104 OK
- HSCAENETComm system detect failed
- HSCAENET Card Init Failed
- HSCAENET Card Reset Failed

[5]identificatore scheda caenet
[6]viene impostato sull'alimentatore

- HSCAENET Card Timeout Failed

I messaggi d'errore vengono generati nel caso in cui l'alimentatore sia spento, oppure che vi siano errori di comunicazione tra il PC e la scheda PCI CAENET A1303, ad esempio nel caso di una inizializzazione fallita.
Di seguito viene riportato un esempio del sorgente C utilizzato dal modulo CIN:

```
1 /* CIN source file */
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5 #include <errno.h>
6
7 #include "extcode.h"
8 #include "libreria.h"
9
10 MgErr CINRun(int32 *arg1, int32 *arg2, int32 *arg3);
11
12 MgErr CINRun(int32 *arg1, int32 *arg2, int32 *arg3)
13 {
14 int i, crateNum;
15 unsigned char tempBuff[24];
16
17 a1303index = *arg1;
18 crateNum = *arg2;
19
20 *arg3 = initCard();
21
22 if (*arg3==101){return noErr;}
23 if (*arg3==102){return noErr;}
24 if (*arg3==103){return noErr;}
25
26 code = BOARDS_IDENT;
27 response=HSCAENETComm(HSCAENETDev,code,crateNum,(NULL),(0)
                    ,(tempBuff));
28
29 if (response != TUTTOK) {
30         *arg3 = 100;
31         endCard(); }
32 else    {
33         *arg3 = 111;
34         }
35 endCard();
```

```
36return noErr;
37}
```

Nel esempio sopra riportato si vede la funzione CINRun(), all'interno della quale stato inserito il codice che il code interface node andrà ad eseguire.

Principali funzioni presenti nel sorgente:

- riga 20: initCard() inizializza la scheda CAENET a1303, preparando le strutture oppurtune del sistema operativo;
- riga 27: HSCAENETComm() Spedisce il comando alla scheda caenet;
- riga 31/35: endCard() invia il reset alla scheda CAENET a1303 facendosi carico del reset delle strutture inizializzate in precedenza.
- riga 27: code è il codice del comando da inviare alla caenet ad esempio:
  - 0x35 Kill all Channels
  - 0x36 confirm Kill all Channels
  - 0x3 Read boards characteristics
  - 0xo Board Identifier
- riga 30/33: *arg3 codice di output del programma
  - 111 comando eseguito con successo
  - 100 Modulo spento: HSCAENETComm System detect failed
  - 101 Inizializzazione fallita: HSCAENETCardInit failed
  - 102 Reset Fallito: HSCAENETCardReset failed
  - 103 Timeout: HSCAENETTimeout failed

2.4 Channels Monitor

Questo modulo esegue il monitoraggio dello stato dei canali dell'alimentatore. Viene qui riportato il pannello frontale

In Figura 2.6 si notano i parametri di ingresso nel riquadro *STEP2*, che sono:

- caenNum: numero di riferimento alla scheda PCI caenet a1303.

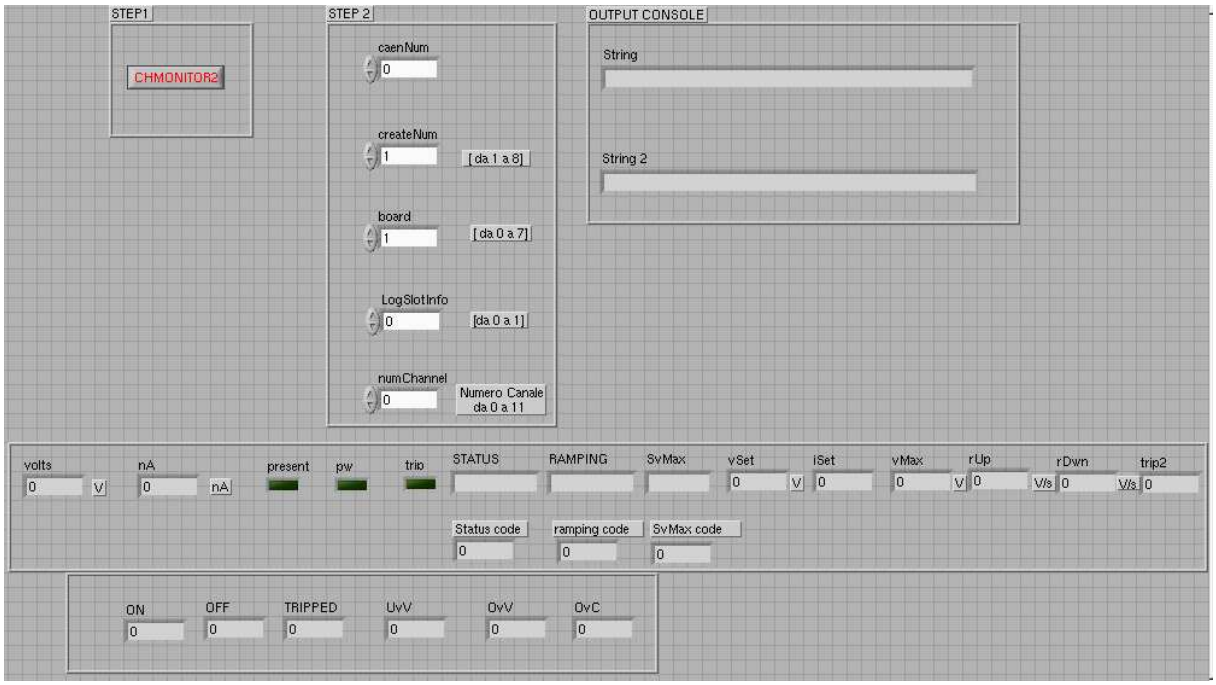

Figura 2.6: pannello frontale ChMonitor

- crateNum: numero del crate.
- board: il numero della board da monitorare.
- numChannel: Numero del canale da monitorare.

mentre i parametri di uscita sono:

- volts: Tensione misurata.
- nA: Corrente misurata.
- flag: Present, pw e trip: canale presente, canale acceso/spento e canale in uno stato di trip.
- status: visualizza lo stato dei canali, che può essere ad esempio in overcurrent, undervoltage ecc.
- Ramping: indica l'andamento della tensione, se in aumento o diminuzione in base ai parametri Ramp-up e Ramp-down
- Vset: tensione impostata dall'utente.

- Iset: corrente impostata dall'utente.
- trip: tempo in secondi, per i queli un canale può rimanere dello stato di overcurrent.
- ON-OFF-TRIPPED-UvN-OvV-OvC: indicano il numero dei canali che si trovano nello stato indicato, cioè quando accesi, spenti, undervoltage, overvoltage e overcurrent.

Sono presenti due box di testo che visualizzano un qualunque messaggio d'errore.

Channels monitor utilizza come componente principale il CIN. Al sorgente C originale sono state apportate le seguenti modiche:

È stato sostituito il main del sorgente originale, con la funzione *CINRUN()* che viene utilizzata dal CIN per l'esecuzione dei sorgenti da Labview.

Tutte le *printf()* presenti sono state omesse, e i messaggi di output che il programma restituisce sono gestite da Labview. Ad esempio nella funzione *initcard()*:

```
1         puts("HSCAENETCardInit failed");
```

viene sostituito con la seguente riga.

```
1         return 100;
```

Quindi in caso di fallimento nell'inizializzazione della scheda caenet, viene prodotto un codice d'errore che il software interpreta restituendo a video il messaggio *HSCAENETCardInit failed*

2.5 Set Channels

Il modulo setChannels, serve a impostare i parametri per l'alimentatore. Di seguito viene riportato il pannello frontale realizzato tramite Labview:

Come si vede dalla Figura 2.7 i parametri che si possono settare sono i seguenti:

- caenNum, crateNum e board: Identificano la scheda caenet, il crate e la board a cui applicare i parametri.
- Vset: tensione a cui si vuole impostare la board

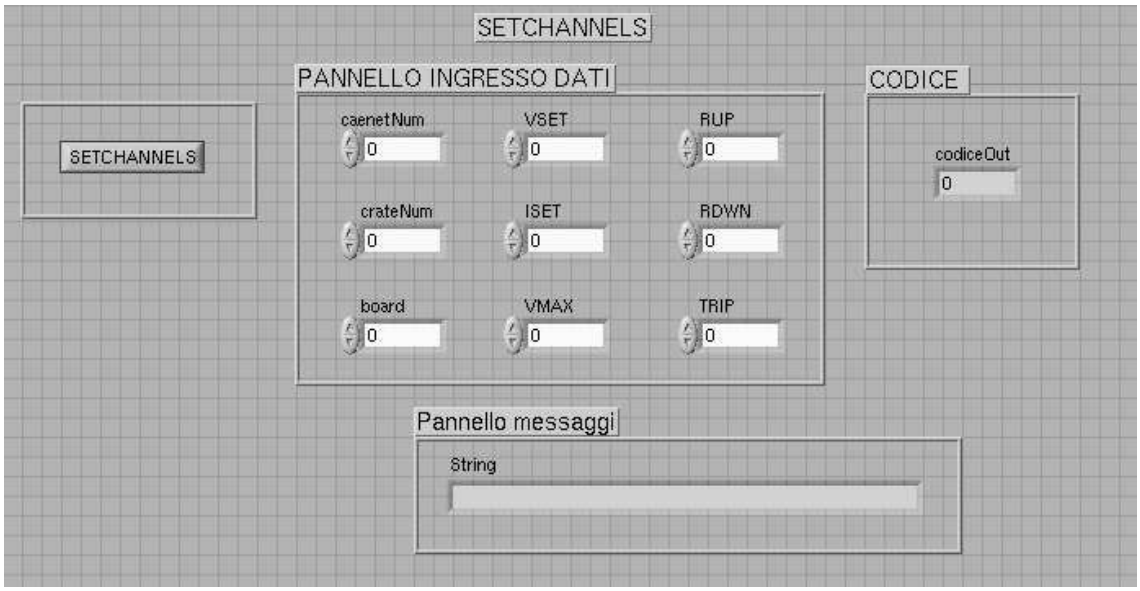

Figura 2.7: pannello frontale setChannels

- Iset: corrente massima di funzionamento di un canale
- Vmax: valore di tensione massimo permesso
- Rup: massimo valore in V/s con cui la tensione passa da un valore più basso ad uno più alto
- Rdwn: massimo valore in V/s con cui la tensione passa da un valore più alto ad uno più basso
- trip: è il tempo massimo per cui si può rimanere in uno stato di overcurrent.

