Alma Mater Studiorum · Università di Bologna

SCUOLA DI INGEGNERIA E ARCHITETTURA

Corso di Laurea in Ingegneria Informatica

Implementazione di un'interfaccia utente per il controllo di una fotocamera intensificata in ambiente LabVIEW

CANDIDATA

Elisabetta Corain

RELATORE

Chiar.mo Prof. Gabriele Neretti

CORRELATORE

Dott. Matteo Taglioli

Sessione III

Anno Accademico 2015/2016

Alla mia famiglia.

Introduzione

Nell'ambito dello studio dei plasmi generati tramite scariche elettriche si rivela necessario catturare gli spettri generati da suddette interazioni per studiarne diverse proprietà quali composizione e temperatura. Per effettuare tali misurazioni sono necessari due strumenti: un monocromatore che scomponga e rifletta la fonte di luce in input e una fotocamera che catturi l'immagine dello spettro.
Risulta quindi necessaria un'interfaccia utente in grado di controllare simultaneamente i due strumenti in modo da facilitare il percorso di acquisizione e manipolazione degli spettri. Lo scopo della tesi è implementare l'interfaccia richiesta attraverso il linguaggio di programmazione grafica LabVIEW, con l'ausilio delle API dei due strumenti.
È presente un progetto preesistente che realizza l'interfaccia di controllo del monocromatore e di una fotocamera non intensificata. A partire da questo progetto se ne è realizzato uno nuovo che riutilizza alcune parti del precedente, considerando lo stesso monocromatore e una nuova fotocamera intensificata.
A partire quindi dal programma esistente si è effettuato uno studio dello stesso per poterlo così adattare alla nuova fotocamera. Dopo un primo tentativo di riutilizzo e refactoring si è preferito realizzare un programma ex novo, modificando anche alcuni elementi dell'interfaccia grafica. Chiaramente si sono mantenuti alcuni elementi, in particolare quelli legati al monocromatore, dal momento che lo strumento è rimasto lo stesso, così come le librerie ad esso associate.

Indice

Elenco delle figure

Capitolo 1

Documento dei requisiti

Si descrive nelle seguenti sezioni la specifica dei requisiti del sistema software che si intende sviluppare.

1.1 Scopo

Come accennato nell'introduzione si rende necessaria la realizzazione di una nuova interfaccia che possa controllare simultaneamente il monocromatore HR460 e la fotocamera intensificata 4 Picos. Il precedente programma aveva la capacità di controllare lo stesso monocromatore ma una diversa fotocamera, SensiCam (version 3.0).

1.2 Descrizione generale

L'utente, attraverso il programma da progettare, deve poter controllare sia il monocromatore che la fotocamera intensificata. È importante che questi due strumenti possano essere controllati simultaneamente poiché le loro funzioni sono strettamente legate: il monocromatore genera un determinato spettro attraverso la sua struttura interna e la fotocamera deve poter catturare tale spettro il più precisamente possibile.

1.2.1 Funzioni del prodotto

Attraverso il programma deve essere possibile inizializzare le principali caratteristiche del monocromatore in base alle esigenze dell'utente: grating, lunghezza d'onda e fenditura di entrata. Deve anche essere possibile settare e modificare le impostazioni della fotocamera, compatibilmente con le funzioni messe a disposizione tramite le API della stessa.

1.2.2 Caratteristiche utente

Il progetto da realizzare viene utilizzato da professori e dottorandi per la caratterizzazione spettroscopica di plasmi a bassa pressione e atmosferici. È necessario trovare un punto di equilibrio tra le esigenze dell'utente e i vincoli di programmazione.

1.3 Requisiti specifici

1.3.1 Requisiti dell'interfaccia utente

L'interfaccia deve risultare il più intuitiva possibile, per questo si deve cercare di mantenere un minimo di omogeneità con il programma passato. Devono infatti coesistere nella stessa finestra sia i controlli per il monocromatore che quelli per la fotocamera.
Vediamo quali sono i requisiti specifici per i due strumenti.

Monocromatore Per quanto riguarda la parte di interfaccia che permette di impostare correttamente il monocromatore è necessario poter modificare il grating (1200 o 2400) e la calibrazione di correzione della lunghezza d'onda. Si deve anche poter modificare la slit di entrata e la lunghezza d'onda [4];

Fotocamera La parte di interfaccia che concerne la fotocamera intensificata deve contenere i seguenti parametri[8] [7]:

- MCP Gain Voltage: guadagno di tensione in volt tra $600V$ e $950V$
- Exposure Time: tempo di esposizione, deve essere positivo e si deve poter specificare l'unità di misura
- Delay: ritardo in secondi dell'acquisizione dell'immagine
- Frames to accumulate: numero di immagini da acquisire
- Trigger Source: -Trig per far scattare l'otturatore esternamente attraverso l'input di trigger negativo collegato elettricamente ad un segnale, Fsync per far scattare l'otturatore internamente dal segnale Fsync
- Gate Control: interno, l'input di controllo dell'otturatore è connesso internamente all'output IntGtP che può essere usato per scopi di controllo o trigger, o esterno, l'input di controllo dell'otturatore è collegato al connettore in input di ExtGtP
- Start Option: Cold Start, non ci sono parametri pregressi da utilizzare, Warm Start, dopo il riavvio della fotocamera si chiede all'utente se si devono caricare i parametri usati precedentemente, Auto Warm Start, dopo il riavvio della fotocamera vengono caricati i parametri usati in precedenza senza la conferma da parte dell'utente
- Detector: per specificare l'area attiva del chip CCD, frame intero, binning 2x2 oppure ROI (Region of Interest, solo 1/3 dell'area complessiva)
- Digitalizer Mode: acquisizione dell'immagine con una precisione di 8 o 14 bit per pixel
- CCD Gain: automatico oppure settabile manualmente (in dB)
- Trigger Mode: trigger diretto (ad ogni segnale di trigger farà scattare l'otturatore) oppure singolo (in questo caso deve essere possibile inserire il numero di trigger per frame)

- Modalità di acquisizione dell'immagine: Single Exposure per catturare una sola immagine, Live Mode per acquisire immagini in modo continuo;

Devono anche essere presenti dei comandi per il post-processing e l'analisi delle immagini acquisite. Come nel progetto preesistente è necessario visualizzare lo spettro in un grafico che abbia nell'asse orizzontale il range di lunghezza d'onda osservato e nell'asse verticale l'intensità relativa dello spettro. Inoltre deve essere possibile sottrarre un background acquisito in precedenza ed effettuare dei calcoli sullo spettro, come integrale sulla lunghezza d'onda, FWHM, picco massimo e numero di picchi. Alla fine delle operazioni, qualora ce ne sia la possibilità, si deve poter salvare i dati in un file di testo che tenga traccia dei valori calcolati per visualizzare lo spettro.

1.3.2 Linguaggio di programmazione

Il linguaggio di programmazione da utilizzare per realizzare il progetto richiesto è LabVIEW.
LabVIEW è un linguaggio di programmazione grafica (G - Graphical Programming Language) che utilizza un modello a flusso di dati invece di linee sequenziali di codice testuale, permettendo di scrivere codice funzionale utilizzando un layout grafico che assomiglia a un diagramma di flusso [5].
L'ambiente di programmazione LabVIEW presenta due principali finestre di lavoro: front panel e block diagram.

Front Panel Finestra in cui si visualizza e si modifica l'interfaccia utente. Sono disponibili diversi stili e comandi per l'interazione con l'utente. Si possono inserire bottoni, caselle di inserimento di stringhe o numeri, led, controlli a tab, indicatori per visualizzare grafici, immagini, matrici ecc.

Block Diagram Finestra in cui si compone il codice a blocchi con controlli, indicatori, subVI e molte alte componenti. Da tale finestra è possibile effettuare il debug del codice e controllare il contenuto delle variabili.

Le strutture principali utilizzate nel codice sono cicli, while e for, e case structure, che funzionano come if o switch. Per mantenere il valore delle variabili e condividerle con altre strutture si usano fili che le interconnettono. Elementi fondamentali nel linguaggio di programmazione grafica sono controlli e indicatori: i controlli permettono all'utente di inserire o modificare i dati, mentre gli indicatori consentono di visualizzare lo stato delle variabili.

Capitolo 2

Analisi del progetto preesistente

2.1 Strumenti coinvolti

Si illustrano di seguito alcune principali caratteristiche degli strumenti coinvolti nel progetto già esistente.

2.1.1 Monocromatore

Un monocromatore è un dispositivo che scompone un singolo fascio di luce policromatica in più fasci di luce monocromatica (che contiene cioè onde di una sola frequenza), permettendo così di analizzare l'intensità in funzione della lunghezza d'onda.
Nello strumento la luce policromatica entra da una fessura; tramite un sistema ottico viene inviata su un reticolo di diffrazione o ad un prisma che scompone il fascio. Una seconda fenditura raccoglie poi il fascio di una determinata lunghezza d'onda.
In questo progetto si considera un monocromatore Jobin-Yvon HR460 con configurazione *Czerny-Turner* in figura 2.1. In questa configurazione la sorgente di illuminazione si rivolge a una fenditura di ingresso; la quantità di energia luminosa disponibile dipende dall'intensità della sorgente nello spazio

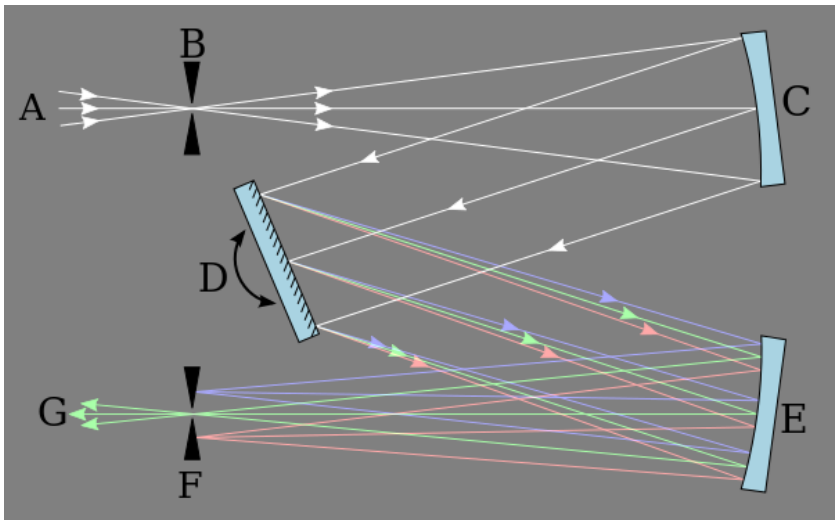

Figura 2.1: Schema della struttura interna di un monocromatore con configurazione *Czerny-Turner*

definito dalla fenditura e l'angolo di accettazione del sistema ottico. La fenditura è posta al centro di uno specchio curvo in modo che la luce proveniente dalla fenditura riflessa dallo specchio venga collimata (focalizzata all'infinito). La luce collimata viene poi diffratta dal grating e quindi raccolta da un altro specchio che riorienta la luce dispersa in una fenditura di uscita [1].

2.1.2 Fotocamera

Una fotocamera intensificata è una fotocamera che al posto della pellicola fotosensibile utilizza un sensore (CCD) in grado di catturare l'immagine e trasformarla in un segnale elettrico di tipo analogico. Gli impulsi elettrici vengono convertiti in digitale da un convertitore analogico/digitale in un chip di elaborazione esterno al sensore. Viene quindi generato un flusso di dati digitali atti ad essere immagazzinati in vari formati su supporto di memoria. Il CCD (Charged-Coupled Device) è un dispositivo attraverso il quale si ottiene un segnale elettrico in uscita, in seguito a una sequenza temporizzata di impulsi, grazie al quale è possibile ricostruire la matrice di pixel che compongono l'immagine proiettata sulla superficie del CCD stesso. Questa

informazione può essere usata come segnale analogico, e quindi essere usata per riprodurre l'immagine su un monitor, oppure può essere convertita in formato digitale.

In aggiunta a queste tecnologie una fotocamera intensificata, in figura 2.2,

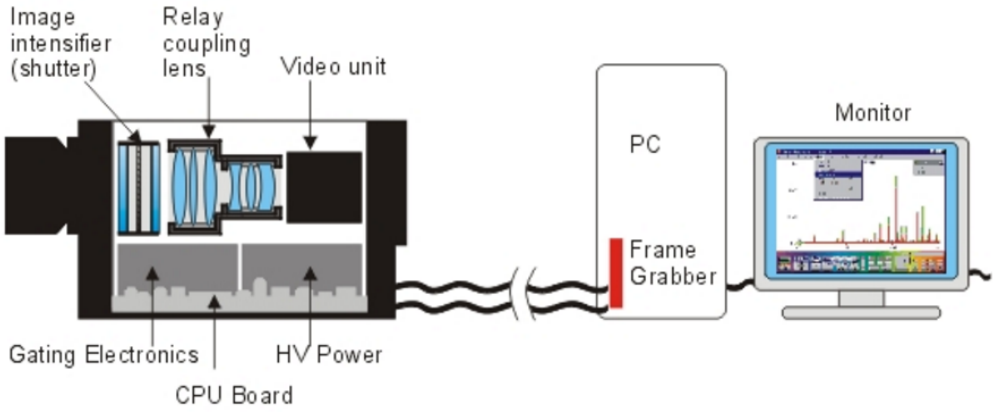

Figura 2.2: Struttura interna di una fotocamera ICCD

presenta un intensificatore di immagine prima del CCD: la luce in ingresso viene prima amplificata dall'intensificatore e poi l'immagine intensificata viene proiettata sul sensore CCD mediante delle lenti accoppiate. In questo modo si riescono a catturare immagini anche in presenza di poca luce, o utilizzando tempi di esposizione estremamente ridotti (anche di centinaia di nanosecondi) [6].

2.2 Principali funzioni

Il progetto preesistente ha le stesse funzioni richieste nel nuovo programma, con alcune eccezioni per quanto riguarda la fotocamera. Quella utilizzata precedentemente presentava alcune funzioni che la nuova fotocamera non mette a disposizione. Ad ogni modo si descriverà il progetto cercando di evidenziare le funzionalità che necessitano di essere abbandonate a causa della diversità dei due strumenti.

L'interfaccia non presenta alcuna divisione tra i controlli dei due strumenti

e si possono quindi controllare simultaneamente senza alcuna distinzione. È chiaro che lo strumento che ha priorità in fase di inizializzazione è il monocromatore, in quanto composto da elementi fisici che devono essere riportati con movimenti meccanici ad una posizione di default; solo una volta che gli specchi e il grating all'interno dello strumento si trovano nella posizione iniziale è possibile cominciare ad utilizzare il programma.

2.2.1 Monocromatore

I comandi presenti per controllare il monocromatore sono gli stessi richiesti nel nuovo progetto: selezione del grating (e rispettive calibrazioni per correggere eventuali errori), impostazione della lunghezza d'onda e slit di entrata. Sono inoltre presenti dei led che mostrano eventuali errori: un errore generico in fase di inizializzazione, lunghezza d'onda inserita errata e superamento dei limiti dei valori consentiti.

