Università di Pisa

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea Magistrale in Ingegneria Elettronica

TESI MAGISTRALE

Sviluppo di un sistema di test automatizzato per fotosensori per il calorimetro elettromagnetico dell'esperimento Mu2e di Fermilab

CANDIDATO:

Ergys Hajkaj

RELATORI:

Prof. Roberto Saletti

Dr. Simone Donati

Dr. Franco Spinella

Dr. Gianantonio Pezzullo

Anno Accademico 2013 – 2014

Introduzione

Il Fermilab, anche noto come FNAL (acronimo di **F**ermi **N**ational **A**ccelerator **L**aboratory), è un laboratorio di ricerca dedicato allo studio della fisica delle particelle elementari situato a Batavia, IL (USA), che deve il nome al celebre fisico italiano Enrico Fermi. Scienziati di tutto il mondo collaborano con il Fermilab, eseguendo ricerche pionieristiche nel campo della fisica delle particelle e sviluppando tecnologie innovative per la scienza e il sostegno dell'industria statunitense.

Tra i vari esperimenti proposti al Fermilab, io ho scelto di lavorare all'esperimento Mu2e, al quale collaborano circa 160 scienziati e ingegneri provenienti da 28 Università e Laboratori di tutto il mondo. In Italia la collaborazione Mu2e è composta dalle Sezioni di Frascati, Lecce e Pisa dell'INFN e dall'Università di Pisa.

L'esperimento Mu2e si propone di cercare la conversione coerente, senza neutrino prodotto, di muone in elettrone nel campo del nucleo. L'osservazione di questo processo costituirebbe una evidenza senza ambiguità dell'esistenza di fenomeni fisici non previsti dal Modello Standard. Ciò aprirebbe la strada a scenari di nuova fisica che vanno ben al di là di quelle che sono le nostre attuali conoscenze della fisica delle particelle elementari e quindi potrebbe essere meritevole di premio Nobel per la fisica, come è stata la scoperta del bosone di Higgs annunciata nell'estate dell'anno 2012.

La collaborazione italiana Mu2e sta lavorando alla realizzazione del calorimetro elettromagnetico, il quale svolge un ruolo cruciale nell'esperimento, cioè quello di misurare energia, posizione e tempo d'impatto delle particelle delle quali il sistema di tracciatura ha ricostruito la traiettoria, per eliminare i casi nei quali la traiettoria ricostruita non è dovuta ad una particella reale, ma ad una combinazione spuria di hit nel rivelatore.

Nel programma attuale l'inizio della presa dati dell'esperimento è prevista per il 2019. Mu2e dovrebbe raccogliere all'incirca 10^{18} muoni frenati su un sottile bersaglio di alluminio in un tempo di due/tre anni di presa dati. In questo modo si pensa di poter migliorare di un fattore 10^{4} la sensibilità raggiunta dagli esperimenti di precedente generazione per la ricerca della conversione coerente del muone in elettrone.

Ho svolto il mio lavoro di Tesi presso la sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare nel periodo compreso tra marzo 2014 e settembre 2014 e mi sono dedicato allo sviluppo di un sistema di test automatizzato per i fotosensori utilizzati nel calorimetro.

In particolare mi sono occupato dello sviluppo del firmware per il microcontrollore PIC24FJ12GA010 per la movimentazione e il controllo di precisione del piano motorizzato, sul quale sarà posizionata la matrice di fotosensori da testare. Ho, inoltre, progettato una scheda di interfaccia per la lettura dei due encoder della stazione di test ed ho sviluppato il firmware per il microcontrollore della scheda di interfaccia.

Dal momento che, il calorimetro Mu2e sarà realizzato mediante una matrice di cristalli letti da fotosensori, circa 3720 unità, lo sviluppo di una stazione di test automatizzata è un progetto di cruciale importanza per il successo dell'esperimento.

Nel seguito presento una breve sintesi del contenuto di ciascun capitolo della Tesi.

Il Capitolo 1 descrive il complesso degli acceleratori di Fermilab che fornisce il fascio di protoni all'esperimento Mu2e. Viene descritto, inoltre, l'apparato sperimentale Mu2e ed in maggior dettaglio il calorimetro elettromagnetico.

Il Capitolo 2 illustra il principio di funzionamento dei fotodiodi, in particolare vengono descritti i fotodiodi APD e i SiPM che sono i fotosensori che si stanno valutando per essere utilizzati nel calorimetro elettromagnetico di Mu2e.

Il Capitolo 3 descrive il progetto della stazione di test, in particolare il funzionamento del piano motorizzato e i dispositivi utilizzati, ossia i finecorsa, il motore passo – passo e l'encoder assoluto mono-giro.

Il Capitolo 4 illustra le caratteristiche del microcontrollore PIC24FJ128GA010 che gestisce l'intero sistema; vengono inoltre descritte in dettaglio le periferiche del microcontrollore utilizzate nel progetto.

Il Capitolo 5 descrive il progetto della scheda di interfaccia utilizzata per la lettura dei dati degli encoder di posizione, a partire dal disegno dello schema elettrico per arrivare alla realizzazione fisica della scheda e alla successiva progettazione del firmware installato nella scheda.

Il **Capitolo 6** descrive lo sviluppo del firmware per il controllo del piano motorizzato della stazione di test; in particolare vengono descritte le funzioni *init_motor()* e *go_to()* e lo sviluppo di un protocollo di comunicazione seriale tra il microcontrollore e il PC.

Il **Capitolo 7** descrive il test del firmware; i risultati mostrano che il firmware funziona correttamente e che gli spostamenti del piano motorizzato avvengono con la precisione desiderata.

Il **Capitolo 8** riporta le conclusioni del mio lavoro di Tesi e le prospettive per gli sviluppi futuri.

Indice

Capitolo 1

Il complesso degli acceleratori di Fermilab e l'apparato sperimentale Mu2e

La Collaborazione Mu2e è attualmente composta da circa 160 scienziati e ingegneri provenienti da 28 Università e Laboratori di tutto il mondo ed ha proposto un nuovo esperimento da realizzare al Fermi National Accelerator Laboratory negli USA. L'obiettivo dell'esperimento è la misura del rapporto $R_{\mu e}$ tra la probabilità di conversione coerente del muone in elettrone nel campo del nucleo, senza che sia prodotto alcun neutrino, e la probabilità di cattura ordinaria del muone nel nucleo:

$$R_{\mu e} = \frac{\mu^- + A(Z,N) \rightarrow e^- + A(Z,N)}{\mu^- + A(Z,N) \rightarrow \upsilon_\mu + A(Z-1,N)}$$

Questo processo di conversione si verifica con una violazione della conservazione del numero leptonico, un effetto che nei leptoni carichi non è mai stato osservato sperimentalmente. Attualmente, il miglior limite sperimentale sulla conversione del muone in elettrone $\mathrm{R_{\mu e}} < 7 \cdot 10^{-13}$ (con un livello di confidenza del 90 %) è stato ottenuto dall'esperimento SINDRUM II[21]. La figura1.1 mostra la storia dei limiti che fino ad oggi sono stati posti sui processi di violazione del numero leptonico di famiglia usando muoni[1]. L'esperimento Mu2e si inserisce in questo contesto con l'obbiettivo di migliorare di ben quattro ordini di grandezza il limite ad oggi posto da SINDRUM II, misurando $R_{\mu e}$ con una sensibilità di $6 \cdot 10^{-17}$ ad un livello di confidenza del 90%.

L'osservazione di questo processo costituirebbe un'evidenza senza ambiguità dell'esistenza di fenomeni fisici non previsti dal Modello Standard e potrebbe dare un contributo significativo ad individuare nuovi fenomeni che sono al di là della portata degli esperimenti effettuati attualmente al Large Hadron Collider del Cern di Ginevra.

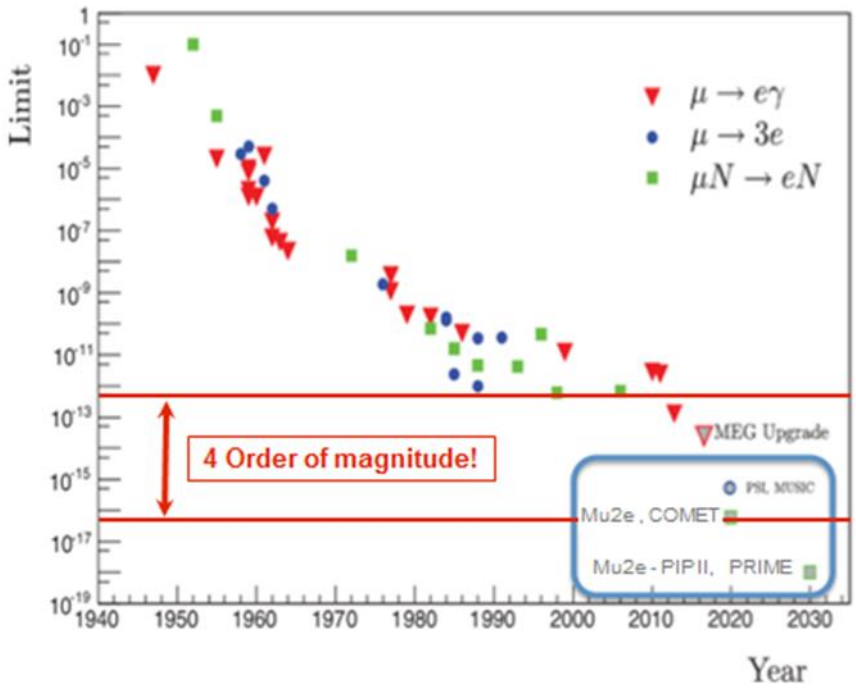

Figura 1.1: Limiti sperimentali sul processo di conversione del muone in elettrone ottenuti in passato.

La conversione del muone in elettrone nel campo del nucleo è un processo coerente che produce un elettrone mono-energetico in prossimità dell'energia a riposo di un muone che è circa 105 MeV. Questa segnatura distintiva ha numerosi vantaggi sperimentali, tra i quali, la possibilità di eliminare quasi completamente la contaminazione dovuta agli eventi di fondo.

L'apparato sperimentale di Mu2e è in avanzata fase di progettazione ed in attesa delle necessarie approvazioni del Department of Energy per la costruzione. L'inizio della presa dati è previsto per l'anno 2019.

Nel seguito di questo Capitolo descriveremo brevemente il complesso degli acceleratori di Fermilab necessari per la produzione di un fascio intenso di muoni, che viene fatto collidere sul bersaglio in alluminio, nel quale viene ricercato il processo di conversione. Descriveremo inoltre l'apparato sperimentale Mu2e.

1.1 Il complesso degli acceleratori di Fermilab

Il primo stadio di accelerazione è fornito da un generatore Cockcroft-Walton che ionizza atomi di idrogeno e successivamente li accelera mediante un campo

elettrostatico all'energia di 750 KeV[2]. Successivamente il Linear Accelerator (Linac) accelera gli ioni H^- all'energia di 400 MeV. Prima di entrare nello stadio di accelerazione successivo, gli ioni H^- attraversano fogli di carbonio e sono trasformati in ioni H^+, o protoni. Il terzo passo è rappresentato dal Booster, un anello di 468 m di circonferenza che usa magneti per mantenere il fascio di protoni sull'orbita circolare. I protoni provenienti dal Linac ruotano nel Booster circa 20,000 volte in 33 ms e sono sottoposti all'azione di campi elettrici che consentono di guadagnare energia ad ogni rivoluzione. Il fascio di protoni all'energia di 8 GeV e ad elevata intensità ($4 * 10^{12}$ protoni per pacchetto) è estratto dal Booster e può essere utilizzato per l'esperimento Mu2e. Questo fascio viene trasferito ad altri apparati del complesso di acceleratori di Fermilab, detti Recycler e Delivery Ring, per realizzare una struttura a pacchetti di "spessore" 100 ns, separati da intervalli di 1.7 μs.

Figura 1.2: Rappresentazione schematica delle parti rilevanti del complesso di acceleratori di Fermilab che produce il fascio di protoni utilizzato per l'esperimento Mu2e. Il fascio di protoni, che esce dal Booster, viene trasportato attraverso la linea MI–8 al Recycler Ring, dal quale esce attraverso la linea P1, per essere trasportato al Delivery Ring; infine dal Deliver Ring il fascio di protoni è inviato all'apparato sperimentale di Mu2e.

L'esperimento Mu2e riceve un fascio di pacchetti contenenti ciascuno in media circa $3.4 * 10^7$ protoni, per un totale di $1.8 * 10^{13}$ protoni/s. Il fascio di protoni colpisce un bersaglio nel quale vengono prodotti pioni e mediante il decadimento di questi, si ottengono muoni.

Mediante il sistema di trasporto i muoni raggiungono il bersaglio in alluminio nel quale si arrestano. La vita media del sistema prodotto è di circa 670 ns, che il rivelatore Mu2e attende prima di iniziare a registrare i dati, per i successivi circa 925 ns, prima che sia ricevuto il nuovo pacchetto di muoni sul bersaglio.

La figura 1.3 mostra la struttura temporale del fascio; in particolare, le due finestre temporali per il rallentamento dei muoni nel bersaglio e per la successiva presa dati dell'esperimento.