2.6 Switch Channels

Il modulo switch channels, spegne o accende le board presenti nel sistema. Il pannello frontale realizzato con labview viene riportato qui di seguito:

I parametri che si possono impostare tramite il pannello sono:

- caenNum, crateNum e board: Identificano la scheda caenet, il crate e la board a cui applicare i parametri.
- on/off, il parametro con cui viene acceso oppure spento la board.

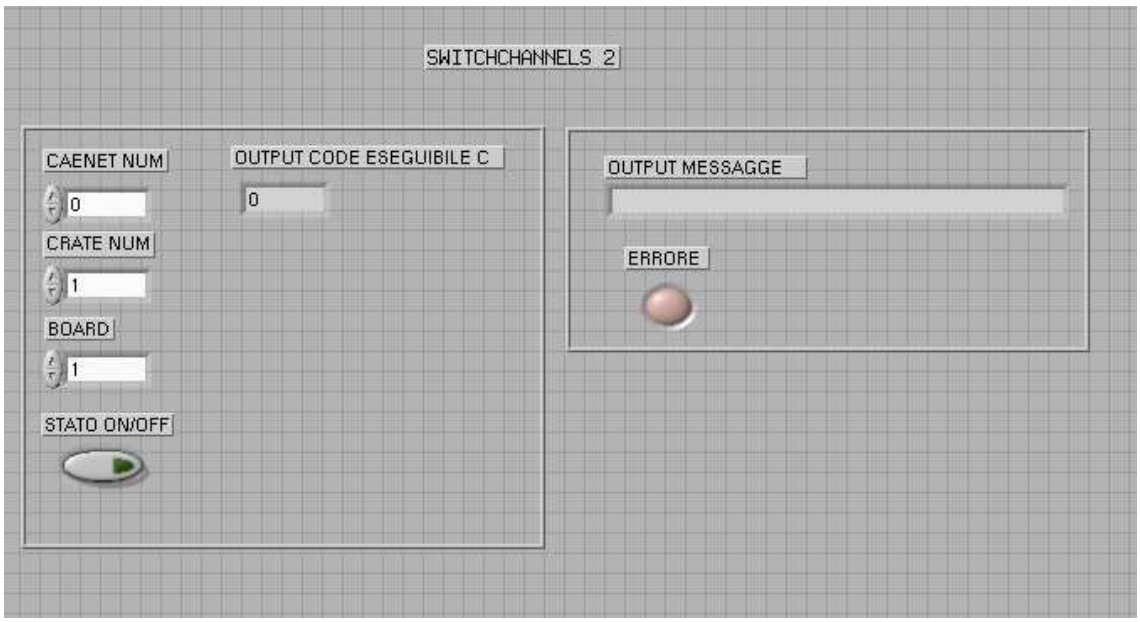

Figura 2.8: pannello frontale switchChannels

Nel pannello è possibile visualizzare anche qualunque tipo di messaggio d'errore e di OK.

2.7 Kill All Channels

Questo modulo esegue lo spegnimento di tutti i canali. Può essere usato in caso di anomalia o qualche tipo di errore, per il quale sia necessario l'immediato spegnimento delle board. Di seguito viene riportato il pannello frontale dell'applicazione costruita con Labview:

I passi principali che il modulo compie per eseguire la procedura che spegne tutte board, sono eseguite dal modulo CIN presente nell'applicazione labview. Di seguito viene riportato uno spezzone di codice, il quale esegue il comando di spegnimento di tutti i canali:

```
1response=HSCAENETComm(HSCAENETDev, code, crateNum1, (NULL),
                    (0), (NULL))) != TUTTOK
```

In conclusione queste librerie utilizzano come modulo base il CIN, vengono usate per la realizzazione di applicazioni più complesse tramite Labview. Ad esempio procedure automatiche che richiedano l'uso dell'alimentatore ad alta

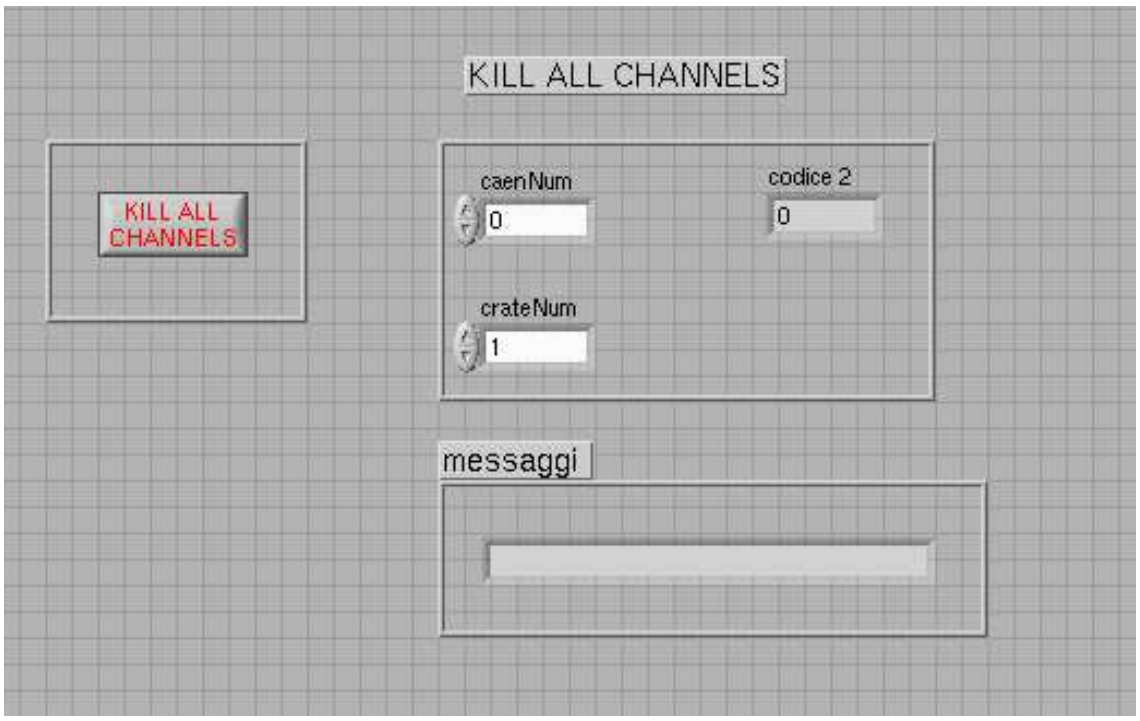

Figura 2.9: pannello frontale KillAllChannels

tensione.
Nei prossimi capitoli verranno trattate due applicazioni:

i Una proceduta automatica per il condizionamentodei tubi LST

ii Una procedura per la misura automatica dei Plateau

Capitolo 3

Procedura automatica di condizionamento

In questo capitolo si dà uno sguardo generale alla procedura di condizionamento dei tubi LST, passando poi alla descrizione dell'applicazione software realizzata per effetuare tutto in modo automatico. Questa applicazione è data dalla necessità di avere una modalità automatica, la quale gestisca in modo autonomo il cambio di tensione con cui i rivelatori sono alimentati. Il problema principale è il fatto che per ottimizzare la proceduta i tubi devono essere sottoposti a tensioni variabili dai 4500 ai 5900 V per periodi di tempo anche lunghi. Bisogna quindi produrre una procedura automatica e sicura, dovuto dal fatto che se li camere assorbono troppa corrente per un periodo di tempo prolungato rischiano di danneggiarsi irrimediabilmente.

3.1 Procedura di condizionamento dei tubi

Una delle procedure necessarie per la messa in funzione dei rivelatori LST, è il condizionamento: nel tubo appena costruito e messo in tensione si osservano fenomeni di scarica causati da particelle di polvere, punte di graffite e umidità. Per la soluzione di questo problema si alimenta il tubo partendo da tensioni relativamente basse, e aumentandone graduatamente il valore. Questo metodo fa si che si brucino lentamente le impurezze, rendendo i tubi pronti per la messa in funzione definitiva. Al termine del condizionamento il tubo deve essere in grado di sopportare una tensione di 5700 V per un tempo indeterminato ed un valore di corrente non superiore a 200 nA.

Tappe del condizionamento

L'alimentazione dei tubi fornita dall'alimentatore ad alto tensione CAEN SY546. Prima dell'inizio della procedura il tubo viene flussato con la miscela di gas ternaria per almeno 24 ore. La presenza di aria all'interno della camera deve essere ridotta al minimo a causa dell'elettronegatività dell'ossigeno.

La camera che viene sottoposta a tale procedura deve raggiungere la tensione prestabilita senza trovarsi in uno stato OVC. [1] altrimenti sarà classificata come Anomala. A questo punto potrebbe essere ripetuto il processo di condizionamento, oppure potrebbe essere aperta la camera, nel caso in cui si manifesti un grave problema di malfunzionamento. La condizione per cui si ritenga lo step superato, segue il seguente criterio:

1. Step superato: se la corrente massima risultante dalla somma su tutti i canali del tubo inferiore ai 2000 nA

2. Step non superato: se la corrente massima risultante è superiore ai 2000 nA.

Il fatto che una camera, si venga a trovare in uno stato anomalo, può dipendere da una costruzione diffettosa del tubo. Il caso di step non superato si verifica quando nella camera è presente un fenomeno di scarica; in questa circostanza la tesione viene abbassata di 300 V ed il condizionamento riprende dal punto raggiunto con l'abbassamento della tensione. Ogni tubo subisce un condizionamento elettrico per un minimo di 7 giorni, questo accade in fase di produzione altrimenti il condizionamento risulta essere di durata inferiore. Il tutto si svolge nel seguente modo:

- ogni tubo per un minimo di 8 ed un massimo di 24 h raggiungerà i 2500 V; successivamente per un minimo di 8 ed un massimo di 24 h viene portato da 2500 V a 4900 V.

- ogni tubo che supera una corrente di 5 μA viene scartato

- i tubi sono mantenuti alla tensione di 4900 per 96h. La corrente non deve superare i 2 μA per il primo giorno e per i rimanenti 3 giorni non deve oltrepassare i 150 nA per massimo del 5% del tempo. I tubi che non soddisfano tali condizioni saranno scartati.