2.2.2 Fotocamera non intensificata

Per quanto riguarda la fotocamera si distingue fra acquisizione singola o continua ed è possibile inserire tempo di esposizione e ritardo di acquisizione dell'immagine. Si può inoltre scegliere un valore specifico di binning, fino a 1/16 per il binning verticale e fino a 1/8 per il binning orizzontale. Viene anche segnalato un'eventuale caso di saturazione con un calcolo successivo all'acquisizione foto.

2.2.3 Altre opzioni

È possibile effettuare delle operazioni non collegate direttamente ai due strumenti. Oltre a visualizzare l'immagine così come è stata presa, è anche possibile visualizzare lo spettro corrispondente in seguito a determinati calcoli. Su tale spettro si possono richiedere alcune informazioni quali l'integrale su un certo intervallo della lunghezza d'onda, il numero di picchi rilevati al di sopra di una certa soglia; viene inoltre mostrato, senza bisogno

di chiederlo, il valore del picco più alto alla lunghezza d'onda corrispondente e la FWHM (Full Width at Half Maximum), che corrisponde alla differenza fra i valori assunti dalla variabile indipendente lunghezza d'onda quando la variabile dipendente intensità dello spettro è pari a metà del suo valore massimo. È disponibile un comando per l'acquisizione del background (legato però alla fotocamera): infatti nel caso in cui lo spettro venga generato in un ambiente (visivamente) "rumoroso" si può successivamente utilizzare il suddetto dato per sottrarlo a immagini successive, in modo da renderle il più chiare possibile.

2.3 Codice del progetto preesistente

Si analizza nelle seguenti sezioni il codice del progetto preesistente, al fine di giustificare la scelta di progettazione fatta successivamente. La descrizione del codice si articola fra analisi del front panel e del block diagram, che sono i due ambienti di programmazione principale del linguaggio LabVIEW (sottosezione 1.3.2).

2.3.1 Front Panel

Il front panel in figura 2.3 è diviso in blocchi ideali che non dividono veramente il codice ma danno all'utente tale impressione.
In alto si trova una tab control che mostra, a seconda della selezione, l'immagine acquisita oppure lo spettro; per visualizzare lo spettro dopo aver ottenuto l'immagine e viceversa è necessario riacquisire i dati. Nella sezione dedicata allo spettro sono presenti tre indicatori che mostrano il valore del picco massimo alla lunghezza d'onda corrispondente e il valore della FWHM in nm.
La prima fascia di pannelli è composta da tre parti. La prima contiene un bottone booleano con cui si richiede il numero di picchi oltre una certa soglia, i cui valori vengono visualizzati in indicatori di array. La seconda permette di inserire gli estremi di un intervallo di lunghezza d'onda e di richiederne,

attraverso un bottone booleano, l'integrale. La terza parte non è delimitata da un vero e proprio pannello ma contiene due indicatori di versione (main e boot) del monocromatore, un indicatore della acquisizioni fatte fino a quel momento e il bottone booleano di uscita.
La seconda fascia è divisa in due parti. La prima, dedicata alla fotocamera, contiene un bottone booleano per acquisizione singola e uno per quella continua, due controlli e rispettivi indicatori per ritardo ed esposizione, due quadranti per impostare il binning verticale e orizzontale, un led di indicazione di un'eventuale saturazione e un controllo per inserire il numero di pixel su cui fare la media per generare lo spettro. La seconda parte contiene invece controlli e indicatori per il monocromatore: bottone booleano per il grating (1200 o 2400) e rispettivi controlli per inserire la calibrazione in entrambi i casi, un controllo per inserire la lunghezza d'onda e uno per la slit di entrata, due indicatori per mostrarne i valori reali, e tre led che indicano se il monocromatore è stato correttamente inizializzato o se sono presenti errori generici, lunghezza d'onda non valida e superamento dei limiti consentiti.
La terza fascia è divisa in quattro parti. Nella prima c'è la possibilità di salvare i dati con un file path predefinito (calcolato secondo le impostazioni del programma) oppure da inserire manualmente. Nella seconda parte è possibile selezionare il modo di acquisizione della foto, via software (impostazione di dafault), trigger esterno al fronte di salita o trigger esterno al fronte di discesa; con un bottone booleano si modifica la modalità e un led indica se il programma è in attesa di un trigger. La quarte parte indica la temperatura del CCD e della scheda PCI del calcolatore.

2.3.2 Block Diagram

Il block diagram in figura 2.4 è composto di una macro flat sequence (sequenza che forza il flusso dei dati in una determinata sequenza scelta dal programmatore) divisa in tre blocchi: il primo che delimita l'inizializzazione degli strumenti, il secondo contiene il codice principale del programma, e il terzo chiude in modo corretto la fotocamera.

Primo blocco

Questo blocco, in figura 2.5, è dedicato principalmente all'inizializzazione del monocromatore. È presente un'ulteriore flat sequence di tre blocchi. Nel primo viene invocato il subVI *StartUp.vi* che inizializza il monocromatore e riporta i componenti interni alle posizioni di default. Questo subVI restituisce in uscita una lista di errori e le informazioni riguardo il numero di versione del monocromatore. Nel secondo blocco è presente un timer che mette in pausa il programma per 1 secondo, per permettergli di ricevere la risposta del monocromatore, e nel terzo blocco il led di inizializzazione viene commutato a vero per indicare l'avvenuta inizializzazione dello strumento.
Oltre al subVI del monocromatore ne vengono chiamati altri due che riguardano la fotocamera: *SCSetBoard.vi* e *SCSetMode.vi* servono per inizializzare la scheda della fotocamera e danno in uscita un cluster che contiene eventuali errori.
Al di fuori di questa flat sequence è presente del codice per caricare un file che contiene dati utilizzati successivamente per compiere alcuni calcoli.

Secondo blocco

Questo blocco, in figura 2.6, è composto da un ciclo con al suo interno un'altra flat sequence. Il ciclo while è necessario per rendere il programma sensibile in qualsiasi momento al cambiamento di qualche impostazione (switch di un bottone booleano o del valore di un controllo); viene bloccato solo quando viene invocato lo stop, con l'apposito bottone. La flat sequence all'interno del ciclo è divisa in cinque blocchi che si descrivono qui di seguito.

- Primo blocco, figura 2.7. Contiene un ulteriore ciclo while. All'interno di questo ciclo troviamo alcune case structure che vengono eseguite solo nel caso in cui il controllo al quale sono associate assuma un certo valore. La case structure associata al bottone booleano *Save* permette di salvare i dati qualora questo venga premuto; il nome con cui viene salvato il file viene costruito dalla funzione di concatenazione di strin-

ghe che ha in ingresso le impostazioni caratteristiche del programma nel momento in cui si decide di salvare. La case structure associata al bottone booleano *Waveform Integral* calcola, se richiesto, l'integrale della lunghezza d'onda su un intervallo specificato. È presente una case structure associata a un'ulteriore bottone booleano di *Save* che però al suo interno manca della funzione di concatenazione, il path del file viene infatti inserito manualmente dall'utente in un apposito controllo. Nella case structure associata al controllo booleano *Load Coeff* è presente una funzione per caricare un file con certi valori che possono essere usati successivamente nel programma. La case structure associata al controllo *Peak Detector* calcola, attraverso il subVI *Waveform Peak Detection.vi* il numero di picchi sopra una certa soglia con relativa lunghezza d'onda. Un'altra case structure associata al controllo booleano *Grating* discrimina in base al valore del grating il valore da dare in ingresso al monocromatore. È presente infatti un'altra flat sequence divisa in quattro blocchi: nel primo viene invocato il subVI *Port & Grating.vi* per impostare il grating e il tipo di input del monocromatore, nel secondo attraverso il subVI *Spectral GOTO.vi* si richiede al monocromatore di portare i componenti interni alle posizioni inserite dell'utente, nel terzo con il subVI *Slits.vi* si imposta la slit di entrata, nel quarto il subVI *Spectral Position.vi* restituisce la lunghezza d'onda effettiva a cui si trova il monocromatore.

- Secondo blocco, figura 2.8. Completamente dedicato alla fotocamera. Dapprima vengono invocati i subVI *SCStopCoC.vi* e *SCGetStatus.vi* che restituiscono le informazioni di temperatura e di due componenti della fotocamera. In seguito alla costruzione di un cluster contente le impostazioni della fotocamera vengono invocati i subVI *New Delay and exposure formatter.vi* per la formattazione del ritardo e del tempo di esposizione e *SCSetNewCoC.vi* per inviare le impostazioni alla fotocamera. L'ultimo subVI chiamato è *SCRun.vi* che, come suggerisce il nome stesso, fa partire la fotocamera.

- Terzo blocco, in figura 2.9. Anche questo riguarda solo la fotocamera e contiene tre subVI. Il primo *SCGetImageStatus.vi* è wrappato all'interno di un ciclo while ed è il subVI che permette l'acquisizione vera e propria dell'immagine. I dati acquisiti con questo subVI vengono poi passati al subVI successivo, *SCGetImageSize.vi*, che restituisce le dimensioni in pixel dell'immagine. I dati che escono da questo subVI sono date in ingresso al successivo, *SCGet12BitImage*, che restituisce l'immagine. Infine attraverso una case structure viene commutato il valore del led *Saturation* in base al valore dei pixel risultati dall'ultimo subVI.

- Quarto blocco, figura 2.10. Contiene una case structure associata al bottone booleano *Grating* che calcola i valori da inserire nel grafico che mostra lo spettro. Tali calcoli vengono effettuati a partire dalla dimensione della foto (che varia se è stato utilizzato un particolare binning) e dal valore di ogni pixel; per prima cosa vengono calcolati i valori delle lunghezze d'onda da visualizzare sull'asse orizzontale del grafico, in modo speculare rispetto al centro dell'immagine, e in seguito viene calcolata una media di un certo numero di pixel (dati dal controllo *Pixel to Average*) per i valori da visualizzare nell'asse verticale del grafico. I calcoli effettuati in questo blocco per trovare i valori della lunghezza dipendono dalla conformazione del monocromatore e dalla larghezza dell'immagine della fotocamera; in particolare le formule utilizzate sono:

$$\beta = \frac{D_v}{2} + \arcsin\left(\frac{10^{-6} kn\lambda}{2\cos(\frac{D_v}{2})}\right) \tag{2.1}$$

  Dove β = angolo di diffrazione, k = ordine di diffrazione = 1, n = valore del grating = 1200/2400, $\frac{D_v}{2}$ = angolo di deviazione = 0.144.

$$BP = \frac{10^{-6} w\cos(x)}{knL_B} \tag{2.2}$$

Con BP = banda passante, w = larghezza della fenditura di entrata $= 0.0067mm$, L_B = lunghezza del braccio di uscita $= 451.87mm$.
In questa case structure ne è contenuta anche una associata al bottone booleano *Subtract Dark* per eliminare un eventuale background acquisito in precedenza.

- Quinto blocco, figura 2.11. Consente al programma di visualizzare l'immagine o lo spettro, in base alla scelta dell'utente. In questo punto del programma vengono anche effettuati i calcoli dei valori *FWHM*, *Max Peak* e *WL*.

Terzo blocco

Questo blocco, in figura 2.12, contiene solamente il subVI *SCSetMode.vi* che termina il funzionamento della fotocamera attraverso il valore adatto di un enumerativo. Anche se quest'ultimo passaggio potrebbe sembrare inutile è sempre bene ricordare che tutti i programmi che vengono eseguiti impiegano un certo numero di risorse: utilizzare le giuste funzioni o i giusti subVI di chiusura anche per gli strumenti coinvolti permette di liberare le risorse impegnate scansando il rischio di un uso ridondante della memoria.

2.4 Considerazioni sul codice

L'analisi del codice già esistente si è resa necessaria in quanto uno dei due strumenti, in particolare il monocromatore, non è stato sostituito. Ad una prima lettura superficiale il codice risulta funzionante. Tuttavia in seguito alla partecipazione ad un corso base del linguaggio grafico LabVIEW (LabVIEW Core 1 e LabVIEW Core 2) e al suo superamento sono emersi alcuni difetti del codice.
Per prima cosa è evidente che il codice nel block diagram appare molto caotico e difficile da decifrare; è servito infatti uno studio piuttosto lungo per comprendere la logica con cui è stato creato. Mancando di commenti e nomi

chiari si rivela difficile intuire lo scopo di strutture, variabili e formule inserite. Un altro difetto, che si potrebbe considerare un vero e proprio errore di programmazione, è la mancata considerazione dell'andamento del flusso dei dati e l'uso massiccio di flat sequence: infatti è assolutamente sconsigliato utilizzare flat sequence in quanto forzano il flusso dei dati rischiando di creare errori altrimenti evitabili. È sempre un buon criterio di programmazione cercare di organizzare il codice in modo da sfruttare il normale flusso dei dati e manipolarlo allo scopo di ottenere la logica desiderata.
Si nota anche la totale assenza di subVI che aiutano a rendere il codice flessibile e decisamente più leggibile. Facendo un confronto con qualsiasi altro linguaggio di programmazione che utilizza linee di codice scritto, un programma diviso in più file in base a determinati criteri risulta molto più ordinato, organico e leggile di un programma scritto in unico file, lungo magari migliaia di righe, in cui si fatica a comprenderne la logica e lo scopo delle variabili. Lo stesso concetto vale per VI e subVI nel linguaggio di programmazione grafico LabVIEW.