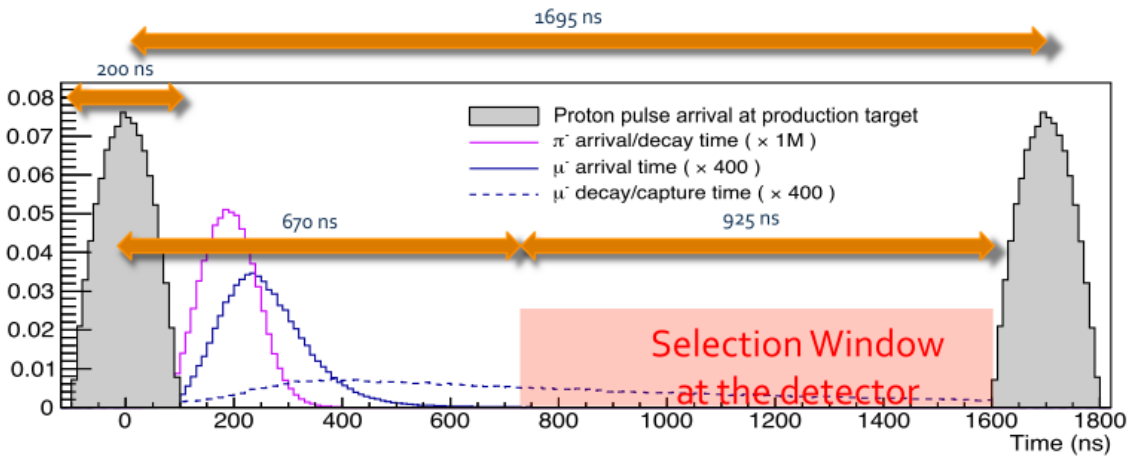

Figura 1.3: Il fascio utilizzato dall'esperimento Mu2e è costituito da una successione di pacchetti contenenti ciascuno in media circa $3.4 * 10^7$ protoni ogni 1695 ns. L'esperimento Mu2e, che riceve il fascio, aspetta circa 670 ns dopo la fine dell'impulso di protoni necessari per consentire il decadimento delle sorgenti di eventi di fondo. Questo lascia una finestra temporale di 925 ns per le misure del DIO (muon decay in orbit) e della conversione coerente del muone in elettrone.

1.2 L'apparato sperimentale Mu2e

L'apparato sperimentale Mu2e è costituito da 3 sistemi di magneti superconduttori, da bersagli e da rivelatori mostrati in figura 1.4[3]:

- Il magnete di produzione (**Production Solenoid**) riceve un fascio di protoni di energia 8 GeV che viene fatto collidere sul "Production Target" nel quale

vengono prodotti maggiormente pioni. Il campo magnetico all'interno del magnete varia da 2.5 T nella regione di ingresso del fascio a 4.6 T nella regione opposta, per creare quello che comunemente viene detto "specchio magnetico". Questa tecnica consente di aumentare la capacità di raccolta delle particelle cariche verso il magnete di trasporto. Il Production Target è un cilindro di tungsteno di 16 cm di lunghezza e 3 mm di raggio, raffreddato in acqua.

- Il magnete di trasporto (**Transport Solenoid**) è utilizzato per selezionare solo i pioni di carica negativa, con un certo intervallo di impulso, e a trasportarli nella direzione del Detector Solenoid. Il volume nel quale transita il fascio è mantenuto alla pressione di circa 10^{-4} Torr per ridurre la probabilità di interazione tra i muoni e gli atomi di gas residuo nel sistema. All'interno del magnete di trasporto questi pioni decadono in muoni negativi che poi raggiungono il Detector Solenoid.

- Il magnete dei rivelatori (**Detector Solenoid**) contiene all'inizio il bersaglio detto "Muon Stopping Target", cioè una serie di targhette di alluminio utilizzate per arrestare i muoni, poi, in serie, si trova il sistema di rivelazione. Quest'ultimo è a sua volta composto da un sistema di tracciatura ed un calorimetro elettromagnetico, necessari per ricostruire le traiettorie degli elettroni prodotti.

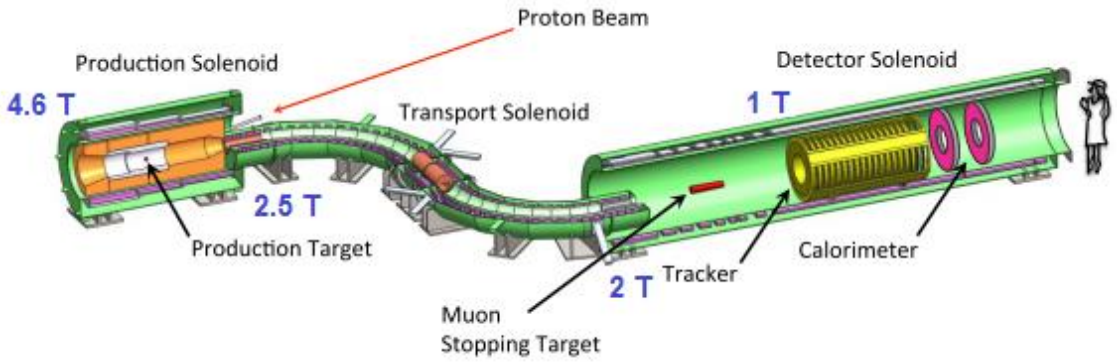

Figura 1.4: Rappresentazione schematica dell'apparato sperimentale Mu2e. Sono indicati i magneti nei quali è contenuto il bersaglio per la produzione del fascio di muoni, detti "Production Solenoid", i magneti per il trasporto del fascio di muoni, detti "Transport Solenoid", ed i magneti nei quali è contenuto il rivelatore, detti "Detector Solenoid".

Il Detector Solenoid contiene il bersaglio, il sistema di tracciatura ed un calorimetro elettromagnetico, necessari per ricostruire le traiettorie degli elettroni prodotti:

- **Bersaglio (Muon Stopping Target):** il bersaglio è costituito da dischi in alluminio centrati e posti ortogonalmente all'asse del solenoide. Il campo magnetico nel quale è immerso il bersaglio decresce da 2 T ad 1 T per creare, come detto in precedenza, nel caso del Production solenoid, uno specchio magnetico che riporti nella regione dei rivelatori le particelle emesse in direzione opposta. Le dimensioni del bersaglio sono state scelte per avere una adeguata capacità di assorbimento dei muoni del fascio, ma allo stesso tempo per non degradare il segnale di elettroni di conversione e non produrre una quantità eccessiva di elettroni di fondo. Anche il materiale, del quale sono costituiti i dischi, è stato scelto tenendo conto di diversi fattori, quali la stabilità chimica, la dipendenza dal numero atomico Z del materiale della probabilità di cattura del muone nel nucleo, la dipendenza dallo Z del materiale dello spettro in energia degli elettroni prodotti nella conversione dei muoni e nei processi di fondo, e la vita media dell'atomo muonico che, nel caso dell'alluminio è 864 ns ed è adeguata alla struttura temporale del fascio di Fermilab. La separazione tra due bunch di protoni del fascio è di 1700 ns e l'intervallo per la misura è compreso tra 800 ns e 1700 ns successivi alla iniezione del bunch sul bersaglio. Circa il 50 % delle catture nucleari si verifica nell'intervallo temporale scelto per la misura.

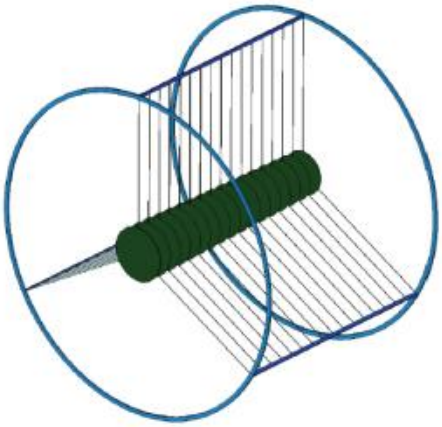

Figura 1.5: Il bersaglio nel quale sono arrestati i muoni è costituito da 17 dischi in alluminio di spessore 0.2 mm, distanziati 5 cm lungo l'asse del Detector Solenoid. Il raggio dei dischi decresce da 8.3 cm a monte a 6.53 a valle.

- **Sistema di tracciatura (Tracker):** il sistema di tracciatura è immerso in un campo magnetico uniforme di intensità 1 T ed è mantenuto nel vuoto alla pressione di 10^{-3} Torr. Questo rivelatore ricostruisce, con elevata efficienza, le traiettorie elicoidali degli elettroni e ne misura i parametri con ottima risoluzione. Il fenomeno dello scattering multiplo nel materiale, che costituisce il sistema di tracciatura, è responsabile della incertezza sulla misura dei parametri della traccia elicoidale. Un'ulteriore sorgente di incertezza è dovuta agli errori di riconoscimento delle tracce, dovuti per esempio all'erronea associazione dei punti colpiti nel rivelatore da particelle diverse.
  Il sistema di tracciatura è realizzato mediante la tecnologia degli straw drift tubes, con gli straw tubes posti nel piano trasverso all'asse del Detector Solenoid. Gli straw tubes hanno un diametro pari a 5 mm realizzati con uno strato di spessore 15 μm di Mylar metallizzato. All'interno dello straw tube è presente un filo conduttore di spessore 20 μm.

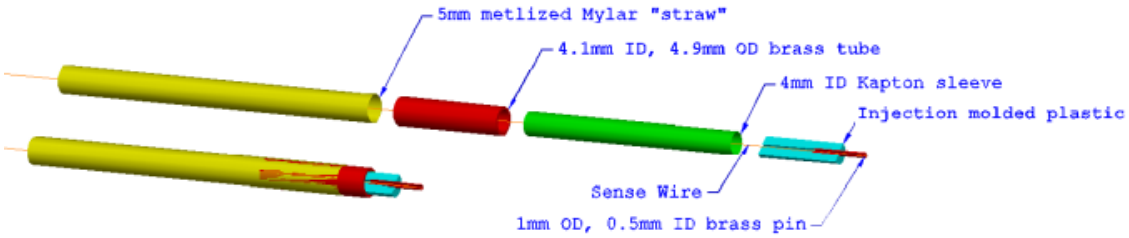

Figura 1.6: Immagine di uno straw tube.

Nel complesso, il sistema di tracciatura utilizza 20,000 straw tubes disposti in 18 stazioni di misura distribuite su una lunghezza di circa 3 metri. I piani sono costituiti di due strati di straw tubes, per massimizzare l'efficienza nella ricostruzione delle traiettorie e risolvere il problema dell'ambiguità "destra-sinistra". La distanza tra due straw tube vicini è di circa 1 mm, per semplificare l'assemblaggio ed offrire una tolleranza per l'espansione del tubo, dovuta alla pressione del gas interno allo straw tube. Il sistema di tracciatura è stato progettato per essere sensibile solamente agli elettroni con impulso trasverso superiore a 53 MeV/c, che costituiscono una frazione di circa solo il 3 % di tutti gli elettroni, sia di segnale che di fondo, prodotti dalla totalità di processi che avvengono nei bersagli di alluminio. Si ricorda che gli elettroni di segnale, dovuti alla conversione coerente del muone, hanno energia pari a circa 104.967

MeV, che rappresenta la massa a riposo del muone meno un piccolo contributo assorbito dal rinculo del nucleo allumino e dell'energia del legame nucleo – muone.

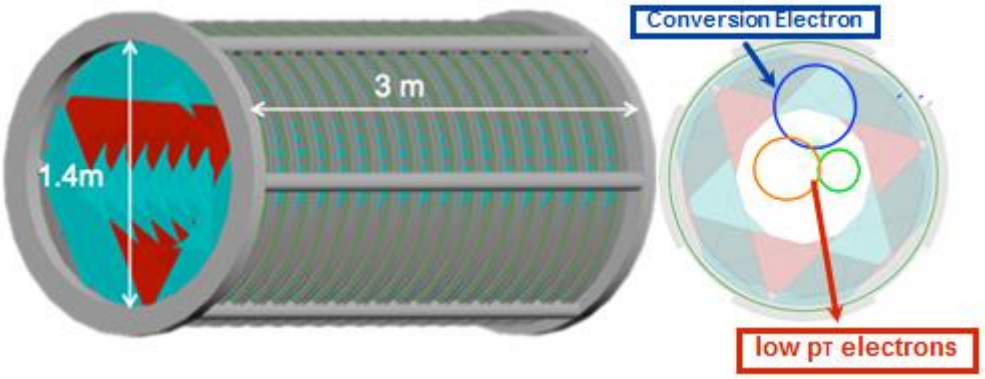

Figura 1.7: Sistema di tracciatura. Gli straw tubes sono orientati trasversalmente all'asse del Detector Solenoid. A sinistra è mostrato il sistema di tracciatura con diametro di 1.4 m e lunghezza di circa 3 m, a destra è mostrata la vista in sezione del sistema di tracciatura e sono rappresentate le traiettorie circolari di elettroni di conversione, ricostruiti nel sistema di tracciatura, e di elettroni che spiraleggiano all'interno del rivelatore e non sono ricostruiti.

- **Calorimetro (Calorimeter):** il calorimetro è stato disegnato per essere estremamente efficace nel rimuovere ogni sorgente di eventi di fondo non riconosciuta dal sistema di tracciatura. In questo senso, il calorimetro ha una funzione complementare a quella del sistema di tracciatura.

  Una sorgente di fondo particolarmente significativa è dovuta alle tracce false erroneamente ricostruite a causa dell'occupazione estremamente elevata nel sistema di tracciatura. Gli hit dovuti a particelle di bassa energia prodotte in processi di beckground possono essere erroneamente combinati e creare traiettorie false e consistenti con le traiettorie degli elettroni di conversione di più elevata energia. Di conseguenza, compito primario del calorimetro è fornire informazione complementare al sistema di tracciatura e consentire di eliminare il fondo dovuto ad errori di ricostruzione.