Di seguito un esempio di tensioni di alimentazione che una camera deve supportare, durante la fase di produzione:

[1] stato di sovracorrente, overcurrent

Step tensione e tempo per il condizionamento delle camere

Step	Tensione	Intervallo
1	3500 V	120 min
2	4500 V	240 min
3	4700 V	240 min
4	4850 V	240 min
5	500 V	240 min

Step	Tensione	Intervallo
6	5000 V	60 min
7	5050 V	60 min
8	5100 V	60 min
-	- - -	- - -
15	5450 V	60 min

Step	Tensione	Intervallo
16	5500 V	60 min
17	5550 V	60 min
18	5600 V	60 min
-	- - -	- - -
23	5850 V	60 min

Step	Tensione	Intervallo
24	5700 V	16 ore

Tabella 3.1: confronto conteggi contatore NIM e applicazione Labview

Come si nota dalle Tab.3.1 la durata della fase di condizionamento si parte da una tensione di 3500 V fino ad arrivare ad una tensione finale di 5700 V. La durata totale di queste fasi si aggira attorno alle 52 ore. Durante questo periodo di tempo si dovrebbe far in modo che le camere durante il funzionamento vero e proprio riescano a resistere ad una tensione di alimentazione di 5700 per un intervallo di tempo non determinato. Questo aumento proprorzionale di tensione, come spiegato precedentemente dovrebbe far si che vengano eliminate eventuali anomalie di costruzione, come ad esempio presenza di particelle di polvere all'interno della camera. Per la nostra applicazione i tempi sono ridotti significativamente, il fatto di effettuare un condizionamento serve ad eliminare eventuali tracce di impurità e umidità che si producono a seguito di un inutilizzo prolungato dei tubi.

3.2 Condizionamento eseguito in laboratorio

Il condizionamento descritto fino a questo momento avviene principalmente in fase di produzione dei tubi LST, nel nostro caso in cui i tubi sono già stati utilizzati i tempi per eseguire il condizionamento vengono drasticamente ridotti. Questa procedura serve per riportare ad uno stato di buon funzionamento i tubi. Infatti dopo un lungo periodo si scarso o nullo utilizzo potrebbero essere presenti all'interno delle camere umidità ed altre impurità. Quindi per eliminare tali impurezze si effettua la procedura con tempi più contenuti.
Ad esempio, si possono eseguire una quindicina di step con valori di tensione che variano da 4500 V a 5600 V con tempi ad esempio di un ora a step.

3.3 Applicazione Labview: condizionamento

Per la realizzazione di questa procedura automatica, si sono svolti i seguenti passi:

Passo 1 Per prima cosa, sono state effettuate delle prove di funzionamento delle librerie già presenti, al fine di ottenere un rapporto sulle modalità di funzionamento di esse.

Passo 2 Cercare di utilizzare funzioni presenti in labview, in modo da riuscire a gestire completamente l'alimentatore ad alta tensione tramite esso.

Passo 3 Creare una metodo del tutto automatizzato, che preveda la scelta iniziale dei parametri, tra cui valore di tensione, di corrente, tempi ecc, tramite un interfaccia grafica. Questa sempre gestita da Labview. Una volta impostati i parametri, si preveda l'avvio dell'applicazione, che terminerà a procedura eseguita.

Passo 4 Effettuati test sul funzionamento dell'applicativo.

Per la realizzazione di questa applicazione, vengono usati come componenti principali le librerie sviluppate, e descritte nel capitolo precedente. Questa procedura risulta essere una delle applicazioni dell'uso delle librerie create. Come si nota, la semplicità d'uso di esse, si pone nel risultato finale che risulta essere quello di inserire un VI nel diagramma principale. Nella parte successiva del capitolo vengono illustrate le parti principali del programma.

Pannello Frontale dell'applicazione

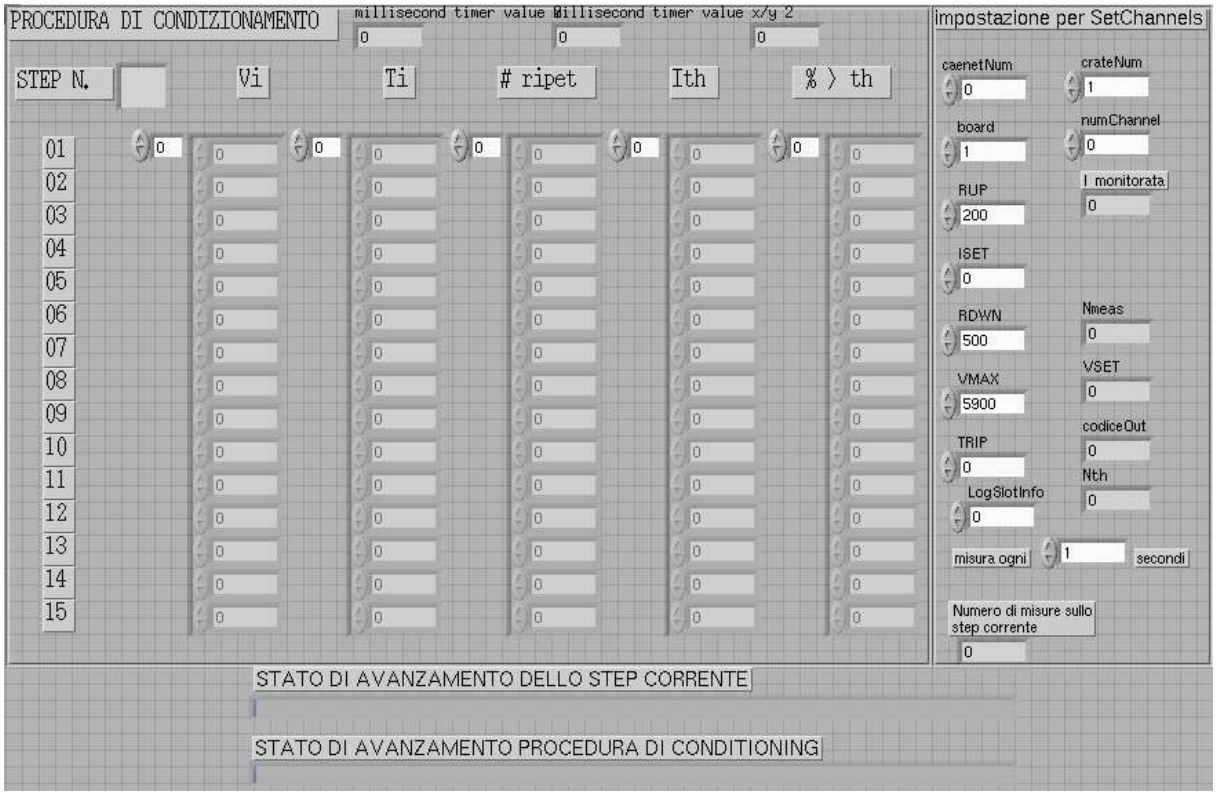

Figura 3.1: Interfaccia utente dell'applicazione per il condizionamento dei tubi LST.

Come si nota dalla Fig.3.1, l'intefaccia che l'utente ha a disposizione si può suddividere in due parti principali: Nella parte centrale vi sono i settaggi che corrispondono agli step di tensione. Infatti come prima cosa da fare, prima di avviare la procedura si devono scegliere i valori di tensione, i tempi per ogni step, il numero di ripetizioni che un qualunque step può subire, la corrente di soglia, e la percentuale in cui la corrente monitorata può superare la soglia impostata.
La ripetizione o meno dello step corrente dipende da due fattori:

Caso 1 Se la corrente supera la corrente di soglia per la percentuale di volte impostate, si deve rieseguire lo step.

Caso 2 Se il numero di volte che la corrente misurata supera la corrente di soglia è inferiore alla percentuale settata allora lo step si considera superato e si procede allo step successivo.

Nella parte di destra del pannello sono presenti impostazioni generali quali: Numero Caen, Numero del Crate, Numero della board, il rump-up, il rump-down, la corrente massima, la tensione massima, i secondi per il trip-time. È inoltre presente un campo, per decidere ogni quanto tempo effettuare la misura della corrente durante i singoli step. Questo valore non deve essere inferiore ad 1 secondo.

Nella parte sottostante sono presenti due barre di avanzamento: la prima per l'avanzamento totale, la seconda indica l'avanzamento di ogni singolo step. Un esempio di impostazione per utilizzare la procedura è il seguente:

volts	tempi (min)	numero rip.	I soglia	perc.
3500	120	3	1000	50
3700	120	3	1000	50
3800	120	3	1000	50
3900	120	3	1000	50
4000	120	3	1000	50
4100	120	3	1000	50
4200	120	3	1000	50
4300	120	3	1000	50
4400	120	3	1000	50
4450	120	3	1000	50
4500	120	3	1000	50
4550	120	3	1000	50
4600	120	3	1000	50
4650	120	3	1000	50
4700	240	3	1000	50

Tabella 3.2: esempio di impostazioni per la procedura di condizionamento

Vmax	5900
Imax	1000
Rup	200
Rdwn	500
trip	60
tempo monitoring I	60

Tabella 3.3: parametri per Set Channels

3.4 Realizzazione dell'applicazione

Il diagramma a blocchi che riassume il funzionameno della procedura è illustrata in Fig.3.2:

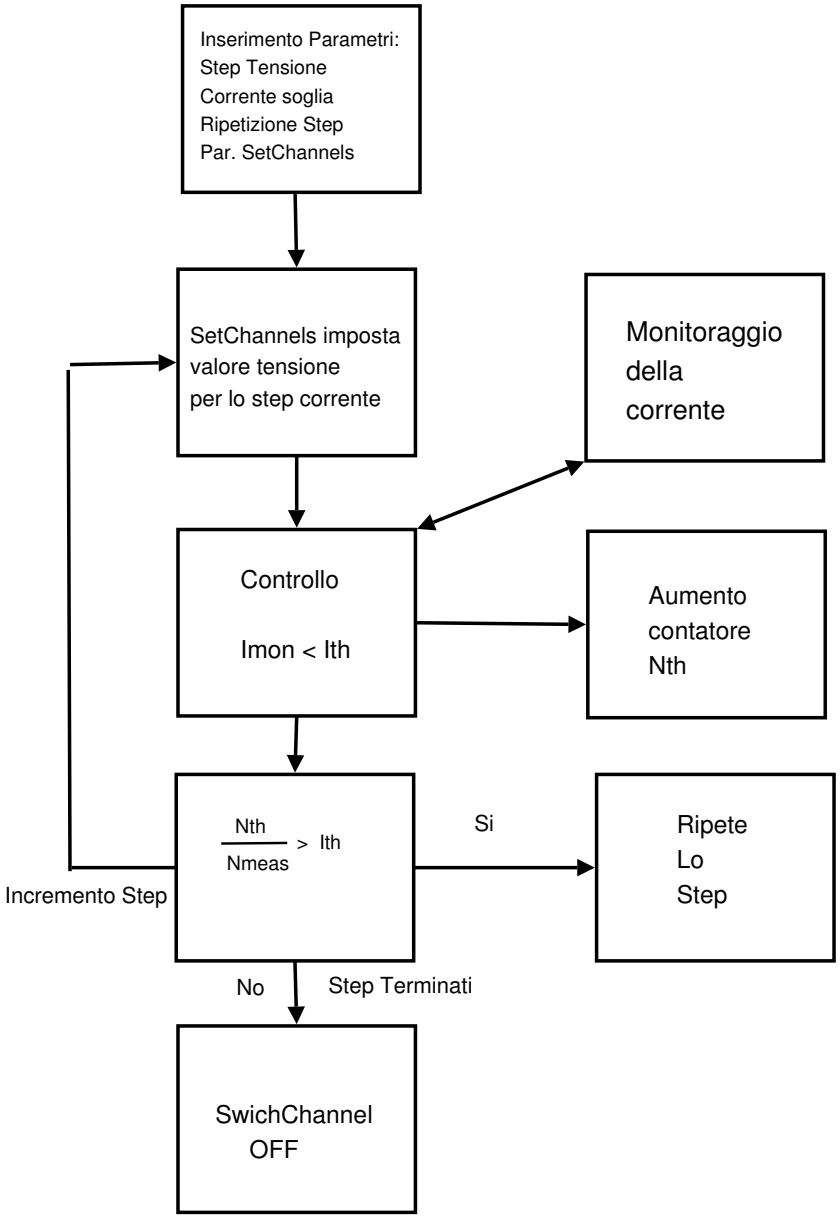

Figura 3.2: diagramma a blocchi che descrive il funzionamento principale della procedura di condizionamento realizzata tramite labview.