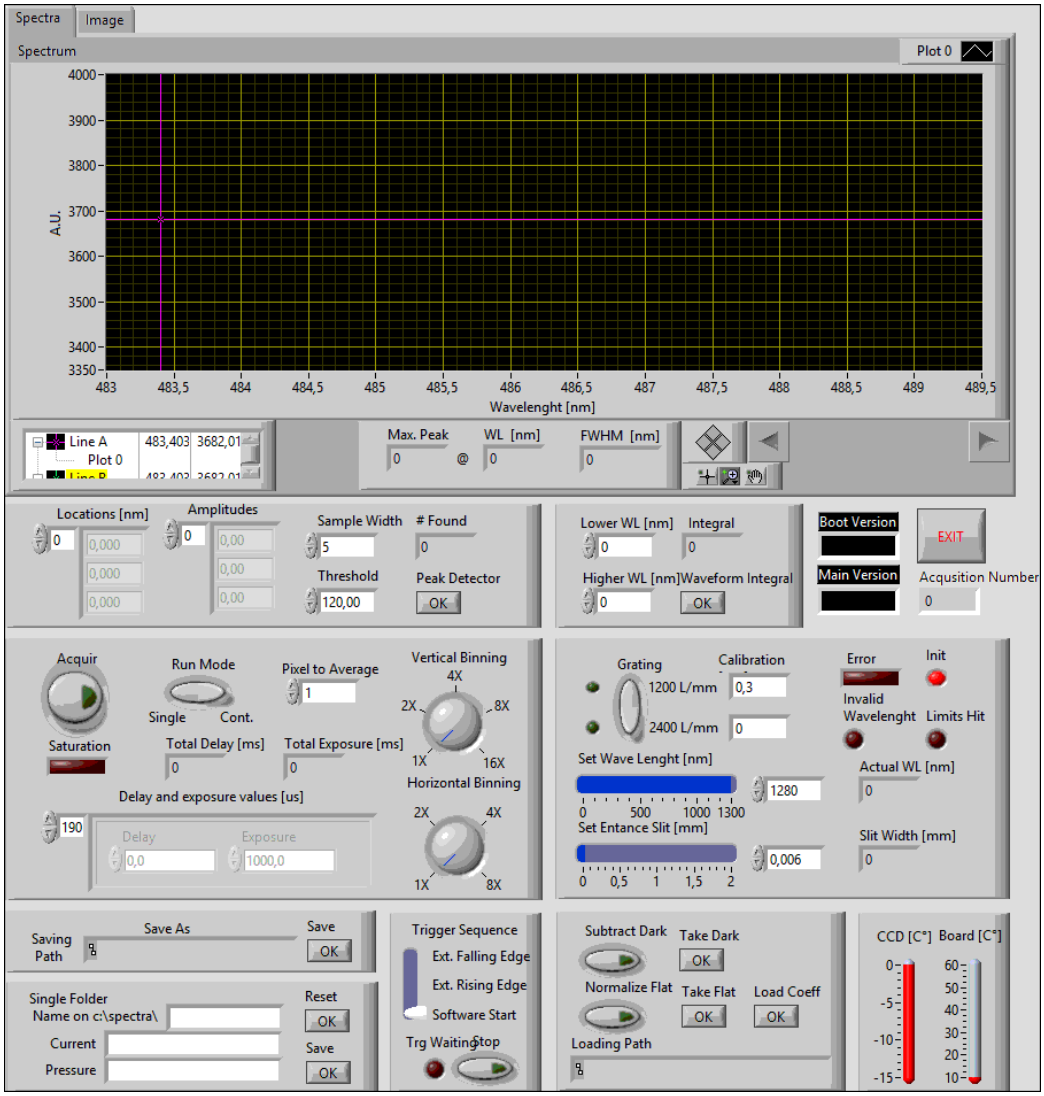

Figura 2.3: Front Panel del progetto preesistente

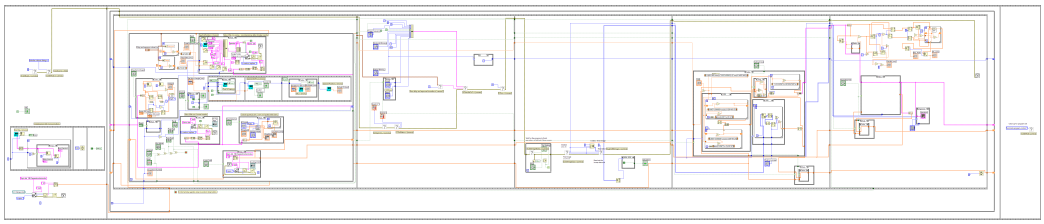

Figura 2.4: Block Diagram del progetto preesistente

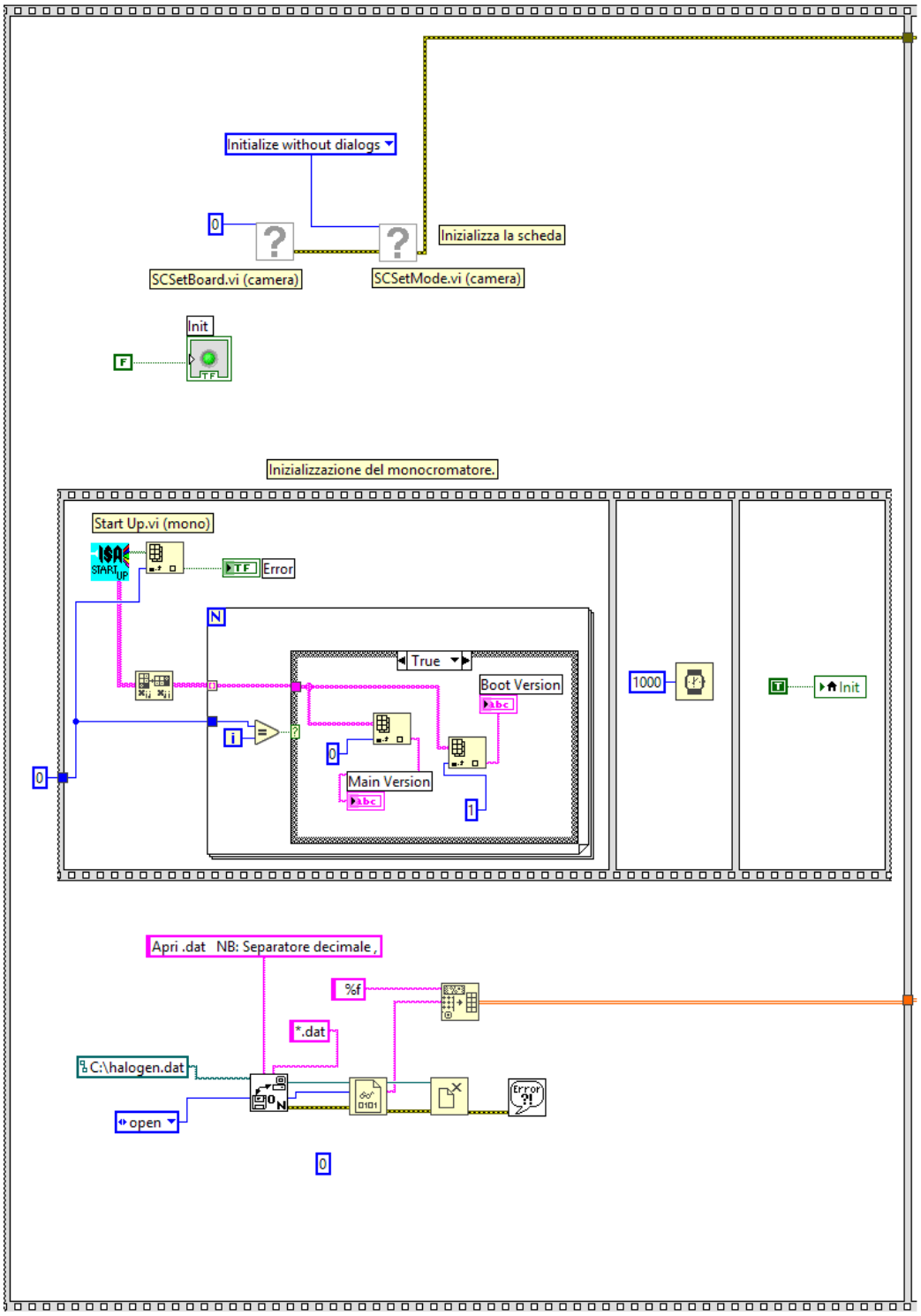

Figura 2.5: Block Diagram del progetto preesistente, primo blocco

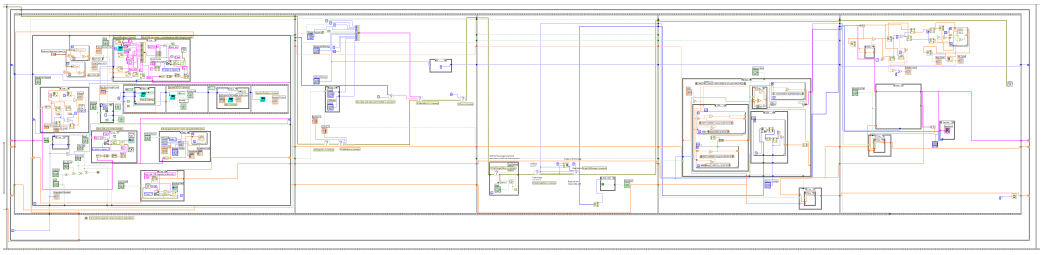

Figura 2.6: Block Diagram del progetto preesistente, secondo blocco

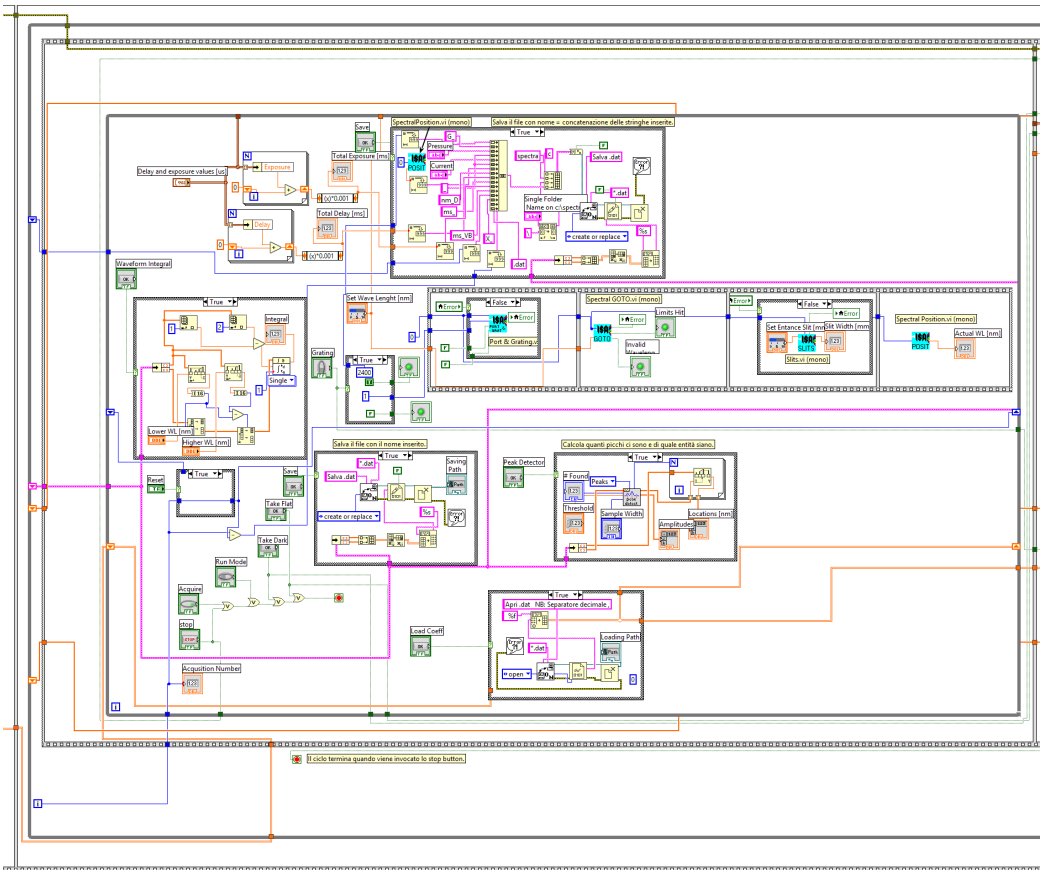

Figura 2.7: Block Diagram del progetto preesistente, secondo blocco, primo blocco della flat sequence interna

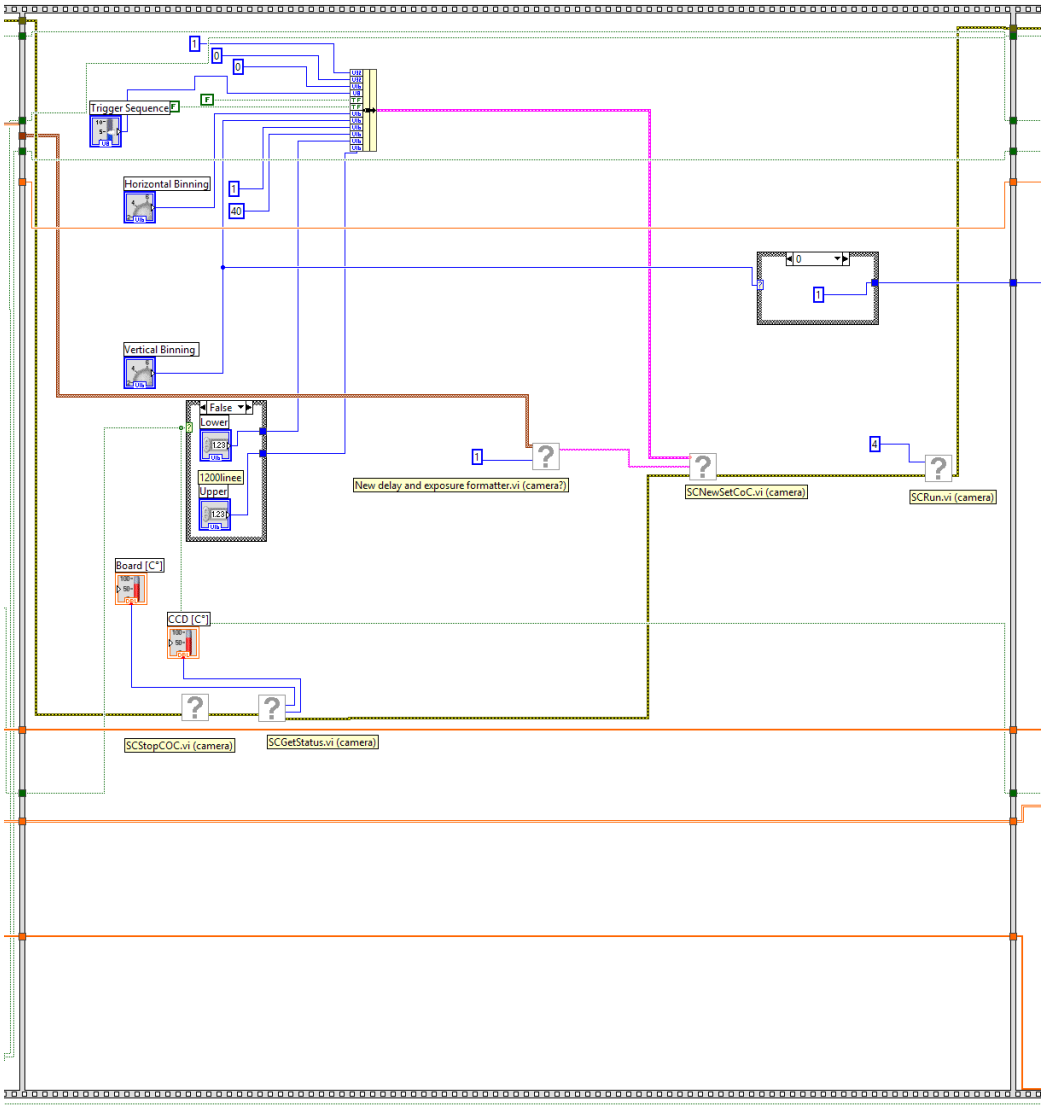

Figura 2.8: Block Diagram del progetto preesistente, secondo blocco, secondo blocco della flat sequence interna