  Un'ulteriore sorgente di fondo è dovuta ai muoni cosmici, che superano l'apposito sistema di veto e interagiscono nel Detector Solenoid, creando una traccia che puo' erroneamente essere ricostruita dal tracciatore come un candidato elettrone da conversione. Queste sorgenti di fondo possono essere eliminate mediante un calorimetro con eccellente risoluzione temporale. Il progetto attuale prevede di realizzare un calorimetro a cristalli scintillanti. La

risoluzione energetica rappresenta un altro parametro importante del rivelatore; anche se la risoluzione in energia non è competitiva con quelle del sistema di tracciatura (~ 100 volte peggio), il calorimetro risulta comunque utile per fornire un'eccellente sistema di selezione online (trigger) indipendente e per ridurre il livello del fondo dell'esperimento.

Dal confronto con altri esperimenti, ci si aspetta che un calorimetro con una risoluzione in energia dell'ordine del 5 % sia adeguato per svolgere questa cruciale funzione. Nel caso di tracce dovute a particelle reali e non ad errori di ricostruzione, l'informazione del sistema di tracciatura e del calorimetro possono essere correlate in tempo. La risoluzione in tempo del calorimetro deve essere paragonabile a quella delle tracce estrapolate dal sistema di tracciatura, che si stima sia di circa 1 ns a 100 MeV. Una simile risoluzione in tempo può essere ottenuta mediante un calorimetro a cristalli scintillanti.

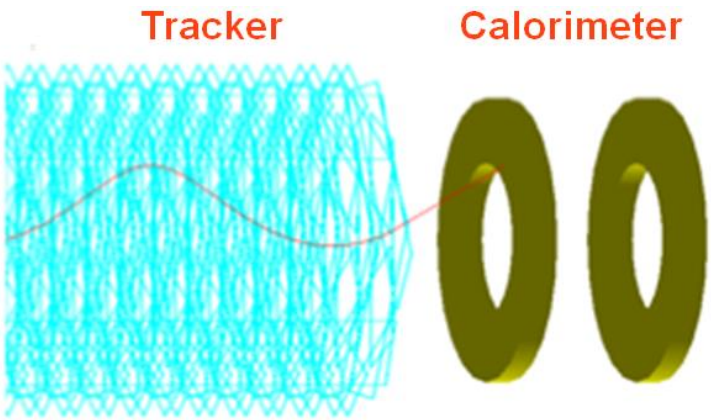

Figura 1.8: Rappresentazione schematica del sistema di tracciatura e del calorimetro, costituito da 2 dischi identici di cristalli scintillanti; la linea in rosso rappresenta la traiettoria elicoidale di un elettrone.

1.2.1 Il Calorimetro Elettromagnetico di Mu2e

La funzione principale del calorimetro di Mu2e è misurare energia, posizione e tempo di impatto delle particelle delle quali il sistema di tracciatura ha ricostruito la traiettoria, per eliminare i casi nei quali la traiettoria ricostruita non è dovuta ad una particella reale, ma ad una combinazione spuria di hit nel rivelatore. Inoltre, l'informazione del calorimetro deve poter essere utilmente combinata con quella fornita dal sistema di tracciatura, per distinguere gli elettroni dai muoni, e deve poter consentire di effettuare

una selezione in tempo reale degli eventi nei quali sono stati rivelati depositi significativi di energia.

È ben noto che, in termini di risoluzione, la misura di energia effettuata dal calorimetro non può essere competitiva con quella effettuata dal sistema di tracciatura, in ogni caso anche una misura con la sola risoluzione del 5 % a 100 MeV è estremamente utile per ridurre il livello di fondo dovuto alle combinazioni accidentali di hit, dovuti alle particelle di più bassa energia, che sono prodotte in misura estremamente abbondante.

Il calorimetro deve, inoltre, poter funzionare in condizioni di elevata e prolungata esposizione alla radiazione ionizzante. Questo richiede di determinare in quale misura la risposta dei cristalli risulta degradata in funzione del livello di esposizione alla radiazione, e di sviluppare elettronica di lettura mediante componenti resistenti alla radiazione.

Il calorimetro di Mu2e è stato concepito per confermare i candidati elettroni da conversione ricostruiti dal tracciatore e per fornire un'efficiente reiezione dei fondi generati dai raggi cosmici. Inoltre il calorimetro fornisce un sistema di trigger di alto livello totalmente indipendente dal sistema di tracciatura.

Per ottenere prestazioni adeguate in termini di risoluzione sulla misura di energia dell'elettrone incidente, di tempo e di posizione di impatto, è stato deciso di sviluppare un rivelatore con assorbimento totale e di utilizzare un mezzo attivo continuo ed omogeneo. Sino al momento della presentazione del Conceptual Design Report la collaborazione Mu2e aveva optato per l'ortosilicato di Lutezio-Ittrio, comunemente detto LYSO. Il recente aumento del prezzo di mercato di questo cristallo ha reso però questa scelta non più accessibile e per questo la collaborazione ha valutato altre scelte. Nello specifico due tipi di cristalli sono stati presi in considerazione: il Fluoruro di Bario (BaF_2) e lo ioduro di Cesio (CsI). Ad oggi il cristallo scelto dalla collaborazione è il BaF2, in quanto questo attualmente offre il miglior compromesso tra prestazioni e costo.

La figura 1.9 mostra la geometria del rivelatore, costituito da due dischi identici dove sono impilati i cristalli scintillanti e da due cornici circolari dietro ciascun disco per ospitare l'elettronica di lettura.

La geometria del calorimetro è stata studiata per massimizzare l'accettanza geometrica del rivelatore per gli elettroni di conversione. Questa consiste in due dischi identici con

raggio interno (esterno) pari a 35.1 (66.0) cm. La distanza fra i due dischi è pari a 75 cm, che rappresenta circa metà passo dell'elica prodotta in media dagli elettroni di conversione. Ciascun disco è composto da 930 cristalli di sezione esagonale con apotema di 1.65 cm e una lunghezza pari a 20 cm.

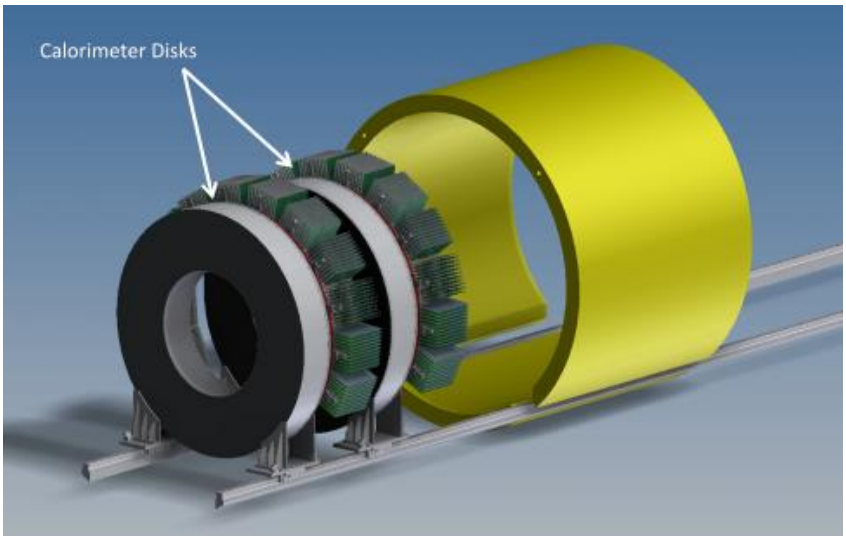

Figura 1.9: Geometria del calorimetro elettromagnetico di Mu2e. Si possono notare i due dischi composti da cristalli scintillanti e, sul retro, l'elettronica di lettura.

Ciascun cristallo è letto mediante due fotosensori a stato solido sia per massimizzare l'efficienza di rivelazione della luce prodotta dai cristalli, sia per garantire un sistema di lettura ridondante. È necessario utilizzare fotosensori di questo tipo per il fatto che il calorimetro è immerso in un campo magnetico di intensità pari a 1 T, che impedisce l'uso dei tradizionali fotomoltiplicatori.

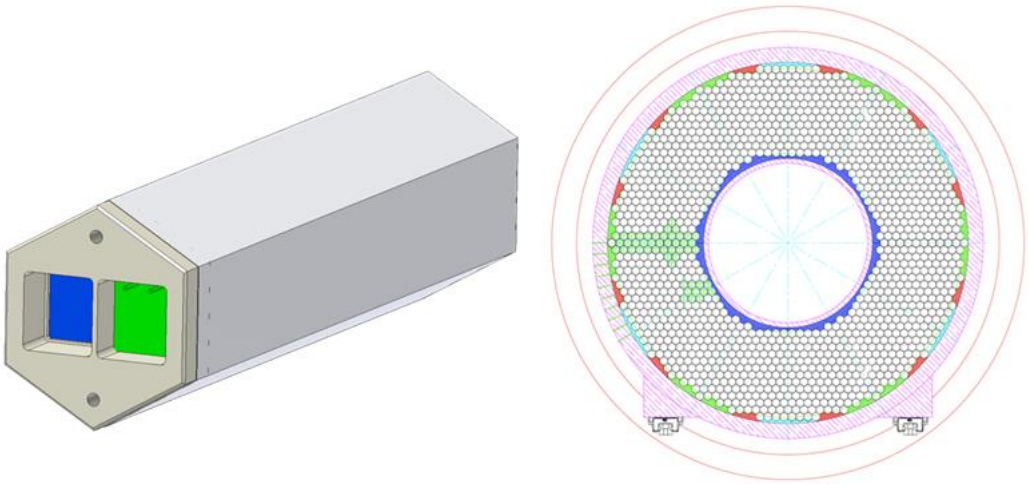

Figura 1.10: A sinistra immagine di un cristallo scintillante utilizzato nei due dischi del calorimetro elettromagnetico di Mu2e; a destra vista in sezione del disco composto da 930 cristalli scintillanti di sezione esagonale.

In termini di risoluzione, le prestazioni richieste al calorimetro sono le seguenti:

- risoluzione sulla misura di energia di circa il 5 % per elettroni di energia pari a 100 MeV, per confermare la misura di impulso degli elettroni effettuata dal sistema di tracciatura;

Numero di Dischi	2
Distanza tra i Dischi	700 mm
Raggio interno ed esterno del disco	351 mm, 660 mm
Materiale, densità, X_0, R_M	BaF_2, 4.9 g/cm^3, 2.0 cm, 3.0 cm
Geometria del cristallo	Esagonale
Area trasversa del cristallo	33 mm tra facce parallele
Lunghezza del cristallo	180 (200) mm
Numero totale dei cristalli	1860
Peso di 1 cristallo	1 Kg
Numero di APD/ cristallo	2
Dimensione traversa di 1 APD	10×10 mm^2
Numero totale di APD	3720

Tabella 1.1: Sommario dei principali parametri geometrici e costruttivi del calorimetro.

- risoluzione in tempo dell'ordine di 500 ps, per garantire che il deposito in energia nel calorimetro coincida temporalmente con il passaggio della particella associata nel sistema di tracciatura;
- risoluzione in posizione dell'ordine di 1 cm, per consentire di confrontare la posizione del deposito di energia con il punto di impatto sul calorimetro della traiettoria estrapolata della particella;

La struttura di supporto dei cristalli che garantisce una adeguata rigidità è realizzata mediante fibra di carbonio. Per quanto riguarda la calibrazione del calorimetro, si pensa di utilizzare un sistema di laser per la calibrazione relativa ed il monitor delle prestazioni dei fotosensori, ed una sorgente radioattiva per la determinazione della scala assoluta in energia e per la equalizzazione dei canali.

In figura 1.11 è riportato lo spettro della luce emessa dal cristallo di fluoruro di bario.

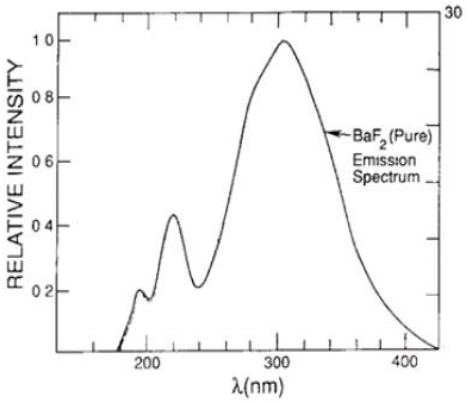

Figura 1.11: Spettro di emissione di luce di un cristallo di fluoruro di bario.

Si possono notare due picchi nello spettro, il principale alla lunghezza d'onda di circa 300 nm, il secondario a circa 220 nm. Le due componenti hanno diversi tempi di emissione, la componente più veloce è quella a 220 nm, ed ha un tempo di emissione di circa 900 ps, le componente meno veloce è quella a 300 nm ed ha un tempo di emissione di circa 650 ns. La presenza di una componente veloce risulta utile nella reiezione del fondo che, nelle condizioni di elevato affollamento attese nel rivelatore, è problematica.

Sono attualmente in studio varie soluzioni per i fotosensori da utilizzare per la lettura del segnale prodotto dai cristalli di fluoruro di bario. La presenza di un intenso campo magnetico impedisce di usare i tradizionali tubi fotomoltiplicatori (PMT), e costringe ad usare fotosensori a stato-solido, APD o SiPM. È stato creato un consorzio tra il California Institute of Technology (Caltech), il Jet Propulsion Laboratory (JPL) e la casa produttrice RMD per sviluppare delle modifiche agli APD che consentano di avere una efficienza quantica dell'ordine di 60 % per luce di lunghezza d'onda di 220 nm e dell'ordine di 0.1 % per luce di lunghezza d'onda sopra i 250 nm. Questa soluzione consentirebbe di avere un numero elevato di fotoelettroni prodotti, e di utilizzare unicamente la componente veloce della luce prodotta dal cristallo.