Array per l'inserimento dei parametri

Per la parte relativa all'inserimento dei parametri da usare per il condizionamento, a livello di programmazione sono stati usati cinque array, qui sono rappresentati: Gli array sopra rappresentati, con numerazione che va da 1 a 5 cono

Figura 3.3: array utilizzati per la gestione dei parametri di ingresso.

rispettivamente utilizzati per immagazzinare le seguenti informazioni:

1. Valori di tensione per gli step.
2. Tempi di durata per ogni step.
3. Numero di ripetizioni che uno step può subire
4. Corrente di soglia.
5. Percentuale in cui la corrente monitorata può superare la soglia.

L'impostazione del valore di tensione a cui l'apparato ad alta tensione deve funzionare per lo step corrente è effettuato dalla libreria setChannels. In Fig.3.4 viene riportato la parte dello schema a blocchi atta ad effettuare l'operazione:

Dopo aver impostato il corretto valore di tensione, viene eseguito un controllo sulla corrente[2]. La misura viene effettuata ad un intervallo di tempo che viene deciso dal utente, impostando il relativo parametro. Questa misura è effettuata attraverso la libreria Channel Monitor, come si vede dalla Fig3.5. In base a queste rilevazioni vengono impostati i parametri che rilevano il numero di misure effettuate (Nmeas) e il numero di volte in cui la corrente misurata a superato la corrente di soglia (Nth). Per la decisione, se un determinato step deve essere ripetuto oppure puó essere considerato superato si effettua la seguente operazione:

[2]Si misura la Imon proveniente dall'apparato

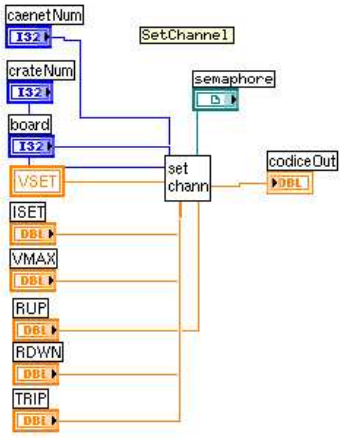

Figura 3.4: Libreria SetChannels, per l'impostazione del valore di tensione a cui l'alimentatore funzionerà.

$$X_{\%} = \frac{Nth}{Nmeas} 100$$

$$X_{\%} > \mathrm{Ith}_{\%}$$

Se si ha che:

$$X_{\%} > \mathrm{Ith}_{\%} \text{ si ripete lo step}$$

mentre se:

$$X_{\%} < \mathrm{Ith}_{\%} \text{ si procede con lo step successivo}$$

Nel caso in cui $X_{\%}$ risulti essere maggiore della $\mathrm{Ith}_{\%}$ lo step è considerato fallito, e viene rieseguito, per il massimo numero di volte che è stato deciso nella fase iniziale, altrimenti si procede con lo step successivo.

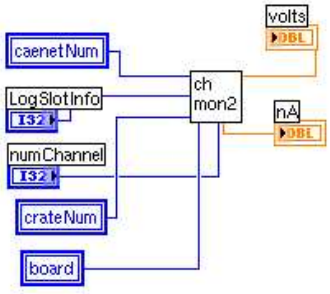

Figura 3.5: Libreria Channel Monitor.

In Fig.3.5 ne viene riportato il diagramma a blocchi che è usato nella procedura automatica per effettuare la misura della corrente per un canale.

Al termine di tutti gli step di tensione, oppure in caso di errore viene eseguito la parte dello schema a blocchi relativa alla libreria *SwitchChannels* con il flag impostato ad OFF per la board, dopodiché viene terminato il VI.

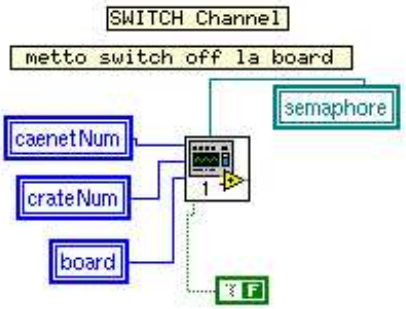

Figura 3.6: Libreria SwithChannels, per lo spegnimento della board.

Come si nota dalla Fig.3.6 viene inviato come parametro d'ingresso alla funzione il flag con valore *false*, questo fa si che venga eseguito lo spegnimento dei canali. Ovviamente se la board si trova per esempio a un valore di tensione pari a 5000 V, lo spegnimento avviene seguendo il parametro Ramp-Dow.

3.5 Grafici di monitoraggio della procedura di condizionamento

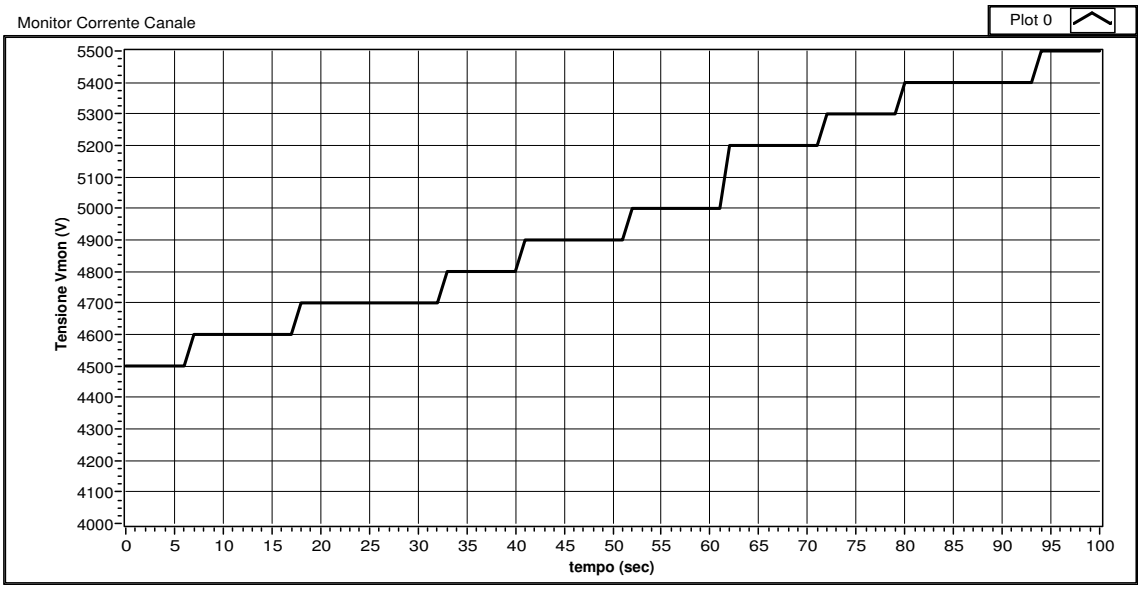

Figura 3.7: Monitoraggio della tensione per una board in funzione del tempo (V_{mon})

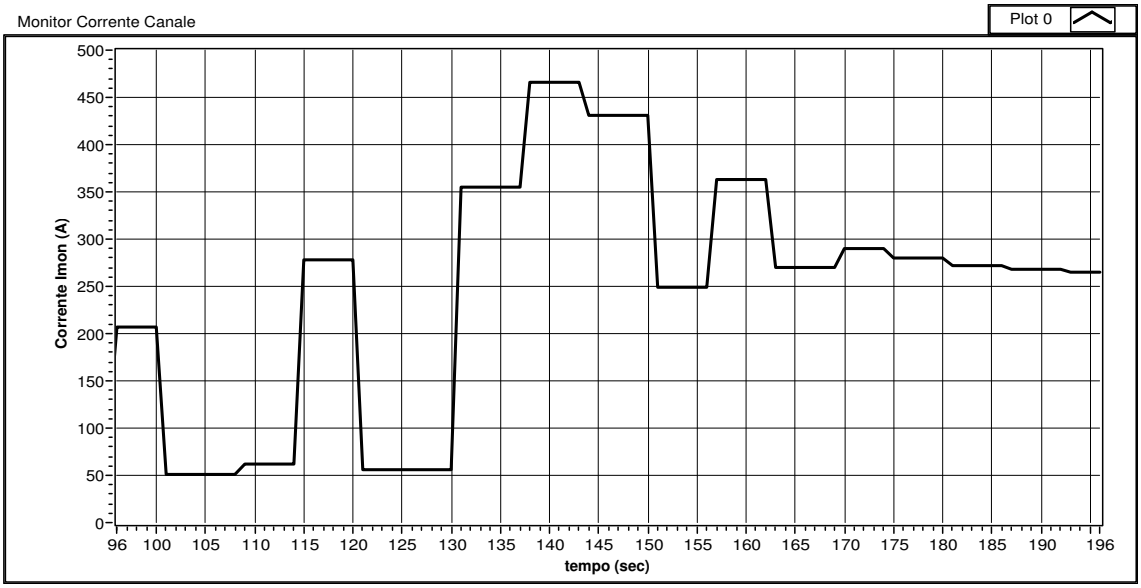

Figura 3.8: Monitoraggio della corrente per un canale in funzione del tempo (I_{mon})

Nel primo grafico viene riportato il valore della tensione per ogni step in funzione del tempo. Il valore assume in partenza 4500 V, fino ad arrivare ad una tensione di 5500 V. Nel grafico rappresentato in figura 3.7 si notano gli step di tensione con cui ogni board alimenta i tubi. Nel grafico viene riportato un esempio di condizionamento che un tubo LST può subire. Nella figura 3.7 per semplicità i tempi sono ridotti. Le unità di misura degli assi sono: nel asse x viene riportato il tempo, mentre nell'asse y viene riportato i valore di tensione in volt. Nel grafico in figura 3.8 viene monitorato l'assorbimento di corrente da parte di una un tubo LST al variare della tensione da un valore di 4500 V ad un valore di 5500 V. Si nota, che nel periodo iniziale l'assorbimento di corrente aumenta, per poi stabilizzarsi attorno ai 230-250 nA. Ogni volta che lo step ne aumenta la tensione, il canale in un primo istante assorbe un valore maggiore di corrente, per poi stabilizzarzi intorno ad un valore che dipenderà dal tubo alimentato. I grafici sono realizzati utilizzando la funzione *Waveform chart* presente in Labview. Per misurare i valori di tensione e di corrente, si utilizza la funzione Channel monitor realizzata, vedi Fig.3.5

Capitolo 4

Procedura automatica per la misura dei Plateau

La curva di plateau serve per:

1. se c'è il plateau è indice di buon funzionamento, e di una corretta miscela del gas. Quindi si riesce a determinare la tensione di lavoro stabile.
2. se non c'è, vi è un malfunzionamento.