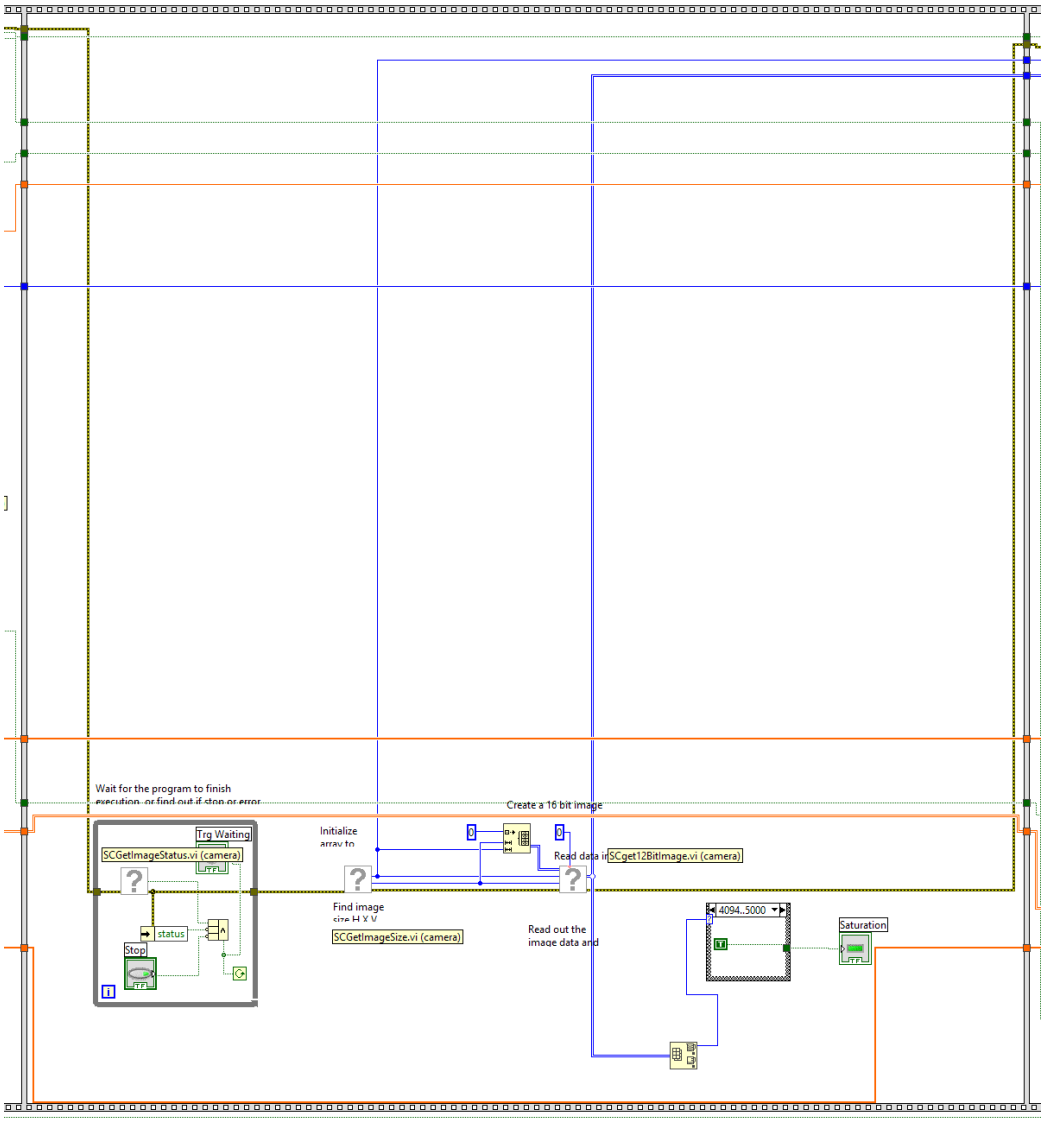

Figura 2.9: Block Diagram del progetto preesistente, secondo blocco, terzo blocco della flat sequence interna

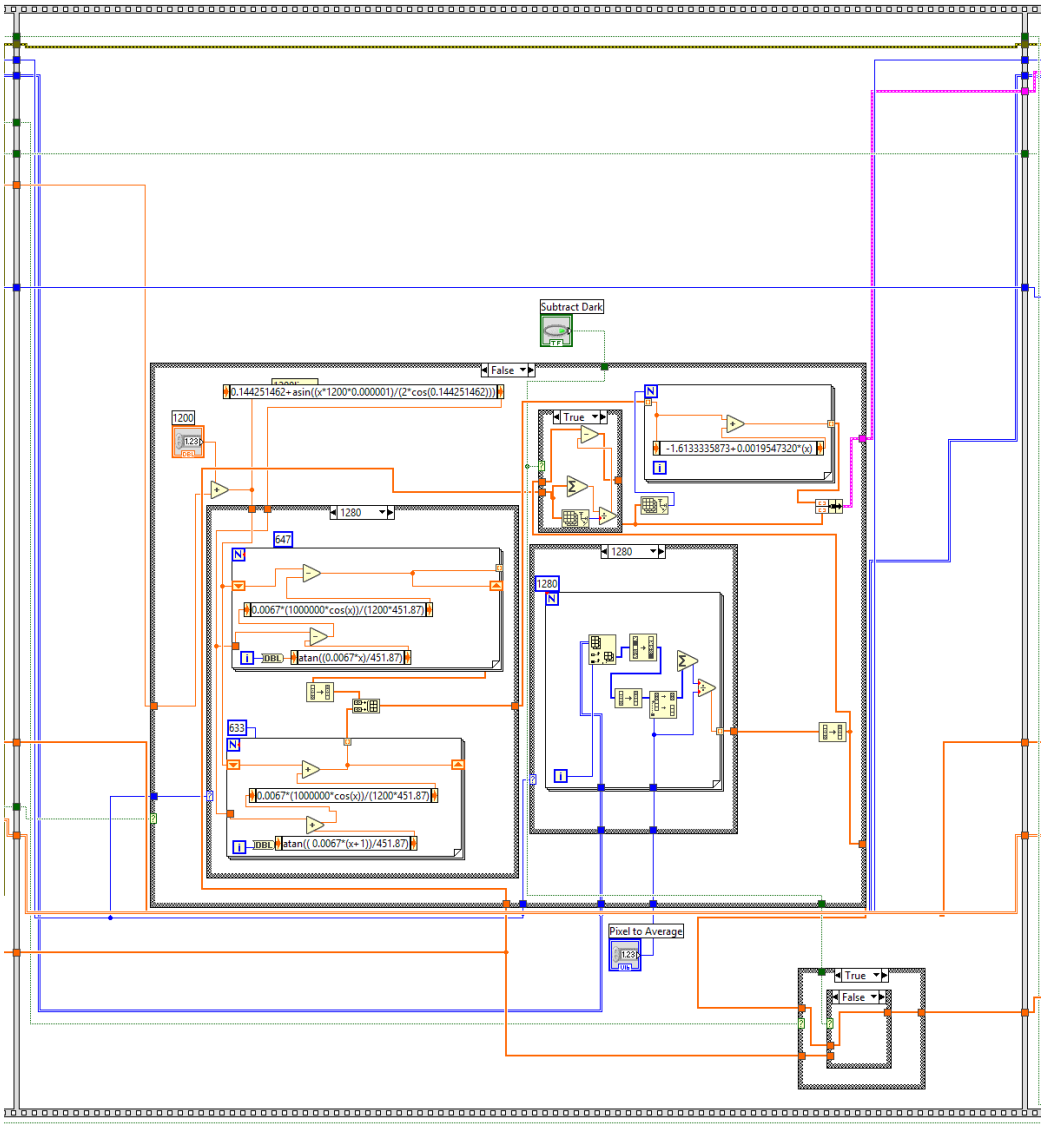

Figura 2.10: Block Diagram del progetto preesistente, secondo blocco, quarto blocco della flat sequence interna

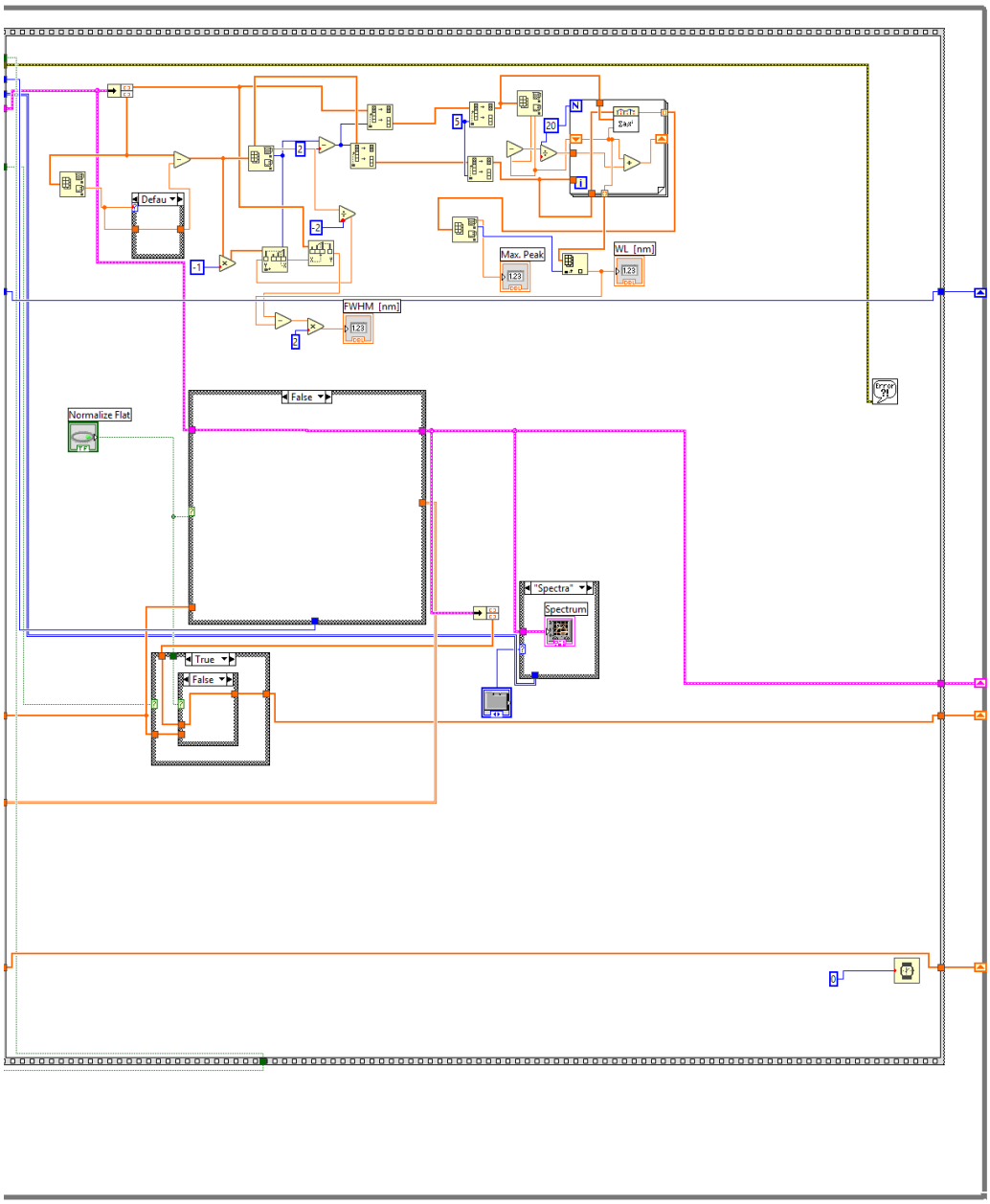

Figura 2.11: Block Diagram del progetto preesistente, secondo blocco, quinto blocco della flat sequence interna

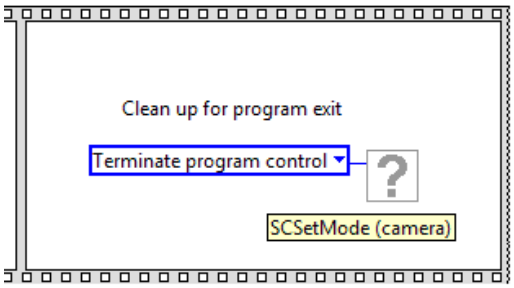

Figura 2.12: Block Diagram del progetto preesistente, terzo blocco

Capitolo 3

Fase preliminare alla realizzazione del progetto

Prima della realizzazione del progetto per controllare entrambi gli strumenti, si è optato per la scrittura di due progetti separati. In questo modo è stato possibile verificare la versione del codice più efficacie per ogni strumento, così da poter unire in seguito le due logiche, evitando conflitti.

3.1 Fotocamera

Per entrare in confidenza con il linguaggio di programmazione grafico LabVIEW e con lo strumento, si è reso necessario partire da un esempio fornito con le API LabVIEW legate alla nuova fotocamera intensificata [2].

3.1.1 Breve descrizione dello strumento

La fotocamera intensificata per cui si deve scrivere il programma che la controlli è la 4 Picos con CCD intensificato di Stanford Computer Optics, Inc. (per la descrizione dello strumento si faccia riferimento alla sottosezione 2.1.2).

3.1.2 Descrizione dell'esempio fornito

L'esempio fornito è molto semplice e non contiene tutte le opzioni disponibili per la fotocamera. Analizziamo come fatto in precedenza i due componenti principali del programma: front panel e block diagram.