Capitolo 2

I Fotosensori

I fotosensori utilizzati per la lettura dei cristalli devono essere scelti e caratterizzati in maniera molto accurata, dal momento che il segnale elettrico da essi prodotto deve consentire di determinare l'energia, la posizione ed il tempo di impatto degli elettroni con la migliore risoluzione possibile.

Attualmente la collaborazione Mu2e sta valutando due opzioni per i fotosensori: gli Avalanche Photodiodes (APD), oppure i Silicon PhotoMultipliers (SiPM). Si tratta di fotodiodi con strutture differenti.

Descriviamo nel seguito il principio di funzionamento dei due dispositivi.

2.1 I Fotodiodi

Un fotodiodo è un particolare tipo di diodo che funziona come sensore ottico in grado cioè di riconoscere la luce di una determinata lunghezza d'onda e di trasformare questo evento in un segnale elettrico di corrente. Si tratta di un giunzione p-n particolare caratterizzata dalla zona p (zona drogata con atomi accettori) molto più drogata rispetto alla zona n (zona drogata con atomi donatori). La zona p si trova molto vicino alla struttura esterna del fotodiodo, che è ricoperta a sua volta da uno strato antiriflesso, sopra al quale è riposta una lente che rende perpendicolari i raggi luminosi incidenti sulla superficie.

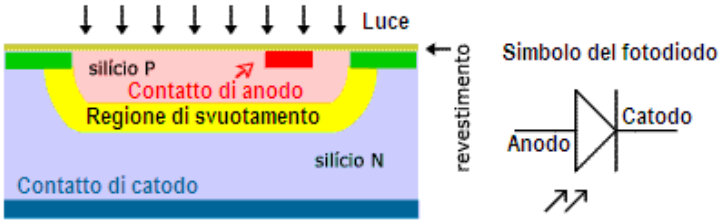

Figura 2.1: Sulla sinistra è mostrata la struttura base di un fotodiodo, mentre sulla destra è riportato il simbolo del fotodiodo.

Il fotodiodo, se polarizzato direttamente si comporta come un comune diodo. La corrente di conduzione segue, in prima approssimazione, la legge esponenziale del diodo. Non essendo tuttavia progettato per funzionare in polarizzazione diretta, il fotodiodo non ha una capacità di corrente tale da suggerirne un simile utilizzo, in quanto il surriscaldamento, dovuto al passaggio di corrente, potrebbe danneggiare gli elementi ottici.

Il fotodiodo funziona correttamente se polarizzato inversamente. In questo caso, il campo elettrico di built-in tende ad aumentare di intensità e favorisce l'ampliamento della zona di svuotamento (depletion region) e quindi della zona utile per l'assorbimento dei fotoni.

Il meccanismo di rivelazione su cui si basa il funzionamento dei fotodiodi è l'assorbimento dei fotoni. Nel momento in cui un fotone incide sulla superficie del fotodiodo, se la sua energia $\mathrm{E} = \mathrm{h}\nu$ è maggiore dell'energia del bandgap tra la banda di valenza e la banda di conduzione del dispositivo il fotone è assorbito, ciò causa la creazione di una coppia elettrone – lacuna libera.

Consideriamo la giunzione p-n polarizzata inversamente della figura 2.2[4].

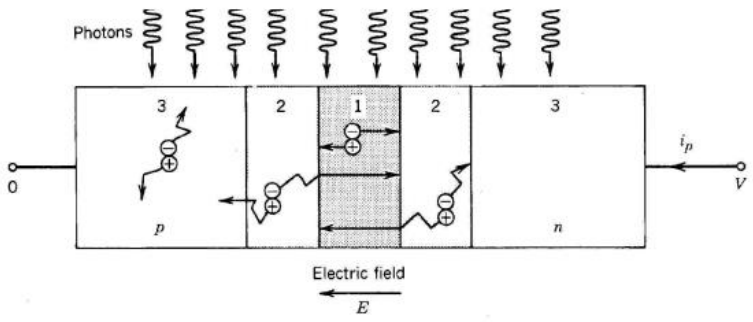

Figura 2.2: Schematizzazione di un fotodiodo, polarizzato inversamente, illuminato da fotoni.

Quando un fotone viene assorbito nel semiconduttore si ha la produzione di una coppia elettrone – lacuna. Lo sviluppo successivo dipende dalla regione del semiconduttore colpita dal fotone e nella quale è stata prodotta la coppia, figura 2.2:

- Fotone assorbito nella regione 1: La coppia di portatori generata viene separata dal campo elettrico presente nella zona di svuotamento. Elettrone e lacuna si dirigono in direzioni opposte; ciò comporta la generazione di una corrente sui

terminali del fotodiodo, diretta nella direzione dalla regione n alla regione p e proporzionale al numero di fotoni incidenti, cioè al numero di coppie elettrone – lacuna.

- **Fotone assorbito nella regione 2**: La coppia di portatori viene generata nelle vicinanze della zona di svuotamento. C'è quindi la possibilità che il portatore di carica entri nella zona di svuotamento per diffusione randomica. Se questo accade l'elettrone proveniente dalla regione p viene trasportato velocemente verso la regione n dal campo elettrico presente nella zona di svuotamento, producendo una corrente elettrica nel circuito. Analoghe considerazioni valgono per una lacuna proveniente dalla regione n.

- **Fotone assorbito nella regione 3:** Le coppie di portatori generate nelle zone più lontane alla regione di svuotamento non possono essere trasportate dal campo elettrico. Dopo una diffusione randomica vengono quindi riassorbite per ricombinazione e non producono nessuna corrente nel circuito esterno.

Un parametro molto importante per i fotodiodi è l'efficienza quantica (η), definita come la probabilità che un singolo fotone incidente sulla parte attiva del dispositivo generi una coppia di portatori elettrone – lacuna. Quando i fotoni incidenti sono molti, l'efficienza quantica è definita come il rapporto tra il numero di fotoni e il numero di portatori generati nel dispositivo, e può essere determinata in funzione del parametro η ($0 < \eta < 1$) con la seguente formula funzionale:

$$\eta = (1 - \mathrm{R})\zeta(1 - \mathrm{e}^{-\alpha x})$$

- Il termine (1 − R) rappresenta l'effetto di riflessione che i fotoni incidenti subiscono sulla superficie del dispositivo; la riflessione può essere ridotta utilizzando opportuni rivestimenti antiriflettenti.
- Il fattore ζ rappresenta la frazione di coppie elettrone – lacuna che contribuiscono effettivamente alla fotocorrente, evitando la ricombinazione col materiale superficiale.
- Il termine $(1 - \mathrm{e}^{-\alpha x})$ dipende dal coefficiente di assorbimento del materiale α. In figura 2.3 sono mostrati i valori dei coefficienti di assorbimento per vari semiconduttori. Il numero di fotoni assorbiti dipende dalla capacità del materiale di creare una coppia elettrone – lacuna nella zona svuotata.

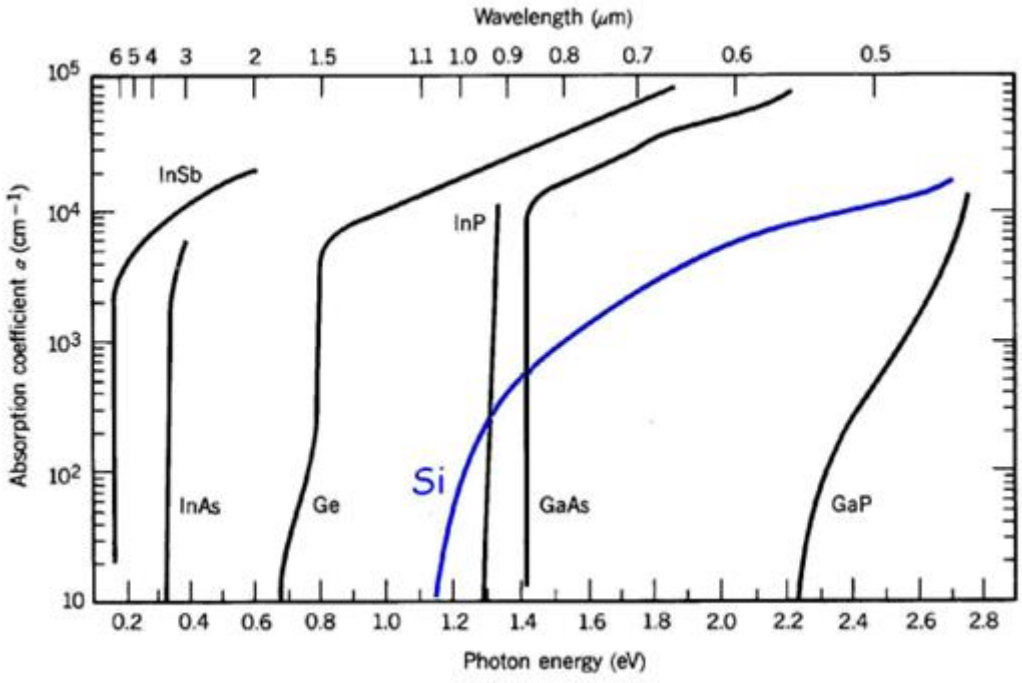

Figura 2.3: Coefficiente di assorbimento per vari semiconduttori al variare della lunghezza d'onda e dell'energia dei fotoni incidenti.

La relazione tra la polarizzazione e la corrente che scorre nel circuito esterno al fotodiodo può essere espressa come:

$$i = i_S \left[\exp\left(\frac{V}{V_T} \right) - 1 \right] - i_{ph}$$

che rappresenta la classica corrente di una giunzione p-n polarizzata inversamente con l'aggiunta della fotocorrente (i_{ph}).

$$i_{ph} = q * \frac{\text{Pot. ottica}}{h\nu} * \eta$$

Questa è determinata dall'assorbimento di un fotone, i_{ph}, ed è proporzionale al flusso fotonico, figura 2.4.

Nei fotodiodi la corrente inversa di saturazione è chiamata "dark current", cioè corrente di buio, perché è la corrente che scorre nel diodo quando non vengono assorbiti fotoni. Tale corrente di buio altro non è che il limite minimo della capacità di rivelazione del dispositivo.

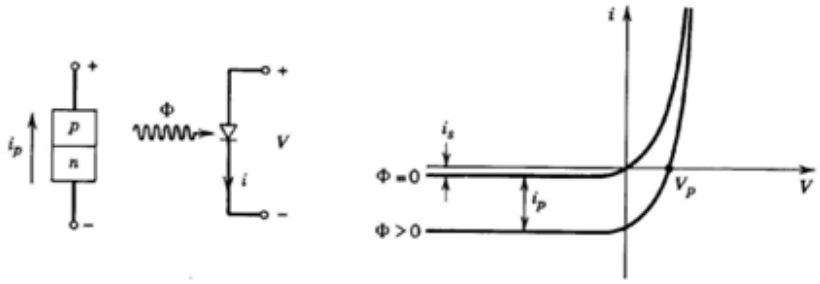

Figura 2.4: Caratteristica I – V di un generico fotodiodo.

Esistono vari tipi di fotodiodi:

- **Fotodiodi P-N**: Il fotodiodo p-n è stato la prima forma di fotodiodo sviluppato e utilizzato. È costituito da una giunzione p-n nella quale la zona di svuotamento risulta meno estesa della zona di assorbimento. Di conseguenza, l'efficienza quantica di questo dispositivo risulta abbastanza bassa.

- **Fotodiodi P-I-N**: I fotodiodi *p-i-n* sono costituiti da una regione intrinseca (regione i) compressa tra la zona p e la zona n. La zona i ha lo scopo di ampliare la regione di svuotamento e quindi rendere più estesa la superficie influenzata dal campo elettrico. Ovviamente, incrementando la regione di svuotamento, aumenta la zona attiva per la cattura della luce e quindi l'efficienza quantica del fotodiodo. Questi dispositivi sono caratterizzati da un guadagno interno unitario (G = 1) ed operano con tensioni di polarizzazione inversa molto al di sotto della tensione di breakdown.

- **Fotodiodi APD**: I fotodiodi a valanga operano con tensione di polarizzazione inversa elevata (vicino alla tensione di breakdown). Il campo elettrico è molto elevato, quindi le cariche che attraversano la regione di svuotamento hanno energia sufficiente per dissociare altre coppie elettrone – lacuna, con una moltiplicazione a valanga (effetto valanga). Gli APD sono quindi caratterizzati da un guadagno interno G = 30 – 300 e lavorano a tensioni di polarizzazione molto elevate, cioè tra 100 – 500 V.

- **Fotodiodi SiPM:** I fotodiodi SiPM sono costituiti da tanti piccoli fotodiodi a valanga collegati insieme in parallelo e operanti in Geiger mode. Rispetto agli APD, che lavorano in zona lineare, i SiPM hanno un guadagno interno molto più elevato ($G = 10^5 \div 10^7$) e quindi sono ideali per rivelazioni di luce a basso

livello. L'idea alla base di questo dispositivo è la rivelazione di eventi di fotoni singoli in sequenza.