4.1 Introduzione: Diagrammi di Plateau per gli LST

Un esempio dei diagrammi viene riportato nel seguente diagramma: Come si

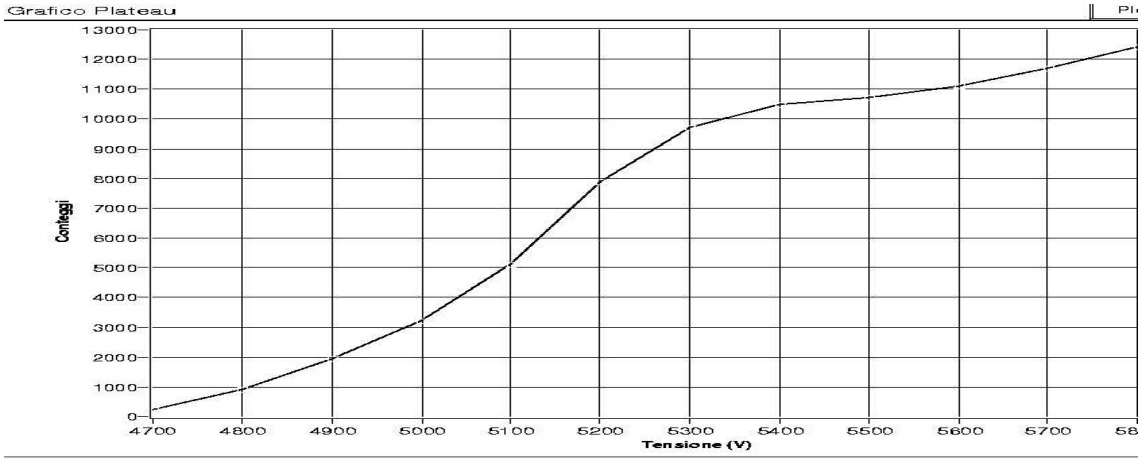

Figura 4.1: esempio diagramma di Plateau

può notare dalla figura 4.1 l'intervallo in cui si deve considerare un buon funzionamento della camera è compreso da 5300 V a 5600 V. In questo intervallo i

conteggi variano poco, al variare della tensione. Il rate dipende dalla geometria delle camere e da alcuni parametri quali miscela gassosa, densità del gas, tempo morto dell'elettronica. La lettura dei conteggi si effettua a tensioni che possono variare da 50 V a 100 V a partire da 4800 V fino a 5900 V. Riportando in un grafico il rate in funzione della tensione si nota una *zona di plateau*, nella quale il rate non cambia di molto nonostante l'aumento della tensione; le due estremità di tale zona vengono chiamate ginocchio destro e ginocchio sinistro.

Costruzione manuale dei diagrammi

Prima di realizzare la procedura automatica con Labview, i diagrammi venivano realizzati nel seguente modo:

1. Si imposta da console di comando dell'alimentatore la tensione da 4800 a 5900 V a intervalli di 100V;
2. Si fa eseguire il conteggio degli eventi ad un contatore NIM [1] per ogni step di tensione;
3. Si procede a realizzare il grafico tramite un qualsiasi foglio elettronico.

Per ovviare al problema di dover costruire manualmente i diagrammi si è realizzata una procedura automatica in Labview, utilizzando la scheda PCI di acquisizione dati della National Instrument DIO6533 per il conteggio degli eventi e la scheda caenet per il controllo dell'alta tensione.

4.1.1 Scheda PCI DIO 6533

Caratteristiche della scheda

La scheda DIO-6533 dispone di linee per l'input e l'output così suddivise:

1. linee di I/O utilizzate per la lettura e scrittura di dati digitali
2. linee di handishaking[2]

Le linee descritte al punto 1 sono utilizzate per leggere e scrivere dati digitali. Può esservi un uso singolo oppure in parallelo. Quando le letture/scritture sono

[1] CAEN mod N.145
[2] processo per la sincronizzazione delle linee di I/O

eseguite singolarmente, avvengono in modo indipendente. Quindi letture e scritture che sono su linee differenti non hanno nessun tipo di legame. Al contrario quando le linee vengono utilizzate in parallelo ogni lettura e scrittura viene eseguita in modo simultaneo su tutte le linee utilizzate. Ogni linea viene interpretata come un bit di una parola binaria, la quale può essere scritta oppure letta dalla scheda. Ogni linea determina il corrispondente peso dei bit.

La scheda DIO-6533 è provvista di una morsettiera per il collegamento con apparati esterni, e viene pilotata tramite le librerie NI-DAQmx.

La scheda dispone di un clock harware interno, può raggiungere una frequenza massima di 20MHz. Quando le linee vengono utilizate in parallelo si può sincronizzare la velocità di trasferimeto attraverso questo clock, il quale è impostabile via software. La scheda dispone di morsetti per il collegamento della massa e della tensione di alimentazione di 5 V. In Fig.A.1 viene riportata un immagine della scheda:

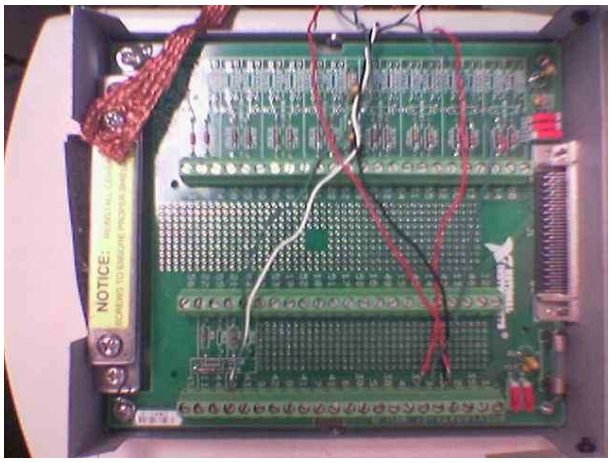

Figura 4.2: DIO 6533

Nella scheda sono presenti 32 linee di input/output. Sono suddivise in 4 blocchi, denominate porte. Ad ognuna viene assegnata una lettera: A, B, C, D. Le linee sono identificate da numero che varia da 0 a 7. Mentre le linee di handshaking sono 4 per ogni gruppo e sono:

- ACK1 e ACK2 (STARTTRIG1/STARTTRIG2)
- STOPTRIG1 e STOPTRIG2
- CLK1 e CLK2

- REQ1 e REQ2

ACK1 e ACK2, sono uscite digitali, che sono poste ad 1 durante il tempo in cui il gruppo di porte a loro associati eseguono operazioni di input/output. CLK1 e CLK2 servono per esportare/importare il clock. Possono essere usate anche per importare un clock esterno, che può fungere da sincronizzazione per le funzioni di lettura e scrittura. REQ1 e REQ2 si possono utilizzare da ingressi, per un segnale di trigger esterno, in modo da sincronizzare la lettura/scrittura. Le funzioni sopracitate sono utilizzabili tramite le librerie NI-DAQmx fonite con Labview. Di seguito verrà introdotto il funzionamento del contatore realizzato tramite Labview

4.2 Librerie DAQmx

Queste sono messe a disposizione per il controllo della scheda. Le librerie DAQmx, creano, accettano, a elaborano un task, ossia un insieme di informazioni per l'operazione che si deve svolgere. Come si vede dalla figura 4.3 i passi da eseguire per effettuare la lettura sono:

1. Creazione del task
2. Opzioni del task
3. Esecuzione del task
4. Stop del task
5. Eliminazione del task

Nel primo passo, ossia nella creazione del task, vengono definite le proprietà dei canali. Si decide infatti se il canale deve essere usate per l'input oppure per l'output. La creazione del task accetta come ingresso un parametro:
il canale della scheda. Ad esempio si passa come parametro: /Dev1/port0/line0:7.

- /Dev1 é la scheda DIO-6533
- port0 é la porta A
- line0 é la linea 0 della porta A

Nel secondo passo, ossia nelle opzioni del task, si imposta per esempio la modalità lettura in linea singola. Tutte le opzione scelte a questo punto sono applicate al canale scelto in fase di creazione del task. Nei passi successivi, si procede all'esecuzione, al reset e all'eliminazione del task. L'eliminazione rende libere le linee per successive modalità di funzionamento.

4.3 Uso della scheda per la lettura di dati a linee parallele

Per la realizzazione del contatore di un singolo canale, deve effettuare la lettura di una singola linea in modalità parallela tramite la scheda della NI DIO6533. Utilizzando solo il primo canale della scheda, il diagramma in Fig.4.3 effettua la lettura del segnale in ingresso alla linea utilizzata in sincronia con il clock impostato. La funzione di lettura restituisce quindi un vettore le cui componenti hanno i valori (0 o 1) assunti dal segnale in ingresso in corrispondenza di ogni fronte di salita (o discesa) del clock. La dimensione del vettore è impostata in base alle esigenze dell'acquisizione da effettuare. Nel paragrafo seguente verrà illustrato come questa modalità di acquisizione sia stata impostata per poter realizzare il contatore dei segnali provenienti dagli LST.