Front Panel

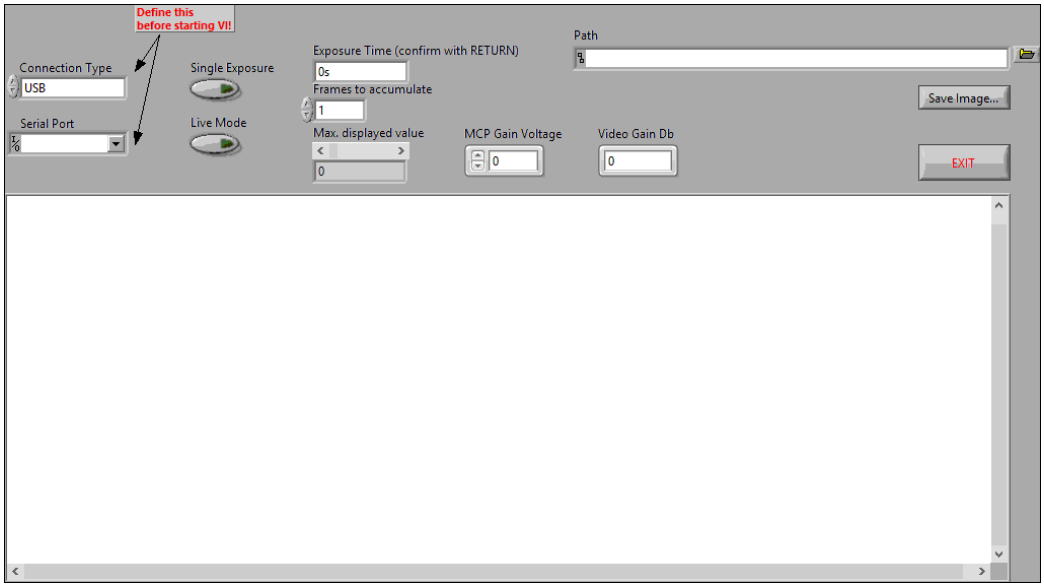

Figura 3.1: Front Panel del progetto per sola fotocamera

Nel front panel in figura 3.1 possiamo vedere due controlli che è obbligatorio definire prima della partenza del VI, e sono il tipo di connessione *Connection Type* e la porta seriale *Serial Port* nel caso in cui il tipo di connessione sia *Analog*. È infatti permesso scegliere fra tre tipi di connessione allo strumento: *USB* (che è la connessione usata nel nostro caso), *CameraLink* o *Analog*. A fianco di questi due controlli troviamo due bottoni booleani per acquisire immagini in un singolo frame oppure in modo continuo. È poi possibile inserire il tempo di esposizione *Exposure Time*, il numero di frame da catturare *Frames to accumulate*, e modificare il valore *Max. displayed value* del fondo scala. Un indicatore di immagine rende possibile la visualiz-

zazione dell'immagine e un bottone booleano permette di salvare l'immagine. Infine troviamo un controllo per terminare il programma.

Block Diagram

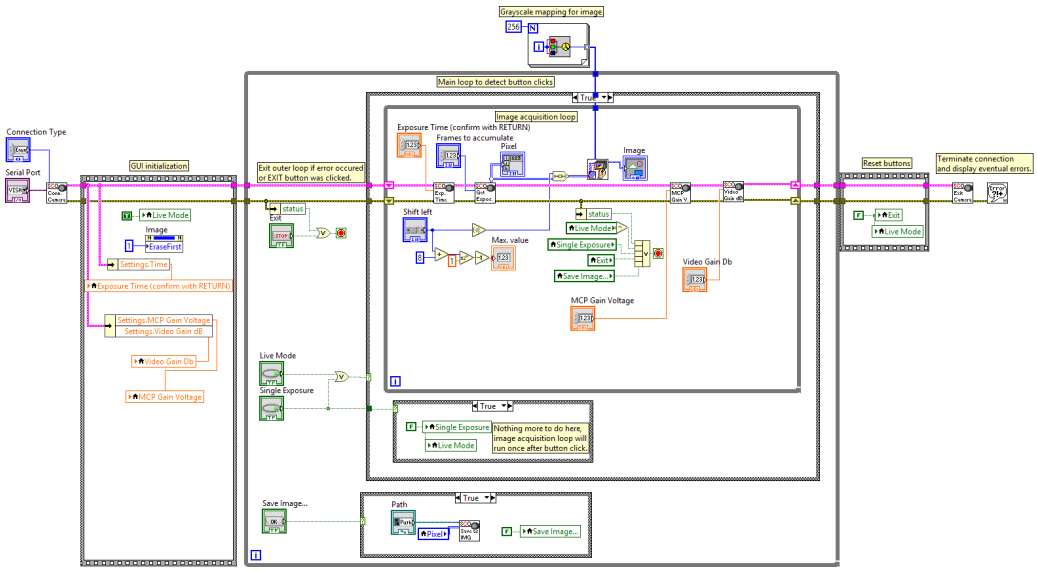

Figura 3.2: Block Diagram del progetto per sola fotocamera

Si può considerare il bock diagram, in figura 3.2, idealmente diviso in tre parti che eseguono in sequenza: una parte di inizializzazione, una intermedia di funzionamento del programma, e una finale di chiusura graceful dell'applicazione.

Il primo subVI invocato è *ConnectCamera.vi* e deve essere invocato prima che la fotocamera sia utilizzata, imposta la connessione della fotocamera dipendentemente dall'input immesso dall'utente; restituisce in uscita un cluster contenente tutte le informazioni che riguardano le impostazioni della fotocamera. Tale cluster di dati, insieme al cluster di errore, entra in una flat sequence che recupera le impostazioni memorizzate da un uso antecedente dello strumento e le mostra nei controlli appositi nel front panel; viene anche impostato il bottone booleano *Live Mode* a false.

In seguito alla flat sequence troviamo un ciclo while con lo scopo di indivi-

duare il cambiamento dei bottoni e dei controlli. All'interno di questo ciclo sono presenti due case structure: la prima associata al bottone *Save Image...* che permette di salvare l'immagine, la seconda associata all'*or* fra il valore dei due bottoni *Single Exposure* e *Live Mode*. Viene eseguito quindi il codice all'interno di quest'ultima case structure qualora o un bottone o l'altro esclusivamente (grazie ad un'altra case structure interna) abbia il valore true. Una volta dentro la case structure viene eseguito un ciclo while all'interno del quale vengono invocate alcune API LabVIEW della fotocamera: dapprima *SetExposureTime.vi* con la quale si setta il tempo di esposizione, poi *GetExposure.vi* che restituisce la matrice di pixel corrispondente alla foto. Attraverso alcuni calcoli che tengono in considerazione anche il valore di fondo scala inserito viene generata l'immagine da visualizzare del controllo destinato. Il ciclo while interno si arresta se viene presa una singola immagine, oppure se non si è selezionata la modalità Live Mode, o se si è premuto il bottone *Exit* per uscire dal ciclo o quello di salvataggio dell'immagine, o ancora se si è verificato qualche errore. Il ciclo esterno si arresta invece se si è verificato qualche errore o se si è invocato il bottone *Exit* per fermarlo.
All'esterno di tale ciclo viene eseguito l'ultimo frammento di codice: in una flat sequence vengono resettati i bottoni di *Exit* e *Live Mode*, impostandoli a false, e infine viene invocato l'ultimo subVI relativo alla fotocamera, *ExitCamera.vi*, che ha il compito di rilasciare le risorse impegnate dal programma e arrestare la fotocamera nel modo corretto.

3.2 Monocromatore

Ad un primo tentativo di refactoring del progetto preesistente il codice relativo al controllo del monocromatore si è rivelato troppo caotico e sparso. Per questo si è realizzato un nuovo programma, prendendo ovviamente spunto dal precedente, dedicato solo al monocromatore, che rispettasse il corretto flusso dei dati e non facesse uso di flat sequence. Si descrivono di seguito front panel e block diagram di tale programma [3].

3.2.1 Front Panel

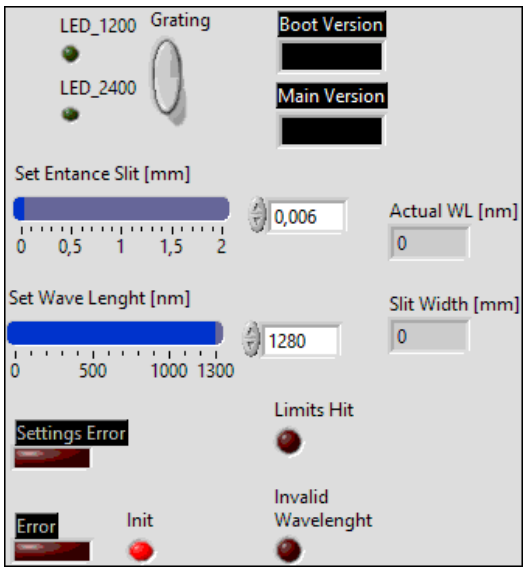

Figura 3.3: Front Panel del progetto per solo monocromatore

Gli elementi presenti nel front panel in figura 3.3 sono gli stessi dedicati al monocromatore nel precedente progetto, li rivedremo comunque brevemente. I controlli presenti sono:

- *Grating* per distinguere il tipo di grating (1200 o 2400)
- *Set Entrance Slit* per impostare la slit di entrata
- *Set Wave Length* per inserire la lunghezza d'onda desiderata

Mentre gli indicatori sono:

- *Init* led che segnala se il monocromatore è stato inizializzato correttamente
- *Error* led che segnala la presenza di eventuali errori in fase di inizializzazione
- *Boot/Main Version* informazioni sulla versione del programma in uso

- *Actual WL* lunghezza d'onda effettiva inserita
- *Slit Width* slit di entrata inserita
- *Settings Error* led che segnala un eventuale errore generico nel settaggio delle impostazioni del monocromatore
- *Limits Hit* led che segnala l'inserimento di una lunghezza d'onda troppo alta
- *Invalid Wavelength* led che segnala l'inserimento di una lunghezza d'onda non valida

3.2.2 Block Diagram

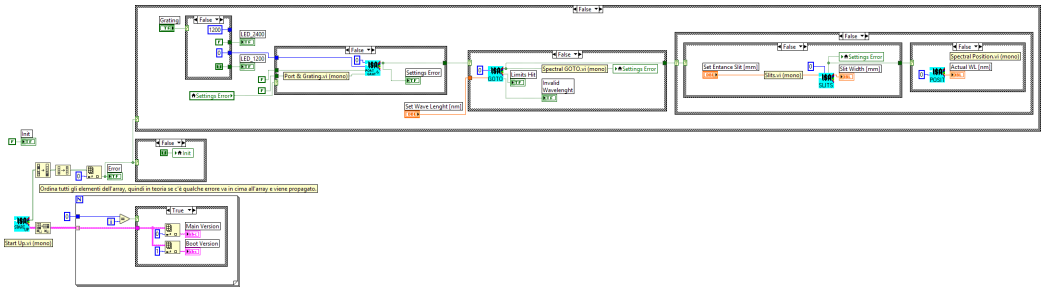

Figura 3.4: Block Diagram del progetto per solo monocromatore

Per la realizzazione del block diagram in figura 3.4 si è chiaramente preso spunto da quello già esistente, ma si è impiegato un consistente refactoring in particolare per eliminare le flat sequence.
Il primo subVI ad essere invocato è *Start Up.vi* che porta i componenti interni del monocromatore nella posizione di default e restituisce il numero di versione del programma. Tale subVI restituisce anche un array di errori: esso viene ordinato in ordine crescente e ne viene estratto l'elemento in cima, se vi è un errore allora non è possibile continuare con il programma, in caso contrario si può procedere. Si entra quindi (in assenza di errori) nella case structure che permette di impostare come si desidera il monocromatore. I subVI che vengono invocati, in sequenza, sono: *Port & Grating.vi* per

specificare il grating, *Spectral GOTO.vi* per portare gli elementi interni alla lunghezza d'onda inserita dall'utente, *Slits.vi* per inserire la slit di entrata e *Spectral Position.vi* per completare gli indicatori corrispondenti. Chiaramente ognuno di questi subVI è inserito all'interno di una case structure e verrano eseguiti solo qualora il subVI precedente non abbia restituito in uscita errori. In questo modo si eliminano completamente le flat sequence e si sfrutta l'andamento del flusso dei dati.

Capitolo 4

Realizzazione del progetto

Gli strumenti coinvolti nel progetto sono quelli descritti nei capitoli precedenti, il monocromatore Jobin-Yvon HR460, si veda la sottosezione 2.1.1, e la fotocamera intensificata 4 Picos, accenno alla tecnologia della stessa nella sottosezione 2.1.2. Di seguito si descrive il programma realizzato per il controllo dei due strumenti attraverso un unico programma.

4.1 Approccio iniziale

Avendo davanti un progetto già scritto e funzionante, anche se per una fotocamera diversa, il primo approccio è stato quello di cercare di integrare il vecchio programma sostituendo i subVI della fotocamera precedente con quelli della fotocamera attuale. Tale tentativo si è rivelato fin dall'inizio fallimentare in quanto il codice del vecchio progetto risulta troppo caotico e avviluppato per essere modificato senza operare ingenti cambiamenti. Si è quindi optato per una riscrittura completa del codice prendendo sempre spunto da quello già scritto per le parti riutilizzabili.

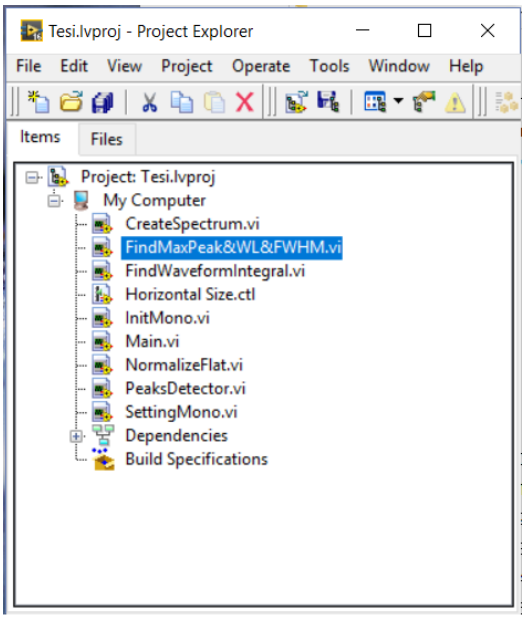

Figura 4.1: Struttura del progetto

4.2 Struttura del progetto

Per rendere il programma maggiormente organico e comprensibile si è deciso di creare un progetto LabVIEW: in questo modo i file e i subVI coinvolti sono raggruppati in un unico progetto (figura 4.1), che risulta sicuramente più gestibile di un semplice insieme di file.
Si è creato all'inizio un VI principale, un *Main.vi* che ha lo stesso scopo dei *main* degli altri linguaggi di programmazione. In seguito si è cercato di creare più subVI possibile per rendere il programma leggibile e ordinato. Si descriveranno più avanti i subVI contenuti nel progetto e i loro scopi.

4.3 Front Panel

La struttura pensata per il front panel di questo progetto è quella dell'interfaccia grafica a *tab* (controllo grafico di navigazione che permette all'utente di muoversi da un gruppo di controlli a un altro). Tale widget permette un'i-

deale suddivisione delle diverse aree di competenza del programma: di seguito verranno elencate con una descrizione degli elementi contenuti in ognuna.