2.2 I Fotodiodi APD (Avalanche Photodiodes)

I fotodiodi a valanga o APD sono un particolare tipo di fotodiodi che opera ad un'elevata tensione inversa di polarizzazione, vicino alla tensione di breakdown. Il funzionamento di un APD si basa sul fenomeno fisico di moltiplicazione a valanga dei portatori, prodotto dalla ionizzazione per impatto. Questo fenomeno si verifica quando il campo elettrico all'interno della regione di svuotamento è sufficientemente elevato da accelerare un portatore, prodotto per l'assorbimento di un fotone, e dotarlo di un'elevata energia cinetica. In questo modo il portatore, urtando gli altri portatori della banda di valenza, trasferisce a loro, parte della propria energia cinetica, permettendogli di "saltare" dalla banda di valenza a quella di conduzione e dando vita ad una nuova coppia elettrone – lacuna che si somma a quella di partenza. Anche questa coppia è accelerata dall'intenso campo elettrico ed acquista energia cinetica generando, sempre per impatto, nuove coppie di portatori. Questi si comporteranno allo stesso modo, dando origine così ad una valanga di portatori che procura un guadagno di corrente, figura 2.5.

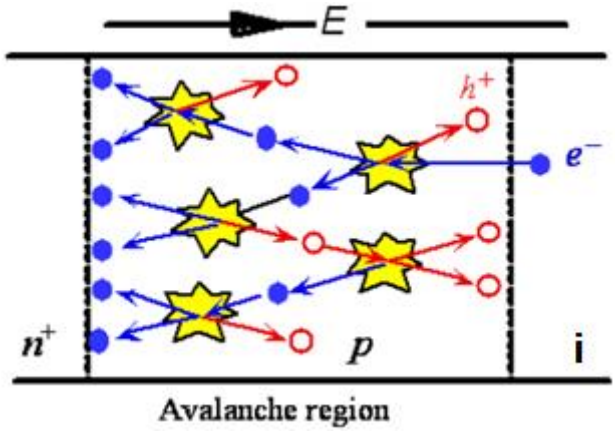

Figura 2.5: Modello di un fotodiodo a valanga e rappresentazione del processo di moltiplicazione a valanga delle coppie elettrone – lacuna.

La struttura di un APD è sostanzialmente costituita da 4 strati di semiconduttore drogati asimmetricamente, vedi figura 2.6[5]. La zona p rappresenta lo strato fondamentale del fotodiodo a valanga, detto strato moltiplicativo o di guadagno, nel quale si generano gli elettroni e le lacune secondarie per effetto valanga, mentre la zona i (intrinseca), come

nel fotodiodo P-I-N serve a tenere quasi costante il campo elettrico e ad aumentare l'efficienza quantica e a diminuire la capacità di giunzione.

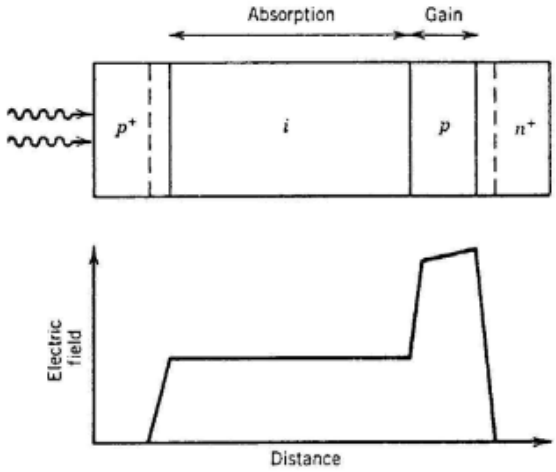

Figura 2.6: Sopra è mostrata la struttura di un modello di fotodiodo a valanga polarizzato inversamente e illuminato da fotoni. Si possono osservare la zona di assorbimento e la zona di guadagno. Sotto è mostrato l'andamento del campo elettrico nella struttura.

I due parametri di efficienza molto importanti per gli APD sono i seguenti:

- **Fattore moltiplicativo random (M):** il fattore M è il rapporto tra il numero di coppie elettrone – lacuna secondarie, generate per valanga, e il numero di copie elettrone – lacuna primarie, ottenute per fotoconduttività. Spesso il fattore M viene anche chiamato guadagno del fotodiodo (G).
- **Fattore di rumore in eccesso (F):** il fattore F è il rapporto tra la media del quadrato di M e il quadrato della media di M. Esso quantifica il rumore introdotto.

Esiste una relazione tra F ed M per cui:

$$\boldsymbol{F = M^x}$$

L'F(M) dipende dal materiale perché dipende dalle probabilità di fare valanga di elettroni e lacune[6]. Se le due specie hanno uguale probabilità di fare valanga allora il fattore di rumore in eccesso è massimo. Se invece c'è una sola delle due specie che ha probabilità di fare valanga, allora il fattore di rumore in eccesso è minimo e tendente all'unità. Per questi motivi gli APD sono generalmente realizzati con materiali che

permettono un solo tipo di portatore. I portatori che rendono massima la risposta temporale sono gli elettroni.

Questa condizione è irraggiungibile con materiali singoli, quindi viene creato un dispositivo composto da un'alternanza, di zone di accelerazione e di zone di assorbimento, in cui viene completamente inibita alle lacune la possibilità di produrre una valanga, possibilità promossa solo per gli elettroni.

La figura 2.7 mostra la struttura a bande all'equilibrio e di seguito quella in polarizzazione inversa di un APD. La zona 1 è composta da Arseniuro di Gallio e Indio (InGaAs) e la zona 2 da Fosfuro-Arseniuro di Gallio e Indio (InGaAsP), con aggiunta graduale di fosforo (P). La prima regione ha gap minore della seconda e all'aumentare della concentrazione di fosforo (P) il gap aumenta sempre di più. Polarizzando la struttura, le bande vengono inclinate. È necessario, quindi, polarizzare la struttura al di sopra di un certo valore minimo, prima di riuscire ad ottenere la valanga. Questo accade perché la pendenza della banda di conduzione della parte 2 risulta negativa. In questo modo, gli elettroni scendono attraverso il "pianerottolo" e acquistano una notevole energia cinetica; a questo punto può essere innescata la valanga per impatto. La zona in cui avviene la valanga è l'inizio della zona di assorbimento.

Al contrario le lacune devono risalire attraverso una struttura che non consente di sviluppare una valanga, dato che la pendenza della banda di valenza è bassa.

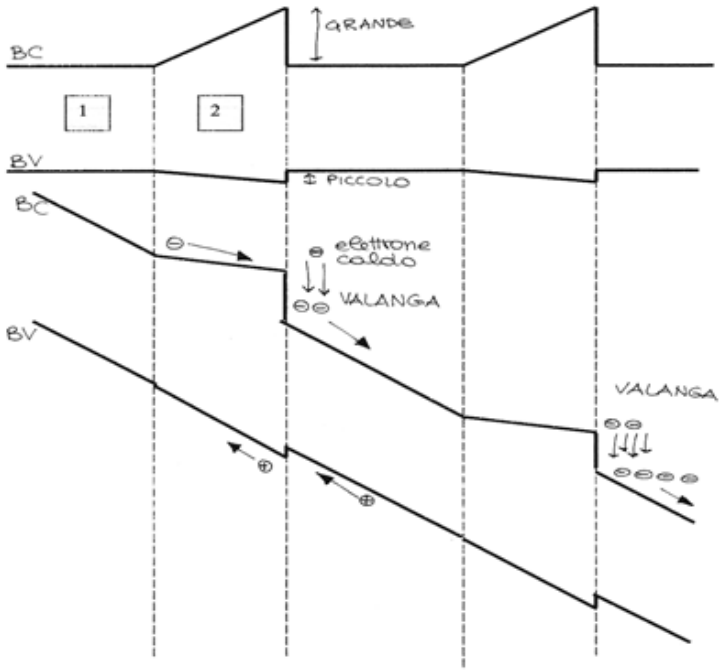

Figura 2.7: Struttura a bande di un APD sviluppata per favorire la valanga dei soli elettroni.

Il guadagno di corrente per un APD è dato dalla seguente formula:

$$G = \frac{1 - \rho}{\exp[-(1 - \rho)\alpha_e w] - \rho}$$

Dove:

- $\rho = \frac{\alpha_l}{\alpha_e}$ è il rapporto tra il coefficiente di ionizzazione per gli elettroni, α_e, e il coefficiente di ionizzazione per le lacune, α_l. Questi coefficienti rappresentano la probabilità per unità di lunghezza che avvenga un urto con successiva ionizzazione. La probabilità di ionizzazione è direttamente proporzionale all'intensità del campo elettrico presente nella zona di svuotamento ed è inversamente proporzionale alla temperatura.
- W è la larghezza della zona di svuotamento.

Dalla formula del guadagno si osserva che per $\rho = 0$ la moltiplicazione dei portatori avviene esponenzialmente mentre al crescere di ρ il guadagno cresce sempre meno, fino a tendere asintoticamente ad 1 per $\rho = \infty$.

Gli APD al silicio vengono fabbricati con $\rho \sim 0.006$ ed hanno ottime prestazioni per lunghezze d'onda della luce tra 0.4 e 0.7 μm.

2.3 I Fotodiodi SiPM (Silicon Photo Multiplier)

Un Silicon PhotoMultiplier (SiPM) è una matrice di APD identici operanti in modalità Geiger, ciascuno in serie ad una resistenza (resistore di quenching), e collegati in parallelo tra di loro nello stesso substrato di silicio. L'insieme costituito dalla serie APD – resistore di quenching è denominato, in letteratura, "pixel".

L'intento è coprire un'ampia area sensibile utilizzando tanti piccoli APD, aventi ognuno migliori prestazioni in termini di rumore e tempi di risoluzione.

Un APD operante in regime Geiger è un fotodiodo a valanga polarizzato ad una tensione inversa maggiore della tensione di breakdown. In questo modo, nella zona svuotata, si ha un campo elettrico tale da trasferire al singolo portatore un grandissimo valore d'energia cinetica, che rende sufficiente solo una singola coppia elettrone-

lacuna per innescare il processo di ionizzazione per impatto e la moltiplicazione a valanga dei portatori, figura 2.8.

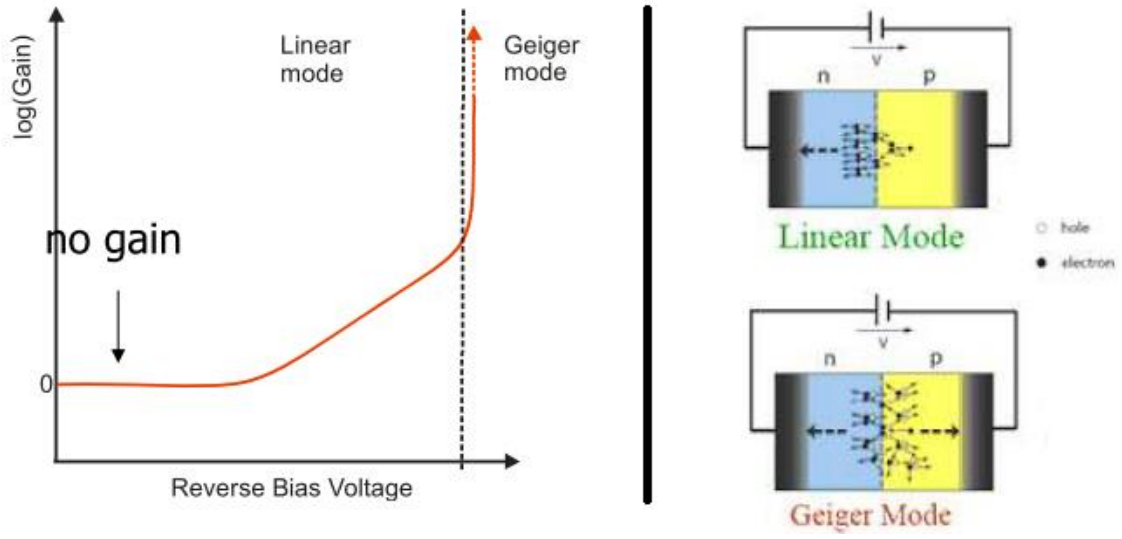

Figura 2.8: Curva caratteristica guadagno-tensione di polarizzazione per fotodiodi a valanga che funzionano in regime lineare o regime Geiger.

La figura 2.8 mostra la sostanziale differenza tra un fotodiodo operante in regime lineare e un fotodiodo operante in Geiger mode: l'elevata energia cinetica fornita dal campo elettrico fa sì che il processo di moltiplicazione si autosostenga e che il guadagno sia elevato ($\sim 10^6$, contro i 10^2 degli APD in zona lineare). Per questo motivo i SiPM rispetto agli APD possiedono un guadagno interno molto più elevato.

È chiaro che un fotodiodo in cui l'arrivo di un fotone innesca la valanga in regime Geiger, non può rivelare l'arrivo di un secondo fotone. È necessario dunque un processo che arresti la valanga, abbassando il campo elettrico ai capi della regione di svuotamento ad un valore tale da non permettere più la moltiplicazione per impatto dei portatori. La tensione inversa di polarizzazione torna sotto il valore di breakdown per un certo periodo, detto tempo di "hold-off". Durante questo intervallo di tempo, il dispositivo non può rivelare l'arrivo di nessun fotone. Per migliorare l'efficienza del fotodiodo è necessario che questo tempo cieco sia il più breve possibile. Ciò comporta l'utilizzo di un circuito di spegnimento detto, in inglese "quenching circuit". In figura 2.9 è mostrato il più semplice circuito di quenching: una resistenza in serie alla giunzione del fotodiodo.