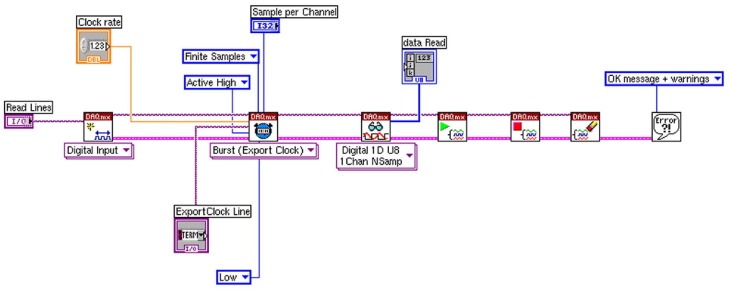

Figura 4.3: Diagramma di un applicazione Labview, con l'utilizzo delle librerie DAQmx

Il diagramma in Fig.4.3 permette la lettura del segnale in sincronia con il clock della scheda. Il segnale letto viene quindi restituito sottoforma di vettore il quale ha componenti corrispondenti a ogni ciclo di clock.

4.4 Realizzazione Contatore con Labview

Per la realizzazione del contatore con la funzione di lettura descritta nel paragrafo precedente, in primo luogo si devono analizzare le caratteristiche dei segnali discriminati degli LST e la durata dell'acquisizione in corrispondenza di ogni singola acquisizione.
Per la realizzazione dei plateau ogni singolo segnale discriminato deve avere una larghezza compresa tra 3 e 5 μs. Il conteggio deve essere effettuato su un treno di tali segnali in un determinato periodo di tempo. In media la frequenza dei segnali provenienti da un singolo LST è circa 200 Hz, questo implica che mediamente la distanza fra due segnali consecutivi è dell'ordine del ms.
Dalle precedenti considerazioni si evince che al fine di determinare da un vettore il numero di segnali in un periodo di tempo definito si deve campionare il treno di segnali con un clock con frequenza dell'ordine del MHz. Tale valore permette di individuare i fronti di ogni singolo segnale e da un buon margine per non confondere due segnali consecutivi che potrebbero verificarsi vicini. Chiaramente se due segnali distano meno di un microsecondo, questi vengono conteggiati come uno unico, ma tale situazione è poco probabile come si dimostra dai test ottenuti e descritti in seguito. In generale, al fine di avere una buona statistica di conteggi in corrispondenza di ogni singola tensione, l'acquisizione deve avere una durata di almeno 10 secondi. Una acquisizione di 10 secondi con un clock di 1 MHz implica dover utilizzare ed analizzare un vettore con 10^7 componenti. Test effettuati con una macchina con processore da 850 MHz con fedora core 3, hanno mostrato che per mantenere prestazioni efficienti si devono utilizzare vettori con al massimo 10^5 componenti.
Tutte le precedenti considerazioni portano a concludere che ogni singola acquisizione debba avere una durata massima di un decimo di secondo, tempo che è necessario sì per il conteggio di circa 10-20 segnali dagli LST, ma che non consente l'accumulo di una statistica significativa. Per aumentare la statistica si è quindi deciso di comporre ogni singola acquisizione di tante micro-acquisizioni da 10^{-1} secondi, in numero necessario ad ottenere un tempo voluto di acquisizione totale.
In corrispondenza di ogni tensione quindi l'applicazione eseguirà in successione un definito numero di micro-acquisizioni calcolato in base ai valori impostati di tempo di acquisizione totale, dimensione del vettore e frequenza di clock impostati, secondo le seguenti relazioni:

Il tempo di acquisizione è inserito dall'utente: T_{ACQ}

$$\Delta T_{ACQ} = n_{bit} \frac{1}{f_{clock}}$$

$$\mathrm{N}_{cicli} = \frac{T_{ACQ}}{\Delta T_{ACQ}}$$

Il ΔT_{ACQ} è il tempo in cui viene effettuata una acquisizione parziale, quindi per ricavare a livello di programmazione Labview il numero di cicli necessari ad effettuare l'acquisizione che viene impostata ad esempio di 10 secondi, si è calcolato N_{cicli}.

Alla fine di ogni micro-acquisizione il vettore viene elaborato e svuotato in modo da poter essere riutilizzato ad ogni ciclo successivo. Il contatore realizzato secondo le impostazioni descritte sopra, viste le precedenti considerazioni, dovrebbe essere ugualmente performante anche a frequenze del segnale più alte. La verifica dell'effettiva risposta in funzione della frequenza del segnale andrebbe ricercata con test specifici. Considerando il fatto che al momento tale applicazione deve essere utilizzata solo per questo tipo di rivelatori e che viene eseguita su una macchina datata, si preferisce attendere una nuova macchina, in fase di installazione, per l'esecuzione di test più approfonditi. Possiamo comunque attenderci che fino a frequenze del segnale di almeno un ordine di grandezza inferiore alla frequenza di clock il contatore dovrebbe essere attendibile entro un limitato errore. Al fine di verificare queste previsioni si è testato il contatore con segnali dell'ordine di 10^5 Hz ottenuti da un generatore di funzioni e si è ottenuta una differenza fra i conteggi acquisiti e i conteggi attesi dell'ordine del 5%. Il risultato ottenuto da questo test conferma la validità del sistema adottato per l'acquisizione dei conteggi.

4.5 Test del contatore Labview

Per verificare il corretto funzionamento del contatore labview si è svolto un primo test di confronto con un conteggio effettuato con un contatore NIM.
Al fine di verificare il conteggio ottenuto durante un'intera acquisizione si è sincronizzato il contatore NIM con le singole microacquisizioni, in modo da impostare automaticamente gli stessi tempi di conteggio per entrambi i contatori. Questa sincronizzazione si è realizzato usando come finestra temporale di conteggio del contatore NIM il segnale di ACk in uscita dalla scheda di acquisizione. Come precedentemente detto ACK assume il valore 1 esattamente durante il tempo di esecuzione della lettura, questo implica che il contatore NIM procederà ai conteggi solo durante la presenza di tale segnale. I risultati ottenuti sono riportati in Tab.4.1 e nei grafici in Fig.4.4 e in Fig.4.5

Tensione Step	Conteggi CAEN	Conteggi Labview	Differenza	Diff %
4800	320	320	0	0
4900	691	688	3	0,44
5000	1526	1516	10	0,65
5050	1807	1804	3	0,17
5100	2594	2588	6	0,23
5200	4397	4383	14	0,32
5300	6889	6867	22	0,32
5400	9525	9480	45	0,47
5500	12270	12212	58	0,47
5600	16111	16044	67	0,42
5700	39702	39602	100	0,25

Tabella 4.1: confronto conteggi contatore NIM e applicazione Labview

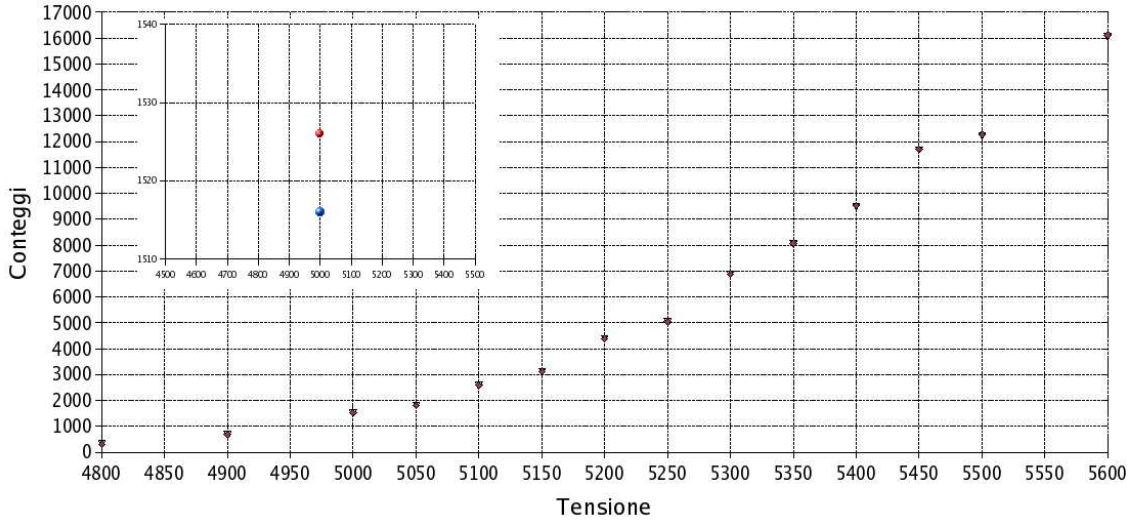

Figura 4.4: grafico differenza dei conteggi
o

In conclusione, dai test eseguiti si vede che la differenza di conteggi è minima e non influisce nella misura dei plateau rendendo il contatore labview ottimo per lo scopo.

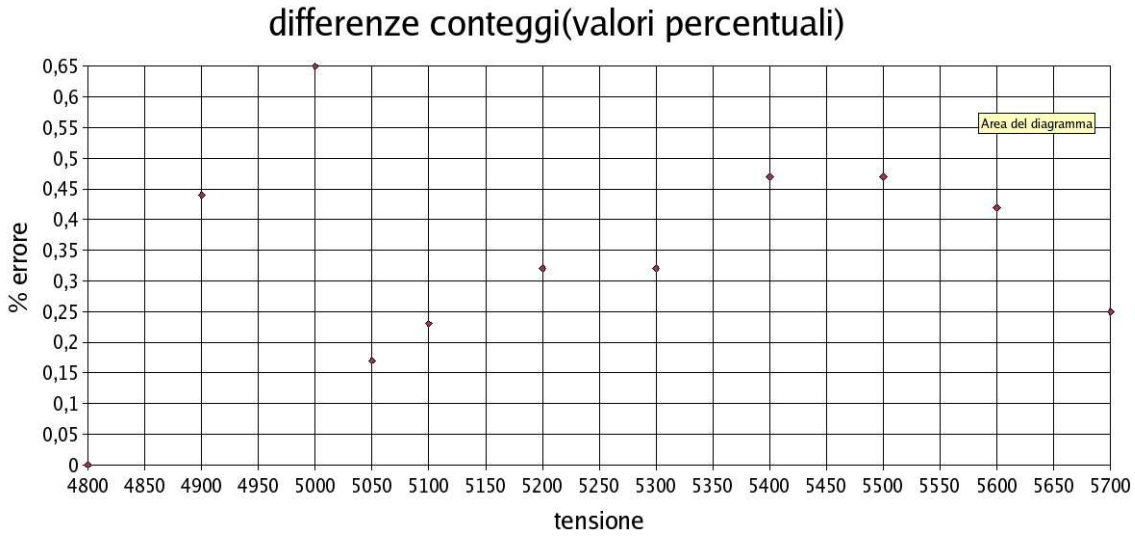

Figura 4.5: grafico differenza dei conteggi in percentuale

4.6 Realizzazione della procedura automatica per plateau con Labview

La realizzazione software per effettuare il grafico e riportare i conteggi effettuati in funzione della tensione, consta dei seguenti punti:

1. Creazione del vettore delle tensioni e dei conteggi.
2. Impostazione dello step di tensione.
3. Conteggio del rate per lo step corrente.
4. Ripeto i passi 2 e 3 per tutti i step di tensione impostati.
5. Viene costruito il diagramma tensione/conteggi.