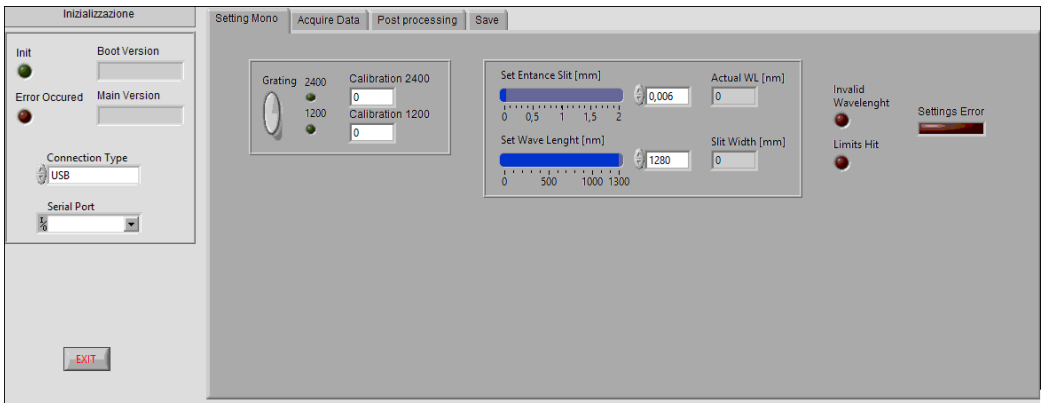

Figura 4.2: Front Panel del progetto, tab *Setting Mono*

- *Setting Mono*, figura 4.2, contiene i controlli necessari al settaggio del monocromatore. *Grating* per impostare il grating (1200 o 2400) e rispettivi controlli per specificarne un eventuale calibrazione. *Set Entrance Slit* e *Set Wave Length* per impostare la slit di entrata e la lunghezza d'onda. *Actual WL* e *Slit Width* per visualizzare lunghezza d'onda e slit effettivamente inserite. Sono presenti inoltre tre led: *Settings Error* indica un errore generico di settaggio, *Invalid Wavelength* indica l'inserimento di una lunghezza d'onda non valida e *Limits hit* evidenzia l'inserimento di un valore oltre i limiti consentiti.

- *Acquire Data*, figura 4.3, contiene tutti i controlli per poter impostare la fotocamera e acquisire i dati.

- *Post Processing*, figura 4.4, permette di visualizzare lo spettro corrispondente all'immagine acquisita ed effettuare dei calcoli su tali dati. Con *Waveform Integral* è possibile calcolare l'integrale della lunghezza d'onda su un certo intervallo. *Peak Detector* consente di visualizzare il numero di picchi trovati al di sopra di una certa soglia da inserire e i loro valori. Attraverso il controllo *Pixel To Average* è possibile scegliere

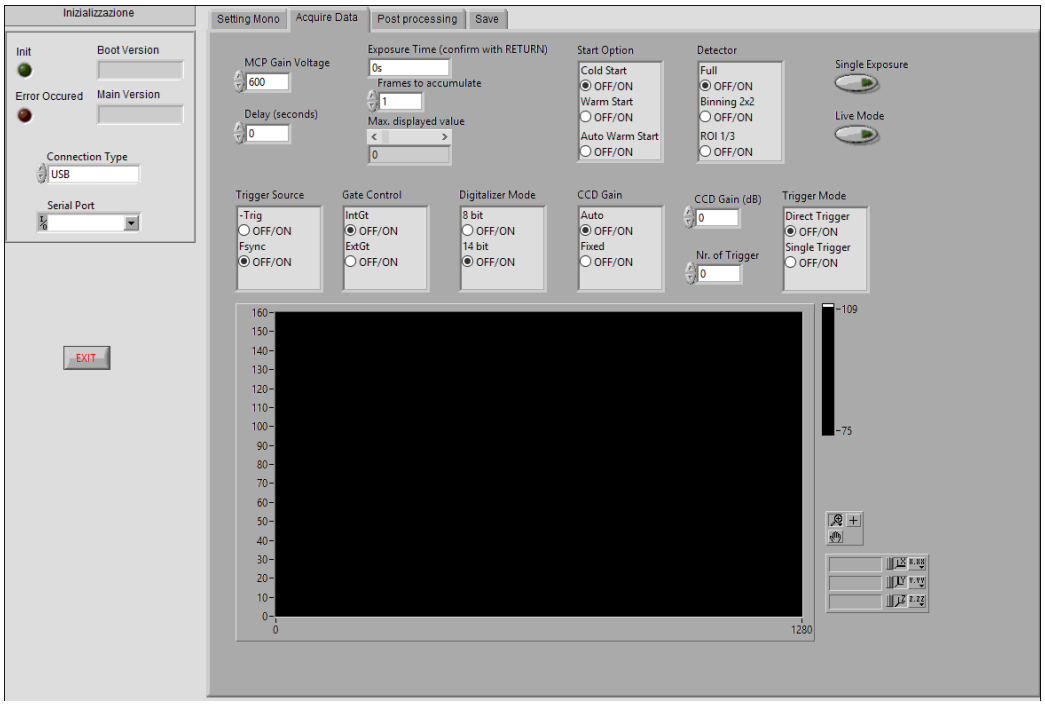

Figura 4.3: Front Panel del progetto, tab *Acquire Data*

il numero di pixel da con cui fare la media da visualizzare poi nel grafico. Sono stati mantenuti anche i controlli per acquisire il background e sottrarlo alla foto.

- *Save*, figura 4.5, presenta due controlli booleani: *Save* salva i dati ben formattati in un file di testo che può essere aperto successivamente con altri editor, *Save Image* salva invece l'immagine vera e propria.

Al di fuori di questo controllo è presente un pannello dedicato all'inizializzazione degli strumenti. Il led *Init* segnala che il monocromatore è stato inizializzato senza errori e viene visualizzata anche la versione del programma, in caso contrario si attiva il led *Error Occured*. Sono presenti anche due controlli per specificare la connessione che si intende utilizzare verso la fotocamera: *USB* per connetterla al calcolatore attraverso cavo USB, *CameraLink* o *Analog* per altre modalità di connessione non contemplate in questo progetto.

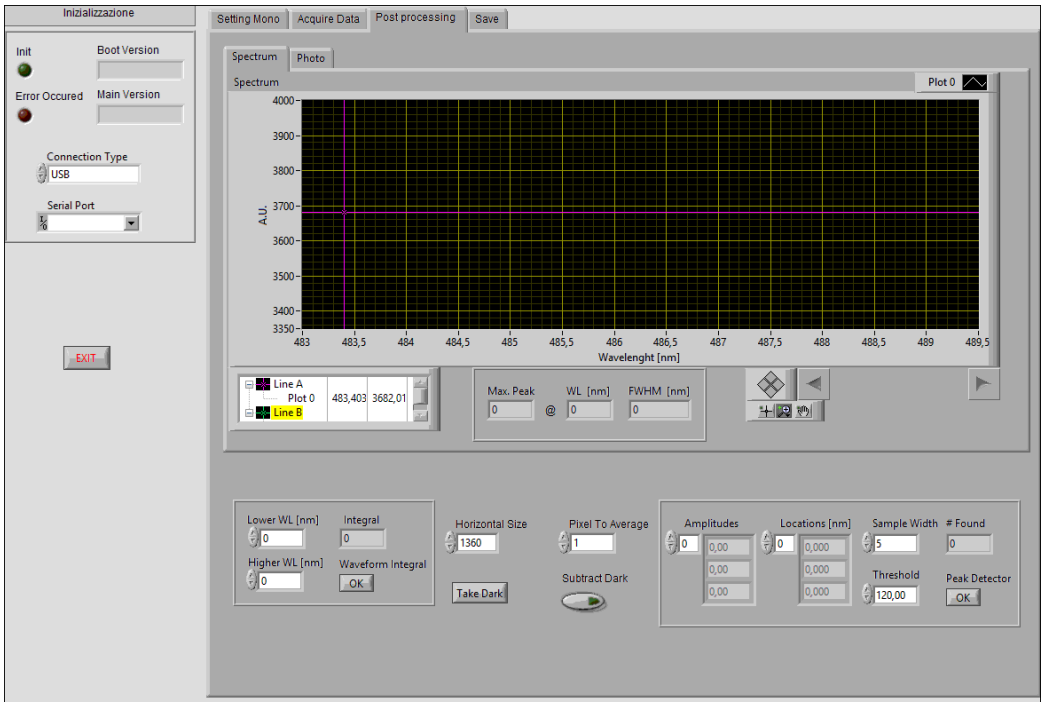

Figura 4.4: Front Panel del progetto, tab *Post Processing*

Infine il bottone *Exit* consente di arrestare il programma nel modo corretto rilasciando le risorse impiegate.

4.4 Block Diagram

Come detto sopra il programma è stato organizzato in un progetto che raggruppa tutti i subVI creati nel corso della scrittura del codice. Partendo dal VI principale *Main.vi* si descrive il block diagram delineando poi nello specifico i singoli subVI.

Il primo subVI che viene eseguito è *InitMono.vi* che, come si può intuire dal nome, ha il compito di inizializzare il monocromatore; solo se tale subVI esegue restituendo nessun errore è possibile continuare con il resto del programma, del resto non avrebbe senso se lo strumento che deve generare il soggetto delle immagini non funzionasse correttamente. Se quindi non ci sono errori in fase di inizializzazione del monocromatore si entra in una case

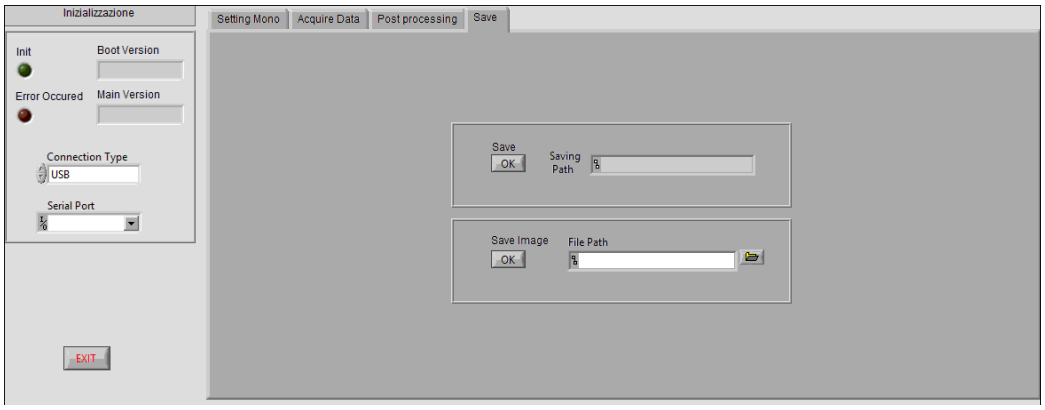

Figura 4.5: Front Panel del progetto, tab *Save*

structure che racchiude tutto il resto del programma: per prima cosa viene eseguito il subVI di connessione alla fotocamera *ConnectCamera.vi* il cui output entra poi in una flat sequence atta a visualizzare le impostazioni della fotocamera inserite l'ultima volta che è stata utilizzata. Successivamente si entra in un ciclo while per rilevare le azioni dell'utente; all'interno di questo ciclo si trova una case structure legata al *tab control* che permette alla struttura di discriminare il codice in base alla tab selezionata dall'utente. La scelta di tale schema prende spunto dal design pattern di macchina a stati finiti: non è presente un enumerativo e non c'è una vera e propria sequenza che viene eseguita, ma lo schema generale ricorda tale pattern. Analizziamo quindi il codice eseguito nei quattro frame:

Setting Mono Figura 4.6. All'interno di un'ulteriore case structure viene eseguito il subVI *SettingMono.vi*, si faccia riferimento al codice descritto nella sezione 3.2.

Acquire Data Figura 4.7. Per questo frame della case structure si è scelto di non creare subVI in quanto le variabili di input e output sarebbero state troppo numerose e avrebbero reso il codice incomprensibile. Viene quindi eseguito il codice per l'acquisizione delle immagini (si fa riferimento a quello spiegato nella sezione 3.1). Il ciclo per l'acquisizio-

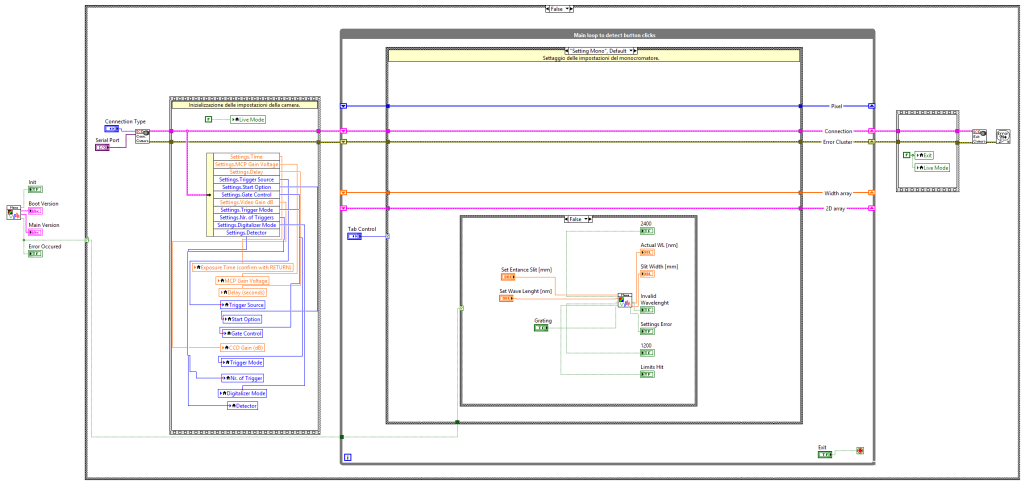

Figura 4.6: Block Diagram del progetto

ne viene arrestato anche se viene selezionata dall'utente un'altra tab.

Post processing Figura 4.8. In questa frame sono stati inserite le elaborazioni sui dati che possono essere eseguite in seguito all'acquisizione dei dati. Abbiamo quindi una case structure che permette di visualizzare i dati come immagine o come spettro. Ci sono poi tre subVI: *CreateSpectrum.vi* effettua i calcoli necessari per generare lo spettro da visualizzare, *PeaksDetector.vi* restituisce i picchi sopra una certa soglia, *FindWavefromIntegral.vi* calcola l'integrale su un intervallo della lunghezza d'onda.

Save Figura 4.9. Questo frame contiene due opzioni di salvataggio: *Save* per salvare i dati in un file di testo, *Save Image* per salvare l'immagine.

I shift register del ciclo esterno sono cinque: *Pixel* contiene la matrice di pixel che compone l'immagine, *Connection* contiene tutte le informazioni relative alla fotocamera, *Error Cluster* riporta gli eventuali errori generati dalla fotocamera, *Width Array* è l'array che contiene i valori delle lunghezze d'onda, *2D Array* è la matrice con i valori da inserire nel grafico dello spettro. Se infine viene arrestato il ciclo con il comando *Exit*, viene eseguita l'ultima

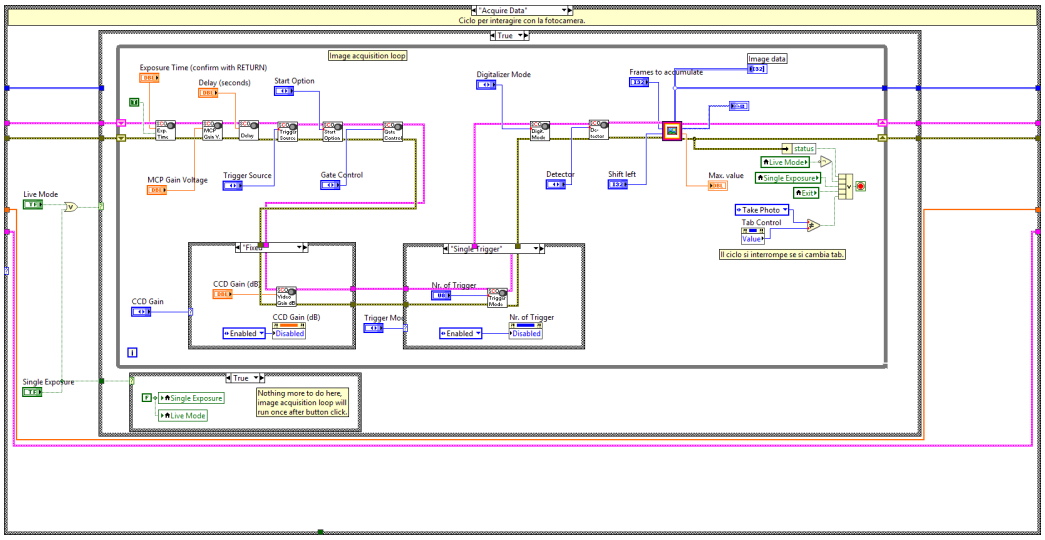

Figura 4.7: Block Diagram del progetto, frame “Acquire Data”

flat sequence che resetta i valori dei controlli di acquisizione delle immagini e viene eseguito anche il subVI *ExitCamera.vi* per la chiusura graceful del programma e il rilascio delle risorse.