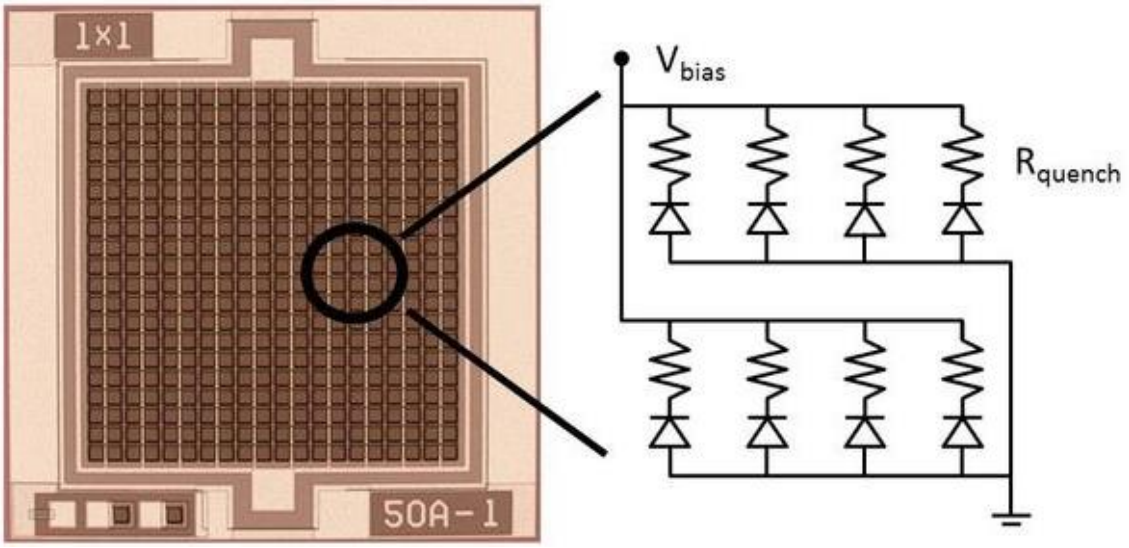

Figura 2.9: Vista dall'alto e schema equivalente di un Silicon Photomultiplier. Ogni fotodiodo ha una resistenza di quenching individuale. Fotodiodo e resistenza di quenching sono indicati come microcella o pixel.

La valanga crea una corrente che, producendo una caduta di tensione sulla resistenza di quenching, riporta localmente la giunzione ad una tensione inferiore a quella di breakdown. A questo punto la valanga termina a causa dell'assenza di un campo elettrico che dia energia sufficiente al suo innesco e la corrente prodotta dal fotodiodo è ridotta a zero. Viene quindi a mancare la caduta di tensione sulla resistenza di quenching e la tensione si riporta al valore di polarizzazione iniziale V_{bias}, maggiore della tensione di breakdown.

Un fotodiodo operante in Geiger mode può essere rappresentato mediante un circuito elettrico analogo a quello riportato in figura 2.10; un diodo avente la sua capacità di giunzione C_D e la sua resistenza di canale R_S, inserito in un circuito di polarizzazione inversa alla tensione V_{bias}.

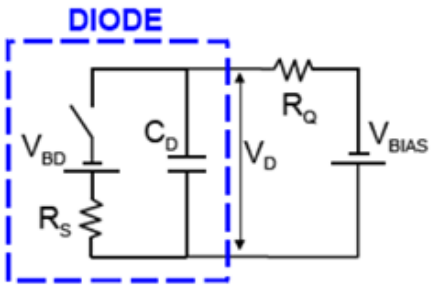

Figura 2.10: Modello elettronico di un fotodiodo operante in Geiger mode.

In figura 2.11 viene schematicamente mostrata la risposta del circuito al segnale impulsato; in assenza di segnale esterno ($t < t_0$) l'interruttore è aperto e la capacità C_D si carica tramite la resistenza di quenching R_S a V_{bias}. Quando un fotone colpisce la parete attiva del fotodiodo ($t = t_0$), l'interruttore si chiude, e la capacità C_D si scarica alla tensione V_{bd} tramite la resistenza R_S con $\tau_S = R_s \cdot C_D$, ($t = t_1$). Quando per effetto della caduta di tensione sulla resistenza di quenching la scarica si interrompe (interruttore riaperto) C_D si ricarica a V_{bd} tramite R_Q con $\tau_Q = R_Q \cdot C_D$ ($t > t_1$).

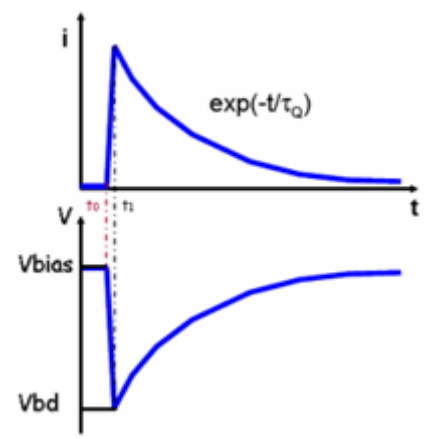

Figura 2.11: Andamento della corrente e della tensione per un impulso.

L'uscita è comune per tutti i pixel ed il segnale elettrico è costituito dalla somma delle cariche emesse dalle singole microcelle "accese" dall'assorbimento di un fotone. Se tutti i pixel sono identici ed emettono ciascuno la stessa quantità di carica quando assorbono un fotone, misurando la carica totale in uscita, si può risalire al numero di pixel accesi. Infatti, con l'ipotesi fatta d'uniformità di pixel, la carica prodotta in uscita è un multiplo, pressoché intero, della carica emessa dalla singola microcella accesa. Questa osservazione consente di determinare il numero di fotoni assorbiti, figura 2.12.

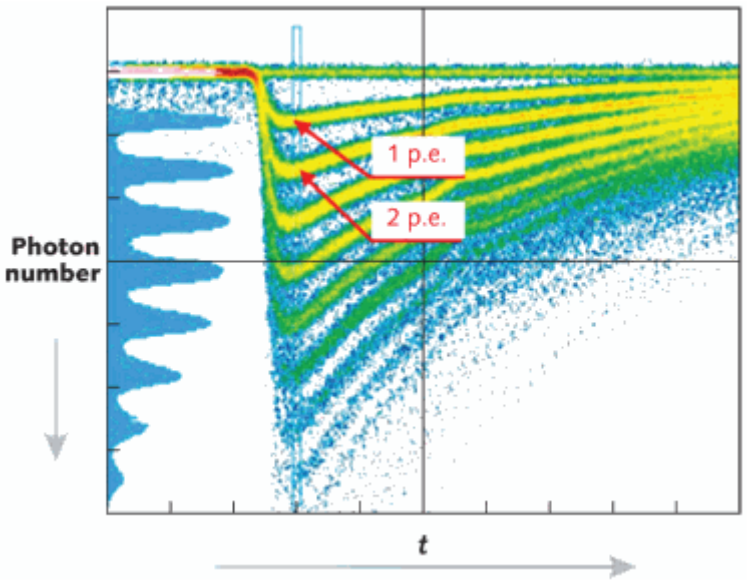

Figura 2.12: Segnale analogico di uscita di un SiPM in funzione del tempo visualizzato tramite oscilloscopio e corrispondente numero di fotoni.

Il valore tipico per il tempo di salita, dovuto alla scarica Geiger, si aggira intorno a 1 ns, mentre il tempo di discesa assume valori tipici di alcune decine di ns. Da questi valori si evince che, lo strumento è caratterizzato da un basso valore di tempo di recupero, che lo rende appropriato per applicazioni con alte frequenze di conteggi.

La tensione operativa dei SiPM è del 10 – 20 % superiore al valore di breakdown e non può essere incrementata a piacimento, poiché il rumore di fondo è proporzionale alla tensione di polarizzazione inversa delle giunzioni.

Il guadagno tipico di un fotomoltiplicatore al silicio, che dipende dalla temperatura e dalla tensione di polarizzazione inversa applicata, è comparabile a quello dei fotomoltiplicatori tradizionali e assume valori che variano tra $10^5 - 10^7$.

Modellando i pixel come un condensatore a facce piane e parallele, si può ricavare il guadagno del SiPM attraverso la seguente formula:

$$G = \frac{Q}{e} = \frac{(V_{bias} - V_{bd}) \cdot C_{pixel}}{e}$$

Il rumore in questo dispositivo è principalmente dovuto a due effetti[7]:

- Il cosiddetto “Dark Count Rate”, dovuto alla generazione spontanea, per effetto termico, di coppie elettrone – lacuna nella regione di svuotamento del fotodiodo, che innescano la scarica Geiger esattamente come un fotone incidente e non possono pertanto essere distinti da un segnale reale.
- Il cosiddetto “Optical Cross Talk”, cioè il fenomeno di crosstalk tra le celle adiacenti della matrice. Durante la scarica Geiger, infatti, vengono prodotti dei fotoni che possono migrare da una cella alla cella adiacente e provocare in essa una scarica spuria, vedi figura 2.13. I rimedi possibili sono due, la riduzione del guadagno oppure l’isolamento ottico dei singoli pixel. In questa seconda scelta è tuttavia necessario evitare la riduzione del Fill Factor per non intaccare ulteriormente l’efficienza dello strumento.

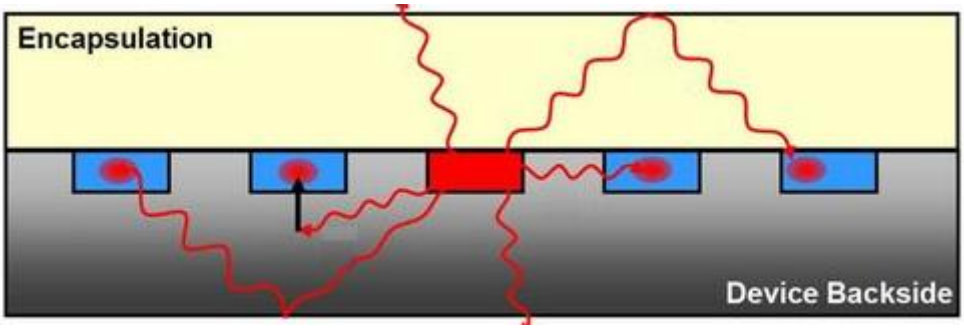

Figura 2.13: Schematizzazione del meccanismo di Optical Cross Talk.

L’efficienza dei SiPM viene ridotta dallo spazio morto tra i pixel che compongono la matrice: un fotone che colpisce questa zona, infatti, viene perso e non provoca alcuna reazione nello strumento. Il parametro fondamentale in questo senso è il fattore di riempimento o Fill Factor che è dato dal rapporto tra l’area attiva di un pixel e la sua area totale, compresi gli elementi circuitali. Maggiore è il fattore di riempimento, maggiore risulta l’efficienza del dispositivo.

Un parametro importante nella scelta di un SiPM è la probabilità di rivelazione di un fotone al variare della sua lunghezza d’onda. Questa è determinata dalla PDE (Photon Detection Efficency), prodotto di tre fattori:

$$PDE = Efficienza\ Quantica * Fattore\ di\ Riempimento * Probabilità\ di\ Valanga$$

Dove:

- L’efficienza quantica è la probabilità, che un singolo fotone incidente sulla parte attiva del dispositivo, generi una coppia di portatori elettrone – lacuna.
- Il fattore di riempimento, detto anche Fill Factor, è definito come il rapporto tra la dimensione effettiva dei pixel e la dimensione totale del SiPM.
- La probabilità di valanga è la probabilià che un fotone assorbito nella zona di svuotamento inneschi la valanga.

Capitolo 3

Il progetto della stazione automatizzata per il test dei fotosensori

Il calorimetro elettromagnetico Mu2e sarà costituito da due dischi contenenti cristalli scintillanti letti da fotosensori. Complessivamente, il numero di fotosensori necessari è circa 3720 unità. Dato l'elevato numero di fotosensori è necessario realizzare una stazione di test automatizzata che permetta di testarne una matrice di dimensione arbitraria in breve tempo. Naturalmente la scelta dei fotosensori, APD o SiPM, non richiede nessuna modifica della stazione di test. Entrambi i dispositivi avranno un'area attiva di circa $100\ mm^2$.

Ogni fotosensore deve essere testato in termini di funzionalità e di guadagno, in modo da poter equalizzare i vari canali quando l'esperimento sarà in fase di presa dati. In particolare le caratteristiche dei fotosensori da misurare sono:

- Il guadagno (G) in funzione della tensione V_{bias} e della temperatura.
- L'efficienza quantica (QE) in funzione della lunghezza d'onda λ del fotone incidente.
- Il fattore di rumore in eccesso (F) in funzione del guadagno.

Il sistema da noi ideato per effettuare queste misure è composto da: una sorgente di luce, un PCB con la matrice di fotosensori da testare, un piano motorizzato, un microcontrollore, un Keithley 6487 voltage source / picoammeter, un Picoamperometro, un sistema di distribuzione della luce ed un fotodiodo PIN. La figura 3.1 mostra la rappresentazione schematica della stazione di test.