Il pannello frontale dell'applicazione è riportato in Fig.4.6:
Nel pannello frontale si nota che le impostazione che si possono eseguire sono:

Passo 1 settaggio di tutti i parametri per la libreria setChannel descritta nei capitoli precedenti:

- caenetNum: 1
- board: 0
- crateNum: 1

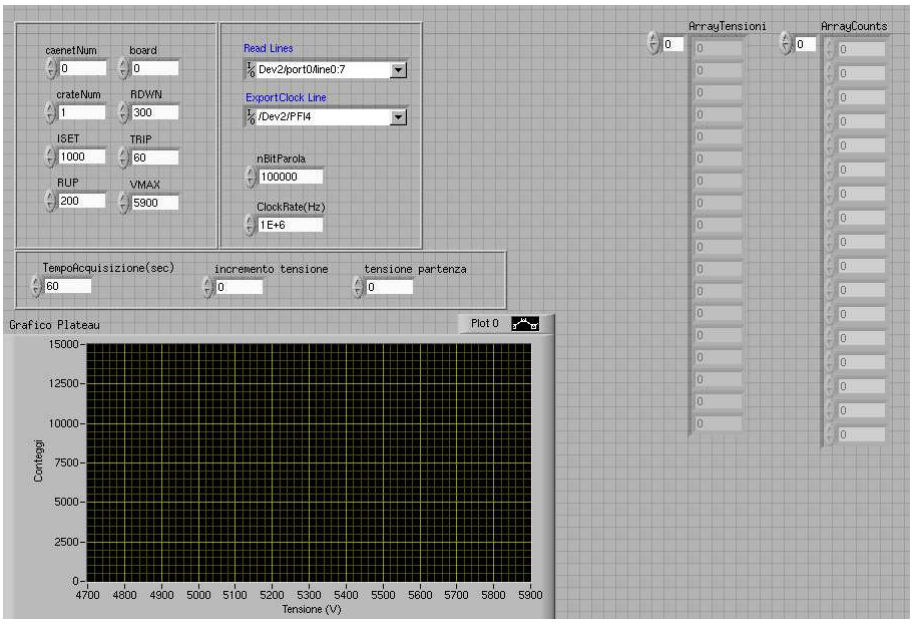

Figura 4.6: Pannello frontale per i Plateau

- Iset: 1000
- Rup: 200
- Rdwm: 500
- trip: 60
- Vmax: 5900

Passo 2 settaggio dei parametri relativi alle librerie DAQmx.

- ReadLines:/Dev2/port0/line0:7
- ExportClock: /Dev2/PFI4
- numero bit per parola: 10000
- Clock Rate: 1 MHz

Passo 3 settaggio delle opzioni di conteggio: tempo, tensione di partenza, incremento di tensione per ogni step.

- Tempo acquisizione: 60 s
- Tensione di partenza: 4700 V
- Incremento ti tensione per ogni step: 100 V

4.7 Parti principali del diagramma a blocchi.

Di seguito viene data una descrizione delle componenti principali utilizzate nell'applicazione realizzata tramite labview.

Calcolo step di tensione

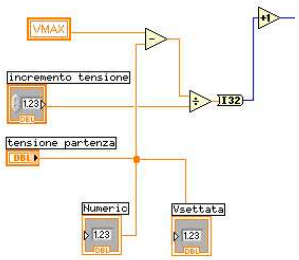

Figura 4.7: Calcolo degli step di tensione in base alla Vmax e all'incremento inserito

Nella Fig.4.7 è riportato la parte di schema a blocchi che si fa carico di determinare i vari step di tensione calcolandone il valore in base a Vmax, Incremento, Tensione di partenza. I valori di tensione calcolati sono memorizzati tramite vettori, rappresentati in Figura4.8.

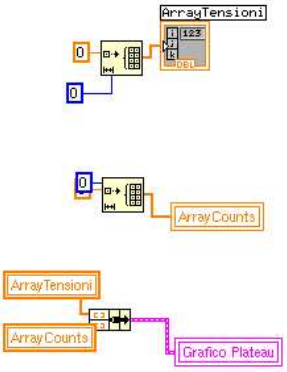

Figura 4.8: Vettori di memorizzazione valori tensione, conteggi.

Nella Fig.4.8 parte sottostante viene realizzato il grafico, utilizzando come parametri di ingresso i due vettori descritti precedentemente.

La Fig.4.9 mostra il settaggio della tensione da parte della libreria *Set Channels*, con un valore di tensione che dipende dallo step. Dopo aver impostato il valore di

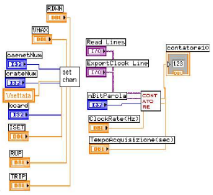

Figura 4.9: Impostazione tensione e Conteggio.

tensione corrente, si effettua il conteggio. Il valore contato viene immagazzinato in una variabile, la quale è usata per copiare il valore nel vettore dei conteggi.

4.8 Grafico differenze in percentuale dei conteggi Labview/Contatore

Viene riportato in fig.4.5 il grafico relativo alla differenza di conteggio che si sono riscontrate tra l'applicazione realizzata tramite Labview e il contatore NIM. Si vede che le differenze di conteggio che si sono avute utilizzando il contatore NIM e quello realizzato nella nostra applicazione sono molto piccole. Nella figura viene proposto in particolare la misura dei conteggi utilizzando i due metodi per un valore di tensione di 5000 V, risultando un conteggio di 1526 per il contatore hardware contro i 1516 della applicazione realizzata tramite Labview.

4.9 Esempio di diagrammi di plateau realizzati con Labview

I grafici in fig.4.10 e fig.4.11 sono state create utilizzando l'applicazione realizzata. Sono stati esportati direttamente da Labview. In conclusione la realizzazione risulta essere immediata e automatica. Questo risulta essere molto comodo nel caso di un test di molti tubi, creando un procedura semplice ed immediata per la misura dei plateau.
Nel grafico 4.10 si nota la zona di plateau della camera che è compresa tra i 5400 V e i 5700 V. Nel grafico 4.11 la zona di plateau risulta essere meno marcata.
In conclusione il metodo utilizzato risulta essere molto pratico per effettuare una misura immediata dei plateau di una camera. Questi tre esempi riportati sono stati utilizzati nel laboratorio per effettuare una misura dei plateau di tre camere presenti, e per valutarne il correntto funzionamento.