4.4.1 Descrizione subVI

Si descrivono di seguito i singoli subVI e le loro funzionalità. Saranno analizzati solo i block diagram, in quanto i front panel hanno il solo scopo di mostrare input e output.

InitMono.vi Figura 4.10. Il compito di questo subVI è quello di interrogare il monocromatore riportandolo al suo stato di default. Rispetto al codice presente nel progetto antecedente è stato leggermente modificato. Sono state eliminate tutte le flat sequence, che forzano il flusso logico dei dati. Dopo che è stato invocato il subVI del monocromatore *Start Up.vi*, il led *Init* viene impostato a true solo ed esclusivamente se non sono presenti errori nella matrice di errori restituita dallo stesso (si veda la sezione 3.2).

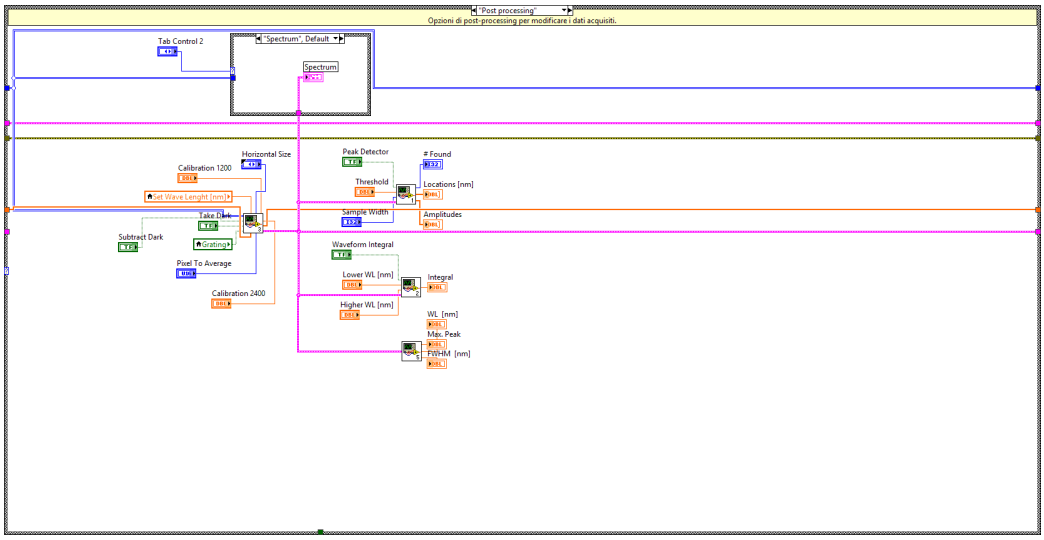

Figura 4.8: Block Diagram del progetto, frame "Post Processing"

SettingMono.vi Figura 4.11. Anche questo subVI è stato modificato rispetto al codice precedente eliminando tutte le flat sequence (si veda la sezione 3.2).

CreateImage.vi Figura 4.12. In seguito all'invocazione del subVI della fotocamera *GetExposure.vi* viene invocata la funzione *Draw Unflattened Pixmap.vi* per creare l'immagine vera e propria che possa essere visualizzata in un indicatore apposito.

PeaksDetector.vi Figura 4.13. Lo scopo di questo subVI è quello di trovare, attraverso la funzione *Waveform Peak Detection.vi*, il numero di picchi presenti nella matrice in ingresso con rispettivi valori di lunghezza d'onda e intensità del picco.

FindWavefromIntegral.vi Figura 4.14. Data in ingresso la matrice con lunghezza d'onda e intensità dello spettro, questo subVI calcola l'integrale, con apposite funzioni, su un intervallo in input.

CreateSpectrum.vi Figura 4.15. Come si può intuire dal nome, l'output di questo subVI consiste in una matrice di valori che rappresentano lun-

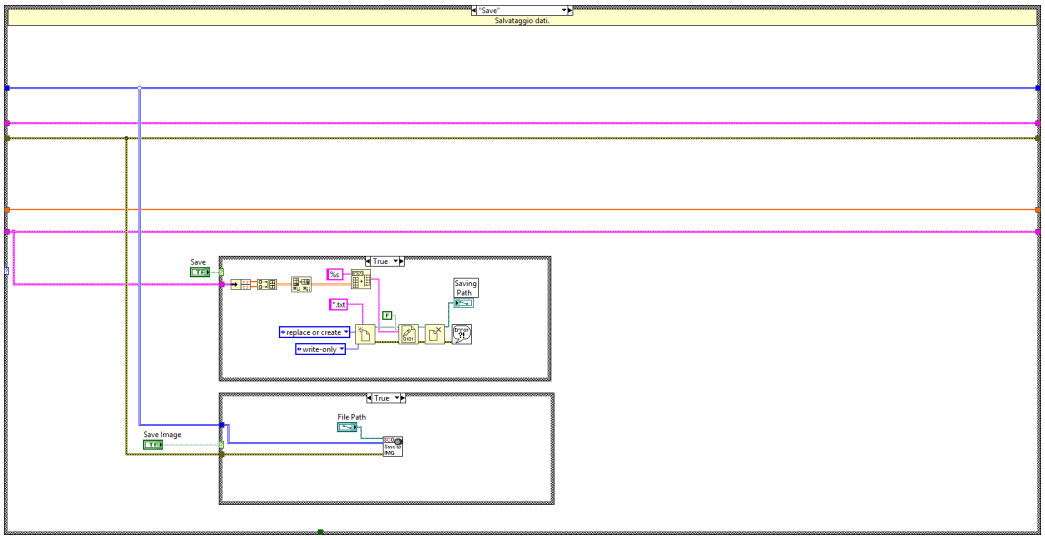

Figura 4.9: Block Diagram del progetto, frame "Save"

ghezze d'onda con rispettiva intensità dello spettro. I calcoli presenti sono stati riportati dal progetto precedente in quanto legati al monocromatore (si veda la sottosezione 2.3.2). È stato modificato il valore del numero di pixel della larghezza dell'immagine (1360) e il valore di w (0.0047).

FindMaxPeak&WL&FWHM.vi Figura 4.16. Questo subVI produce due risultati: restituisce il picco massimo e la sua corrispettiva lunghezza d'onda, e il valore della FWHM (Full Width at Half Maximum).

Nella figura 4.17 si possono vedere le icone con input e output di tutti i subVI citati.

4.5 Prova di utilizzo del programma

Per provare l'effettivo funzionamento del programma, con l'ausilio di una lampada al mercurio e una fibra ottica, abbiamo fotografato lo spettro del mercurio. Dopo aver settato il monocromatore a $365nm$ di lunghezza d'onda, grating 2400 e slit di entrata $0.6mm$, e la fotocamera con $3ms$ di tempo di

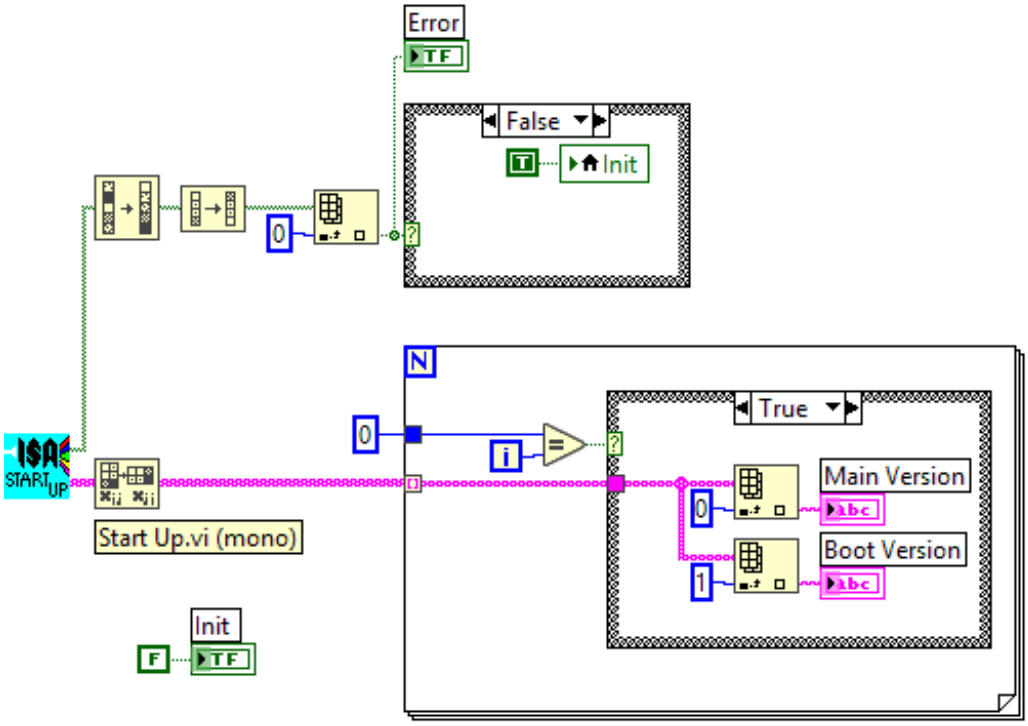

Figura 4.10: Block Diagram del subVI *InitMono.vi*

esposizione e 820V di MCP Gain, abbiamo acquisito un'immagine.

Nella figura 4.18 si possono vedere tre linee bianche di intensità decrescente che rappresentano lo spettro del mercurio; tale spettro si può vedere nella figura 4.19 che mostra il grafico ottenuto dall'immagine. Lo spettro è formato da tre picchi, di cui uno di maggiore intensità rispetto agli altri due, e corrispondono alle tre linee che compaiono nell'immagine.