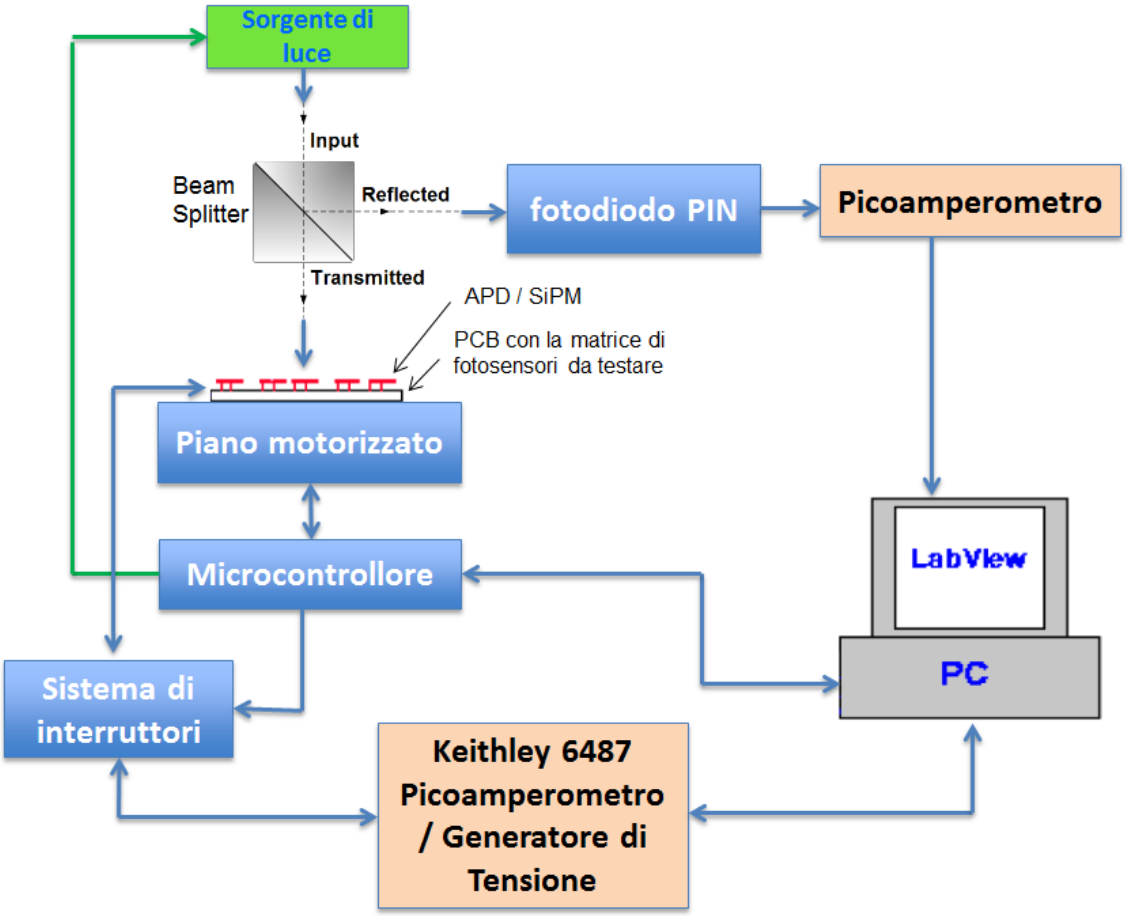

Figura 3.1: Schematizzazione della stazione di test.

Il circuito stampato sul quale è montata la matrice di fotosensori da testare, montati in zoccoli ZIF (zero insertion force), è supportato da un piano motorizzato che permette di posizionare con precisione (frazioni di millimetro) la matrice di fotosensori al di sotto della sorgente di luce utilizzata per stimolare il fotosensore. Per la movimentazione ed il controllo della posizione sono utilizzati motori passo – passo ed encoder assoluti, nonché sensori di fine corsa.

Nel progetto attuale, per stimolare i fotosensori la stazione di test dovrebbe utilizzare dei diodi “Read, Green and Blue” (RGB), ma si stanno valutando anche dei diodi laser. I diodi devono essere opportunamente collimati e comandati in ”pulse width modulation” per emettere la frequenza e la potenza necessari per creare un segnale adeguato nei fotosensori.

Il sistema di distribuzione della luce è costituito da un beam splitter che divide il fascio di luce in due parti uguali, una delle quali viene inviata ad un fotodiodo PIN calibrato

che ne misura la potenza, mentre l'altra parte è inviata ai fotosensori da testare. Per il trasporto della luce si utilizza una fibra ottica. In questo modo, è possibile ottenere un controllo ad anello della potenza emessa dal diodo.

Lo strumento di misura Keithley 6487 fornisce la tensione di polarizzazione applicata ai fotosensori, ma permette anche la misura della corrente in uscita da essi. Inoltre è previsto un sistema di interruttori, ad esempio un circuito a relè, che permette di commutare il Keithley su ogni fotosensore della matrice, vedi figura 3.2.

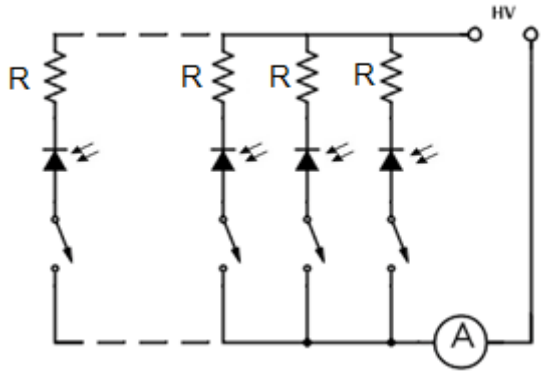

Figura 3.2: Schematizzazione del sistema di interruttori.

Il guadagno dei fotosensori dipende fortemente dalla temperatura, per cui la stazione di test dovrà essere posizionata all'interno di una camera termica ed equipaggiata con sensori di temperatura, utilizzati per il controllo ad anello. Inoltre, la stazione dovrà essere coperta interamente, per impedire che la luce esterna dell'ambiente possa giungere ai fotosensori falsandone la risposta.

La stazione di test verrà controllata mediante un PC attraverso un programma realizzato su LabView. Inoltre, un microcontrollore interfacciato con il PC è necessario per il controllo, del piano motorizzato, della sorgente di luce e del sistema di interruttori. Per ora la scelta è ricaduta sul microcontrollore PIC24FJ12GA010; in futuro verrà valutata la possibilità di utilizzare altri modelli, nel caso siano necessarie maggiori risorse in termini di logica.

Il grosso del lavoro svolto nella mia Tesi è stato lo sviluppo del firmware per il microcontrollore PIC24FJ12GA010 per la movimentazione e il controllo di precisione del piano motorizzato dove sarà posizionato la matrice di fotosensori da testare, e lo sviluppo di una scheda di interfaccia per la lettura dei dati degli encoder.

Per realizzare la stazione di test si è riutilizzata la parte meccanica di una vecchia stazione realizzata all'INFN di Pisa nell'anno 1999 per testare i moduli del rivelatore a microstrip in silicio Intermediate Silicon Layer dell'esperimento CDF di Fermilab, vedi figura 3.3.

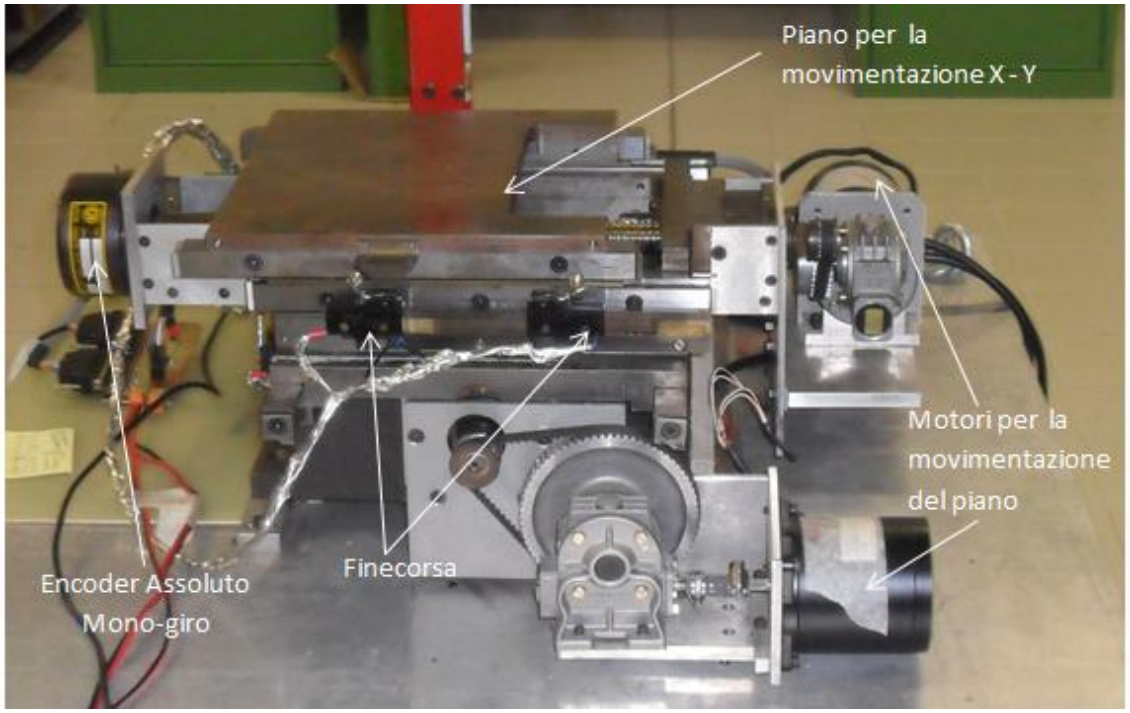

Figura 3.3: Struttura meccanica del piano motorizzato; è costituita da 2 motori passo-passo che consentono di muovere il piano lungo gli assi X ed Y mediante il meccanismo della vite senza fine, da 2 encoder assoluti mono-giro utilizzati per la misura accurata dello spostamento del piano e da 4 interruttori di finecorsa per definire i limiti del movimento del piano lungo gli assi.

Descriviamo nel seguito di questo Capitolo i componenti utilizzati per la parte di movimentazione della stazione di test: i finecorsa, i motori passo – passo e gli encoder.

3.1 I finecorsa

I quattro finecorsa utilizzati per definire l'intervallo di movimentazione del piano sono dei microinterruttori con leva a rotella serie BZ realizzati dalla casa costruttrice HONEYWELL S&C[8].

Figura 3.4: Immagine del microinterruttore usato come finecorsa.

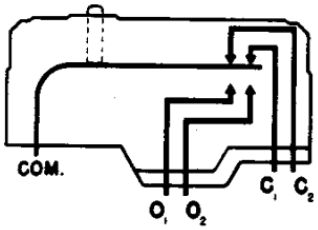

Figura 3.5: Schema dei contatti del microinterruttore usato come finecorsa.

Come si può osservare dalla figura 3.5, si tratta di un semplice deviatore con un contatto (COM, comune) che può essere connesso o con C (contatto normalmente chiuso) o con O (contatto normalmente aperto). Nel nostro caso, come si può vedere dalla figura 3.6, si utilizza una configurazione a pull-up, in cui il contatto comune è connesso a massa e il contatto normalmente chiuso è collegato, tramite una resistenza di $4.7\ K\Omega$, alla tensione di alimentazione. Il contatto normalmente aperto è lasciato flottante. Il segnale in uscita dal finecorsa viene prelevato dal contatto normalmente chiuso ed inviato in ingresso al microcontrollore, il quale gestisce la movimentazione della stazione di test. In questo modo quando la leva a rotella non viene schiacciata verso il basso l'uscita del finecorsa si trova a GND (0 logico), mentre quando la leva a rotella è schiacciata, il contatto si apre e l'uscita va a Vcc (1 logico).

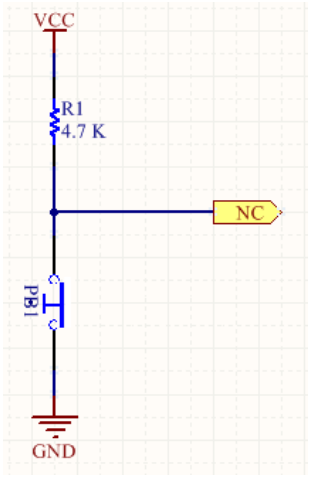

Figura 3.6: Schema elettrico della configurazione dei finecorsa.

3.2 Il motore passo – passo

I motori passo – passo detti anche *stepper motor*, consentono di fare ruotare l'albero del motore di un angolo fisso, detto *passo,* ad ogni impulso elettrico posto in ingresso.

I motori passo-passo, a differenza di tutti gli altri, hanno la caratteristica di mantenere fermo l'albero in una posizione di equilibrio: se alimentati si limitano infatti a bloccarsi in una ben precisa posizione angolare.

È possibile ottenere la rotazione dell'asse del motore solo indirettamente: occorre inviare al motore una serie di impulsi di corrente, secondo un'opportuna sequenza, in modo tale da far spostare, per scatti successivi, la posizione di equilibrio.

È così possibile far ruotare l'albero nella posizione e alla velocità voluta semplicemente contando gli impulsi ed impostando la loro frequenza, dato che le posizioni di equilibrio dell'albero sono determinate meccanicamente con estrema precisione.

Il principio di funzionamento del motore passo – passo è illustrato nella figura 3.7, dove viene mostrato un motore con risoluzione di 3.6°/step[9].