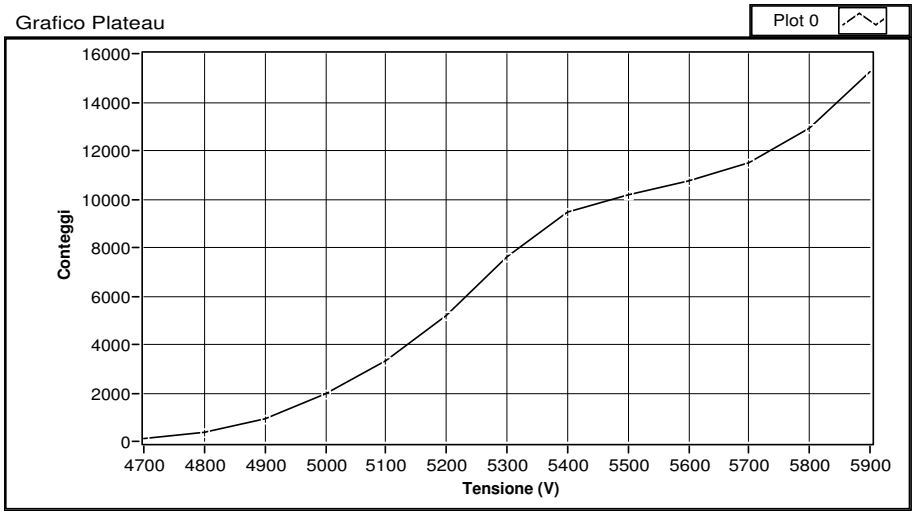

Figura 4.10: diagramma di plateau di una camera

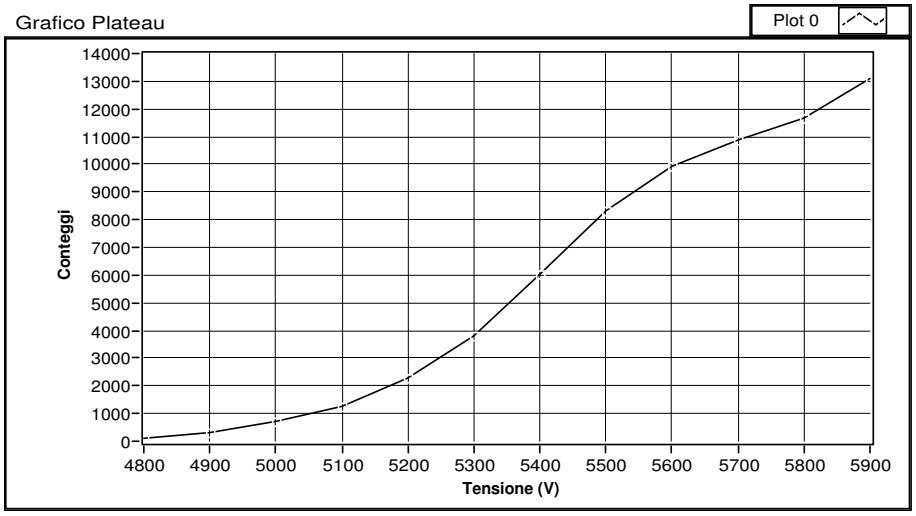

Figura 4.11: diagramma di plateau di una camera

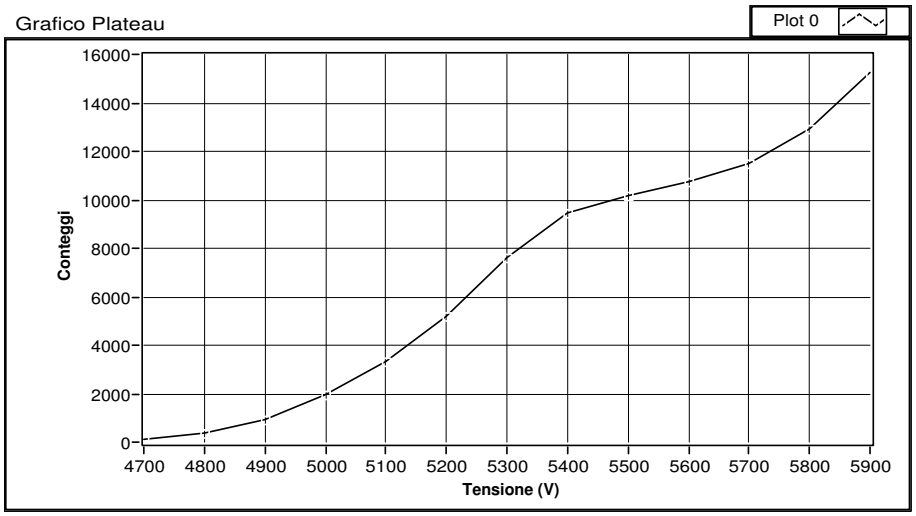

Figura 4.12: diagramma di plateau di una camera 5

Capitolo 5

Conclusioni

Il progetto realizzato in questa tesi è stato quello di riuscire a creare delle librerie atte a pilotare l'apparato ad alta tensione CAEN SY546 tramite Labview, rendendo il tutto molto semplice e veloce. La fase iniziale della tesi ha avuto come scopo lo studio delle librerie esistenti scritte in C, per comprenderne il loro funzionamento. Si è poi passato all'analisi di vari componenti di Labview da utilizzare per richiamare sorgenti scritti in C nel diagramma a blocchi. La scelta si posta sul modulo code inteface node (CIN). Questa funzione rende semplice, dopo aver opportunamente modificato i sorgenti, l'uso di librerie C. Di fatto è stato usato come modulo principale delle librerie create. Lo scopo principale di queste librerie è quello di riuscire a creare un modo rapido per costruire altre applicazioni di uso comune nel laboratorio che abbiano bisogno di utilizzare l'alimentatore ad alta tensione. Utilizzando le librerie sono state fatte due procedure automatiche. Le procedure realizzate sono:

1. Procedura automatica per eseguire il condizionamento dei tubi LST;

2. Procedura automatica per la misura dei plateau.

I risultati ottenuti hanno mostrato la loro utilità infatti, in caso di una riaccensione dei tubi LST dopo un lungo periodo di scarso utilizzo, la messa in funzione é resa più semplice dal fatto di non dover rifare le procedure di condizionamento e di misura dei plateau in modo manuale. Infatti queste applicazioni rendono molto semplice e affidabile il loro riutilizzo. Infatti viene lasciato alle procedure automatiche il compito di dare dei risultati atti a stabilire la condizione dei rilevatori. Sarà poi l'utente ad interpretare i risultati ottenuti per stabilire le condizioni dei rivelatori. Un altro aspetto interessante e la possibilità di utilizzare i programmi realizzati non solo in caso di una riaccensione dei tubi ma anche in ambito

produttivo, in vista di una produizione di massa, rendendo i test automatici e affidabili. Infatti le procedure di condizionamento sono procedimenti standard nella costruzione di rilevatori e possono anche essere impiegati per altre tipi di tecnologie.

Appendice A

I rivelatori di muoni LST

Per effettuare la rilevazione di particelle cariche, vengono utilizzati i rivelatori a gas LST[1], o tubi di Iarocci.

I tubi sono composti da un certo numero di celle, di forma quandrata o rettangolare. Le dimensioni tipiche sono: da 10x10 mm^2 a 100x100 mm^2. Le pareti interne delle celle sono rivestite di graffite, ed al centro di ogni cella è presente un filo conduttore lungo con il tubo. Quanto appena descritto è contenuto in un contenitore di PVC il quale permette la circolazione della miscela di gas all'interno delle celle. I tubi che sono utilizzati nel laboratorio di criogenia di Ferrara hanno 8 celle con dimensioni di 17X15 mm^2 hanno una lunghezza di 350 cm. Il filo presente all'interno delle celle ha la funzione di anodo, mantenuto ad una tensione di circa 5600 V. La graffite che è presente sotto forma di rivestimento dei tubi, ha invece la funzine di catodo, è collegata a massa. La miscela presente all'interno dei tubi è cosi formata: Argon - Isobutano - Anidride Carbonica (3%/9%/88%). Quando una particella carica passa attraverso il tubo, al suo interno si verifica la ionizzazione del gas. Per effetto del campo elettrico tra anodo e catodo si ha una migrazione di ioni negativi verso l'anodo e di ioni positivi verso il catodo. Con l'ausilio del gas presente si ha una moltiplicazione della carica raccolta su anodo e catodo. La miscela inoltre fa in modo che la ionizzazione avvenga in un area ristretta al punto di passaggio della particella. Gli impulsi elettrici dovuti agli ioni permettono di rilevare il punto di impatto della particella.

[1] Limited Streamer Tube

A.1 Caratteristiche dei rivelatori LST

- Fili 100 μm
- Lunghezza Streamer circa 2-3 mm
- Tubi Sezione quadrata: L = 5-10 m s = 9-30 mm
- Impulsi di circa 50 - 100 mV
- Alta Tensione: 4 - 6 kV
- Geometria semplice: profili a pettine (di PVC) ricoperti di grafite

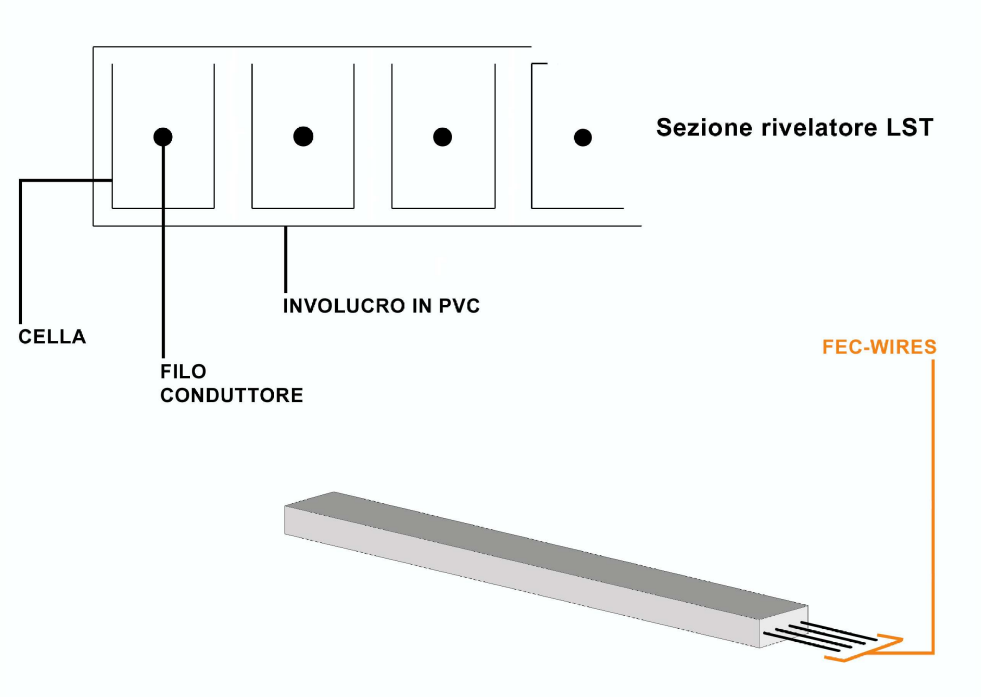

Figura A.1: Sezione e prospettiva di un tubo LST

Bibliografia

[1] Kernighan Brian W., Ritchie Dennis M.
Principi di programmazione e Manuale di riferimento
II Edizione, Prentice Hall

[2] http://www.fe.infn.it/electron/babar_ifr.htm

[3] Manuale della scheda DIO-6533 - National Instruments

[4] Manuale delle librerie DAQmx - National Instruments

[5] Manuale alimentarore CAEN SY546 - C.A.E.N.

[6] Manuale scheda PCI CAENET A1303 - C.A.E.N.

[7] http://www.caen.it/nuclear - sito ufficiale C.A.E.N. S.p.A.

[8] Tesi di Laurea di Mirco Tagliani
Realizzazione di un sistema di acquisizione dati per un rivelatore di muoni
http://df.unife.it/u/mandreot/Tesi/LaureaInfo/TesiLaureaMircoTagliani.pdf.gz

Ringraziamenti

Desidero fare un ringraziamento particolare al Dott. Mirco Andreotti per avermi sopportato tutti questi mesi, un grazie sentito per tutte le sue delucidazioni, per la sua disponibilità e chiarezza. Grazie mille per avermi dato la possibilità di fare la tesi in questo laboratorio. È stata una bella esperienza ed ho imparato moltissime cose. Un ringraziamento al Dott. Gianluigi Cibinetto per il tempo dedicatomi e per i consigli sulla stesura della tesi. Voglio inoltre fare un salutone a tutti i miei compagni di informatica, che mi hanno aiutato in questi tre anni a superare esami ed a passare momenti davvero speciali che non dimenticherò mai. Un grazie particolare va ai miei amici/compagni di studio: Alessandro, Riccardo, Lucio, Michele, Francesco, Diego. Un saluto al mio amicone Claudio, che in ogni occasione non si tira mai indietro per aiutarmi e darmi consigli, è stato grazie a lui se ho ricominciato la laurea; grazie mille. Un saluto davvero particolare va al mio amico Riccardo, che mi ha aiutato in mille occasioni. Grazie per tutte le piade e pizze che hai mangiato con me e alle mille ore di studio che abbiamo fatto assieme. Un saluto va ai miei compagni del laboratorio (stanza 218) del dipartimento di fisica, ossia Mirco(chitarrista), grazie per tutti i consigli su Labview, grafica e molto altro. Laura, per tutte le ore passate in compagnia e per tutte le piccole lezioni di fisica che mi ha dato(ora ho capito bene tutte le famiglie delle particelle) e, quasi dimenticavo, per tutti i plateau fatti. Valentina, mi ha fatto ripassare C, a stampare usando il formato PS per usare la stampante di rete del secondo piano(che ogni tanto ci fa dannare). Devo dire che ho passato bei momenti in laboratorio, tutti sono stati molto disponibili e cordiali. Siete tutti fantastici, vi auguro un in bocca al lupo per il futuro. Grazie di cuore a tutti. Nella speranza di aver lasciato un sorriso a tutti voi, colgo questo spazio come un modo, anche se piccolo, per ringraziare tutti. Infine una cosa che voglio dire a tutti: sorridete che fa bene(zio cannuccia). Un ringraziamento particolare a tutta la mia famiglia.

Ruggero