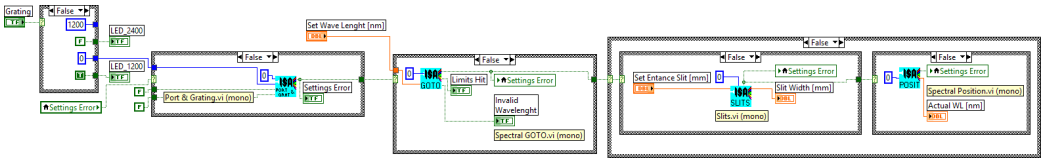

Figura 4.11: Block Diagram del subVI *SettingMono.vi*

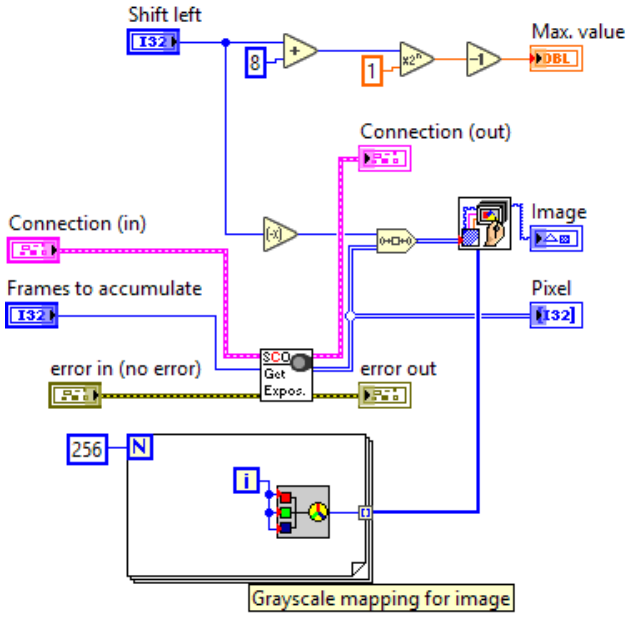

Figura 4.12: Block Diagram del subVI *CreateImage.vi*

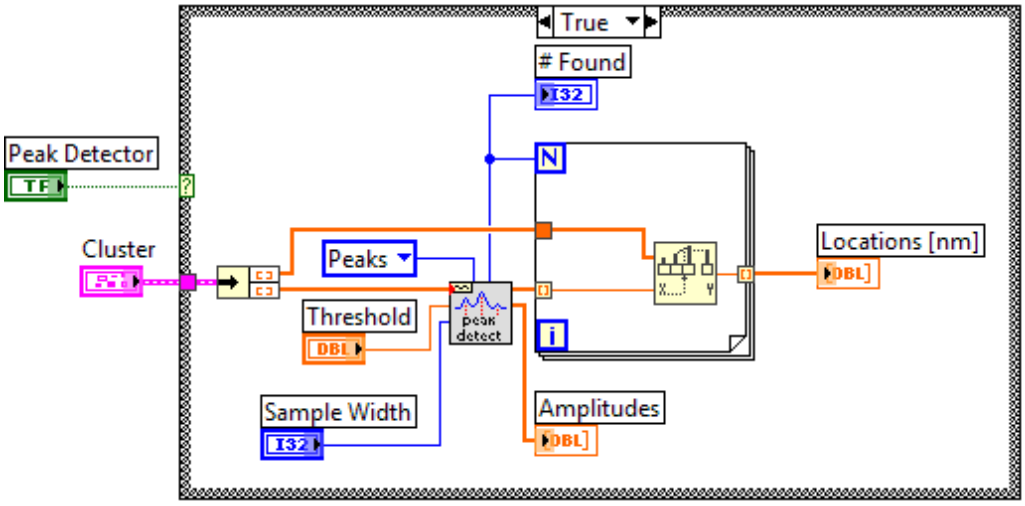

Figura 4.13: Block Diagram del subVI *PeaksDetector.vi*

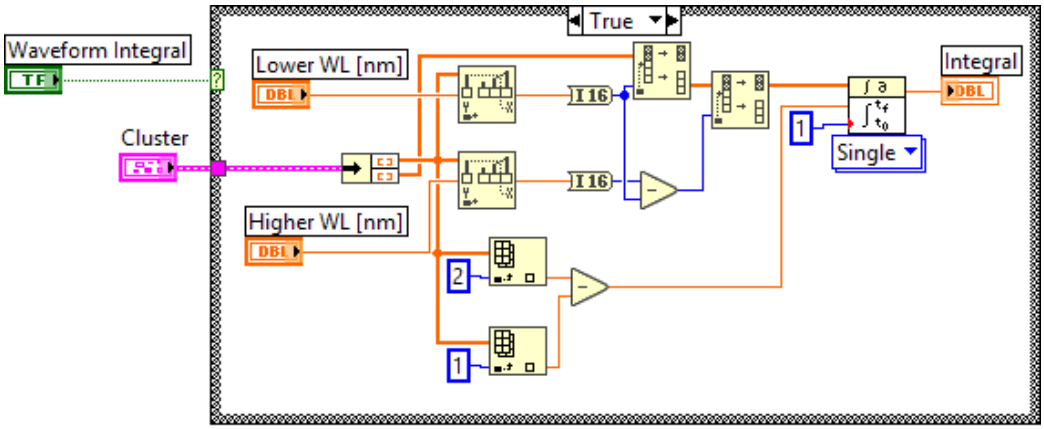

Figura 4.14: Block Diagram del subVI *FindWaveformIntegral.vi*

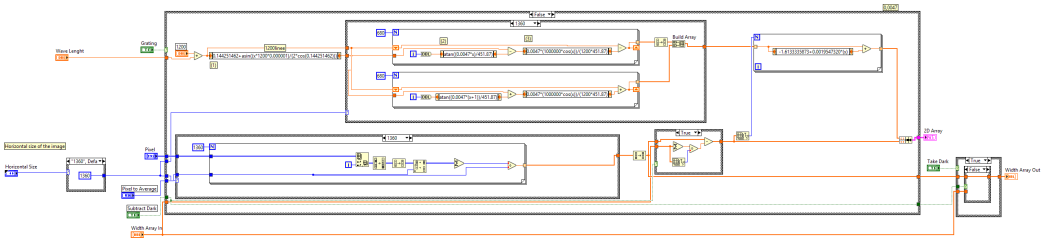

Figura 4.15: Block Diagram del subVI *CreateSpectrum.vi*

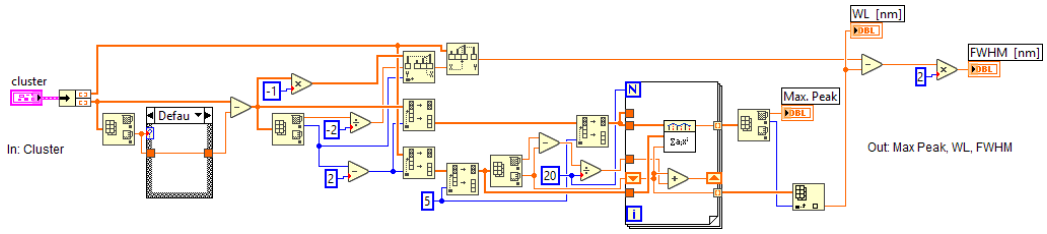

Figura 4.16: Block Diagram del subVI *FindMaxPeak&WL&FWHM.vi*

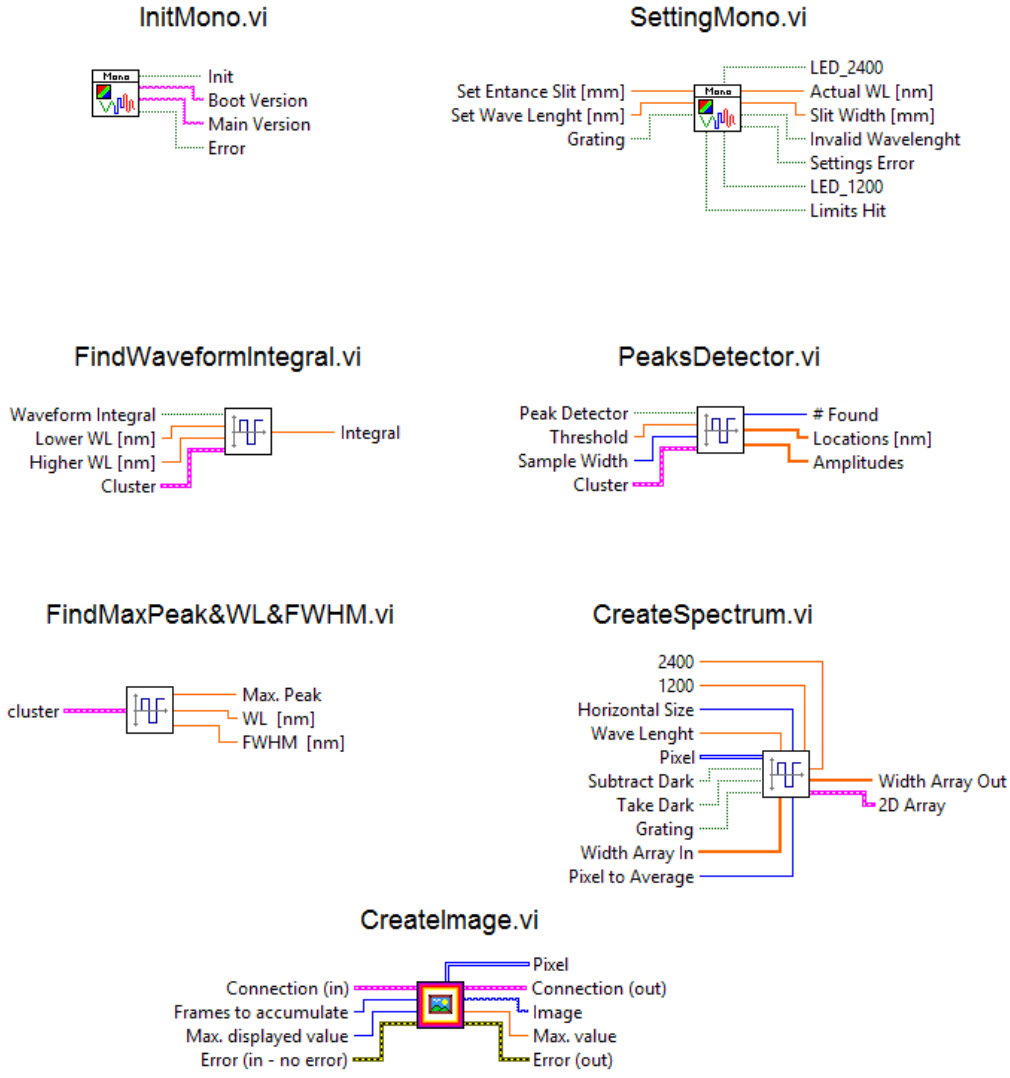

Figura 4.17: Icone dei subVI del progetto

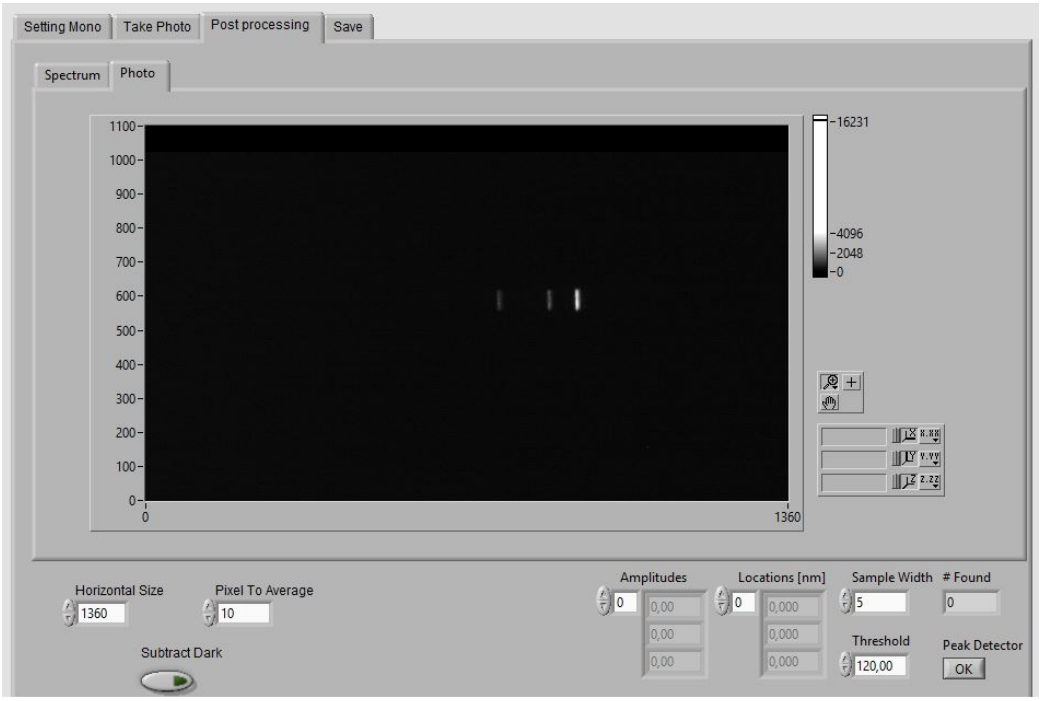

Figura 4.18: Immagine dello spettro del mercurio acquisita con il progetto realizzato

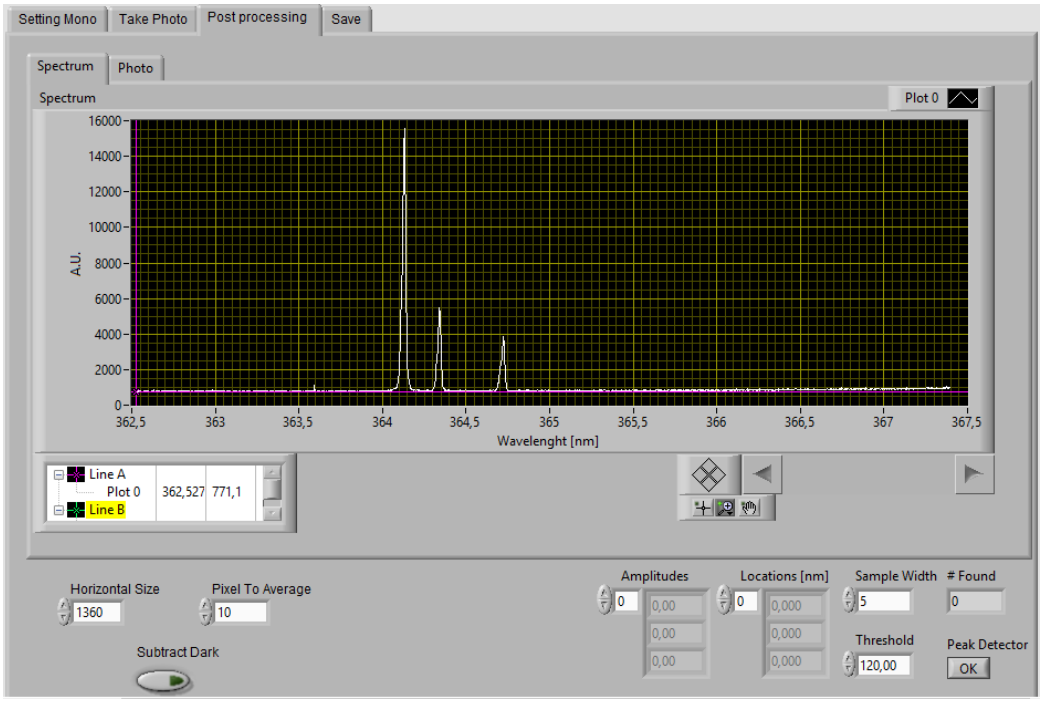

Figura 4.19: Spettro del mercurio processato a partire dall'immagine acquisita con il progetto realizzato

Capitolo 5

Conclusioni

È stato realizzato il codice per il progetto richiesto: interfaccia grafica di controllo per una fotocamera intensificata e un monocromatore in ambiente LabVIEW. Per poter realizzare il programma è stata molto utile la partecipazione ad un corso su tale linguaggio per avere una conoscenza accademica oltre che concreta; tale corso ha anche permesso di conseguire la certificazione base di *CLAD*, Certified LabVIEW Associate Developer.

5.1 Risultati ottenuti

Quasi tutte le specifiche richieste sono state soddisfatte. Il programma controlla correttamente entrambi gli strumenti e permette di salvare i dati acquisiti. Il monocromatore può essere settato correttamente controllando i dati prima di inviarli allo strumento, evitando così di danneggiare lo strumento. Per la fotocamera sono state riprese quasi tutte le opzioni presenti nel software proprietario. Nella fase di post processing dell'immagine è possibile visualizzare gli stessi calcoli presenti nel progetto preesistente.
La sottrazione del background deve ancora essere verificata in quanto la nuova fotocamera non dà la possibilità di acquisire un background, ed è necessario effettuare questa operazione manualmente. Rispetto al progetto preesistente è stato eliminato il comando di *Normalize Flat* in quanto considerato poco

rilevante al fine degli scopi di ricerca. Deve inoltre ancora essere realizzato un comando che permetta di visualizzare solo la parte centrale dell'immagine acquisita che è la porzione di interesse maggiore per quanto riguarda gli spettri.

5.2 Sviluppi futuri

Il programma può senza dubbio essere migliorato, sia dal punto di vista del front panel che del block diagram. Per quanto riguarda il front panel si tratta di una questione prettamente estetica, che collide anche con la versione di LabVIEW installata sul calcolatore dal momento che non in tutte le versione di LabVIEW sono presenti gli stessi controlli e indicatori. Il block diagram si può considerare organico e comprensibile, è comunque sempre possibile aggiungere nuove funzionalità al programma, inclusa quella di *Subtract Dark*.

Bibliografia

[1] *https://en.wikipedia.org/wiki/Monochromator*.

[2] Francois Countaceau. Mise en service d'une camera iccd. Relazione di tirocinio, Luglio 2013.

[3] Horiba Jobin Yvon. *LabVIEW Driver Supplement to the Spectometer Control Manual*.

[4] I.S.A. Jobin Yvon-Spex. *HR460 User Manual*, 1996.

[5] Nation Instruments Corporation. *http://www.ni.com/labview/i/*, 2017.

[6] Standford Computer Optics. *http://www.stanfordcomputeroptics.com/technology/iccd-system-overview.html*, 2013.

[7] Stanford Computer Optics, Inc. *LabVIEW API*.

[8] Stanford Computer Optics, Inc. *Operating Manual for intensified CCD (ICCD) digital video camera system*, 2014.

Ringraziamenti

Ringrazio di cuore la mia famiglia che mi è stata accanto in questi anni di studio con pazienza e sacrificio.
Ringrazio il prof. Gabriele Neretti per avermi permesso di realizzare questo progetto imparando un nuovo linguaggio. Lo ringrazio per la fiducia e per il clima di lavoro presente nel laboratorio.
Ringrazio i miei amici che mi hanno sopportata e supportata anche nei momenti di difficoltà.