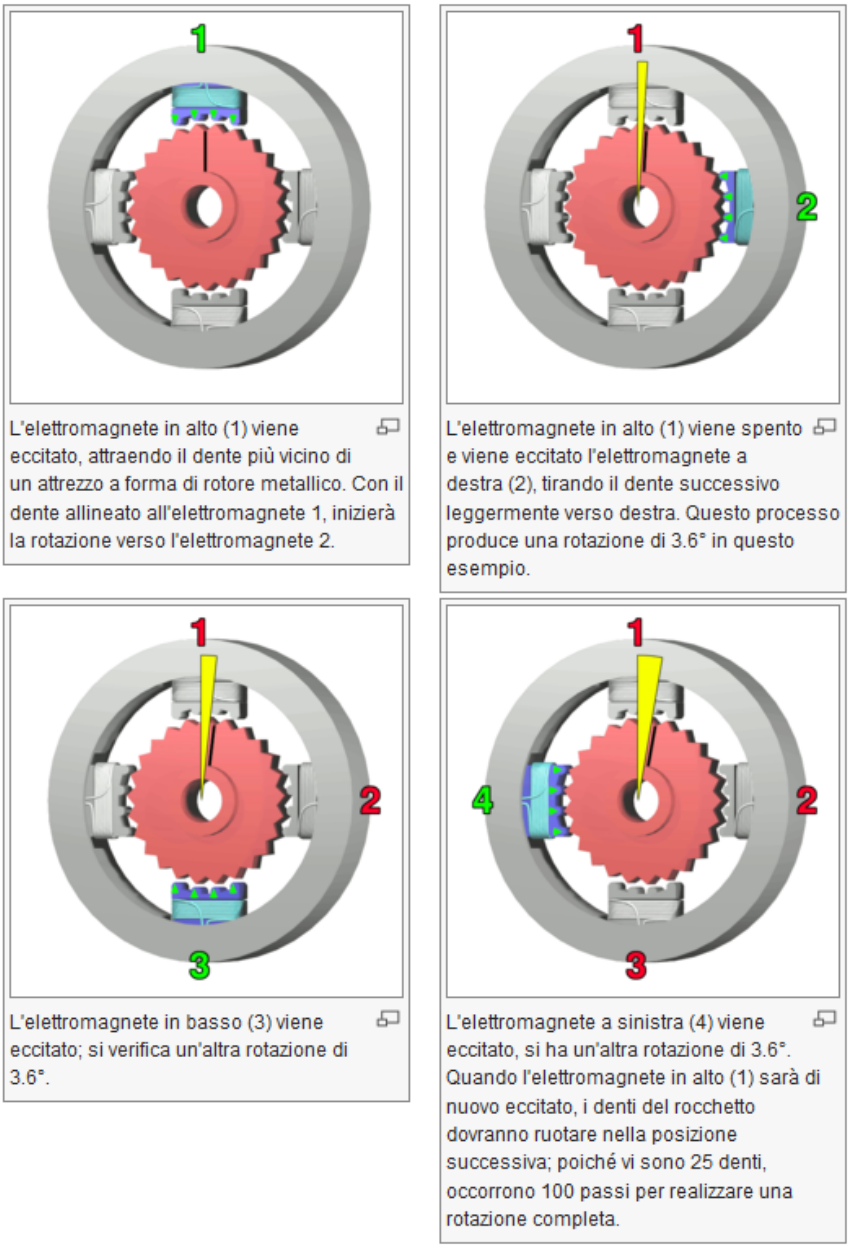

Figure 3.7: Principio di funzionamento di un motore passo – passo.

A differenza di altri tipi di motori, come ad esempio il diffusissimo motore a corrente continua, il motore passo-passo non modifica la velocità di rotazione in funzione del carico, ma la mantiene costante. Se lo sforzo richiesto al motore supera la coppia massima erogabile il motore semplicemente si ferma.

I motori passo-passo si dividono tradizionalmente in tre grandi gruppi: **motori a magnete permanente**, **motori a riluttanza variabile** e **motori ibridi** che combinano le

caratteristiche dei motori a magneti permanenti con quelle dei motori a riluttanza variabile; questi ultimi sono i migliori ed i maggiormente diffusi.

Un motore ibrido è costituito da un **rotore** e da uno **statore**.

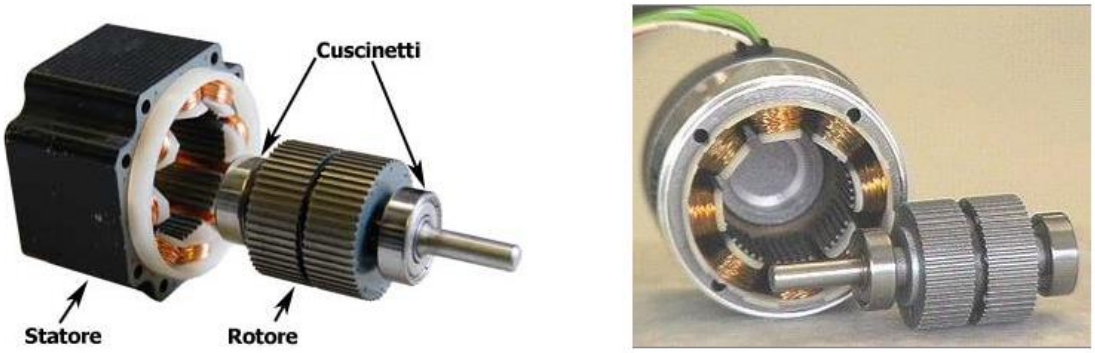

Figure 3.8: Costruzione di un motore passo – passo ibrido.

Il rotore è costituito da una coppia di ruote dentate, i “denti” sono comunemente detti coppette. Le coppette sono affiancate e solidali all'albero e sono costituite da un nucleo magnetico. Le due ruote sono permanentemente magnetizzate, una come NORD, l'altra come SUD e le coppette sono realizzate in materiale ferromagnetico.

I denti delle due ruote non sono allineati, fra essi vi è uno sfasamento esattamente pari ad 1/2 del passo dei denti: il dente di una delle due sezioni corrisponde quindi alla valle dell'altra. Nel rotore non sono presenti fili elettrici e quindi manca completamente ogni connessione elettrica tra la parte in movimento e quella fissa. In genere il rotore è montato su cuscinetti a sfera, anche nei modelli economici.

Lo statore appare come il classico insieme di avvolgimenti, costituito quindi, da diverse "espansioni polari". All'interno dello statore sono presenti piccoli denti che si affacciano esattamente a quelli del rotore o meglio, sono esattamente affacciati al rotore solo il gruppo di denti appartenenti ad una espansione polare e a quella opposta; le altre coppie sono sfalsate rispettivamente di 1/4, 1/2 e 3/4 del passo dei denti.

Avvolti intorno ai poli magnetici dello statore ci sono i fili che, opportunamente percorsi da corrente, generano il campo magnetico.

All'esterno sono presenti le alimentazioni dei vari avvolgimenti; in pratica le fasi possono essere avvolte secondo due schemi:

- Sono presenti due soli avvolgimenti, avvolti su più espansioni polari, e quindi all'esterno arrivano due sole coppie di fili: in questo caso si parla di **motori bipolari** in quanto la corrente deve percorrere le fasi nei due versi per creare gli opportuni campi magnetici.

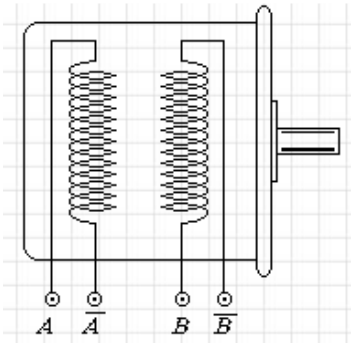

Figura 3.9: Schematizzazione di motore bipolare.

- Sono presenti quattro avvolgimenti avvolti a coppie, in antiparallelo, sulle espansioni polari; all'esterno arrivano almeno cinque fili (spesso sono infatti presenti delle connessioni interne al motore tra le varie fasi). Si parla in questo caso di **motori unipolari** in quanto la corrente nella singola fase ha sempre lo stesso verso. È possibile creare due campi magnetici opposti semplicemente scegliendo in quale dei fili debba passare la corrente. A parità di dimensioni i motori unipolari hanno una minor potenza specifica rispetto ai motori bipolari.

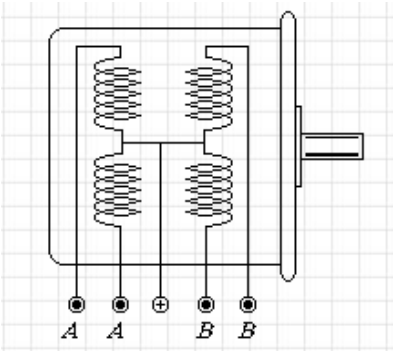

Figura 3.10: Schematizzazione di un motore unipolare.

Per ruotare i motori passo – passo necessitano di un'elettronica di controllo chiamata azionamento o drive.

I possibili modi per pilotare un motore passo – passo sono:

1. **Full Stepping (detto anche Single Stepping):** è il metodo più semplice in cui viene attivata una sola fase alla volta.

2. **Double Stepping:** sono attive contemporaneamente due fasi. Rispetto al caso precedente con questo metodo si raddoppia la coppia con lo svantaggio che non si genera un movimento ed è necessario il doppio della corrente.

3. **Half-Step:** questo può essere ottenuto combinando i due metodi precedenti, con l'effetto principale di raddoppiare il numero dei passi. Ad esempio, se un motore può ruotare di 1,8° per ogni passo, usando il metodo Half-Stepping è possibile spostare il motore di soli 0,9°/step.

4. **Microstepping:** questa è un'evoluzione del metodo half-step, così come è possibile ottenere un passo intermedio alimentando in contemporanea due fasi con corrente ridotta, è possibile ottenere una serie ampia a piacere di posizioni intermedie tra due step inviando due correnti di diverso modulo nelle due fasi adiacenti: il rotore si posiziona tanto più vicino ad una posizione di equilibrio tanto maggiore è la corrente nella fase corrispondente rispetto a quella dell'altra. In pratica le correnti assumono un andamento che tende ad approssimare quello sinusoidale, con uno sfasamento di 90° tra le due fasi. Ciò fa assomigliare il funzionamento del motore passo – passo a quello di un motore sincrono a due fasi.

Per regolare la corrente è necessario una notevole "intelligenza" dell'elettronica di controllo in quanto è necessario inviare invece di una semplice onda quadra un segnale sinusoidale variabile in fase e frequenza. Questa procedura consente di ottenere un enorme numero di posizioni dell'albero, ideale per le applicazioni di precisione.

I 2 motori utilizzati nella stazione di test sono motori passo – passo ibridi bipolari PH299 prodotti dalla casa produttrice Vextra ed hanno un passo di 1.8°/step. Di conseguenza, sono necessari $\frac{1.8°}{360°} = 200$ passi affinché l'albero motore compia un giro completo.

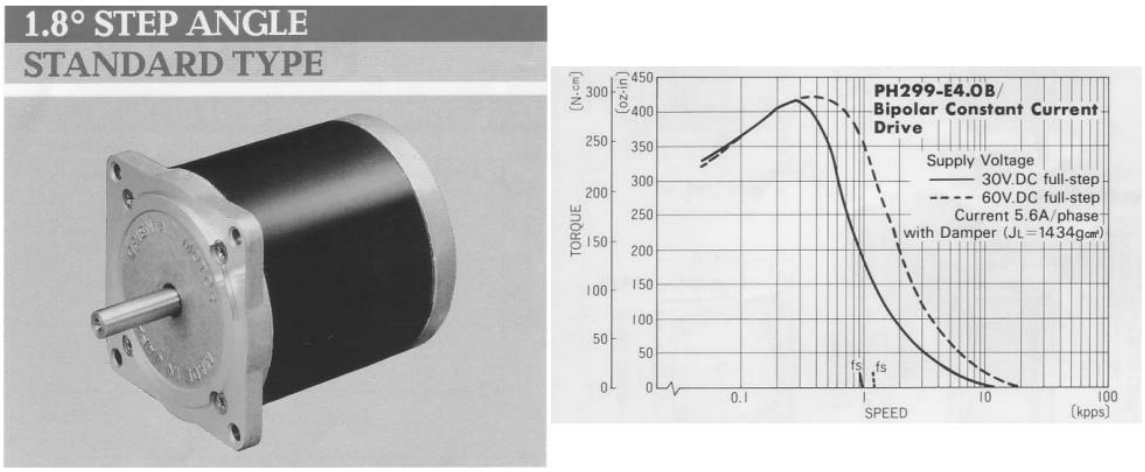

Figura 3.11: Sulla sinistra è riportato l'immagine del motore usato nella stazione di test; sulla destra è riportato il grafico coppia – velocita del motore.

I motori passo – passo sono controllati dal microcontrollore mediante un driver commerciale (modello G203V della casa produttrice Geckodrive), il quale pilota il motore in modalità microstepping con un fattore di moltiplicazione pari a 10. Ciò significa che, affinché l'albero motore compia un giro, è necessario inviare 2000 impulsi al driver. Questo consente un aumento della risoluzione del motore passo – passo (**0.18°/step**)[10].

PHYSICAL AND ELECTRICAL RATINGS

	Minimum	Maximum	Units
Supply Voltage	18	80	VDC
Motor Current	0	7	A
Power Dissipation	1	13	W
Temperature	0	70	°C
Humidity	0	95	%
Motor Inductance	1	50	mH
Input Frequency	0	250	kHz
Step Pulse "0" Time	2		uS
Step Pulse "1" Time	1		uS
Direction Setup (Before step rising edge)	200		nS
Direction Setup (Hold after pulse rising edge)	200		nS
Signal Voltage	3.3	5	VDC
Weight	3.6		oz

Figura 3.12: A sinistra foto del driver G203V, a destra la tabella dei suoi parametri elettrici.

3.3 Encoder assoluto mono-giro

L'encoder è un apparato elettromeccanico, cioè un trasduttore di posizione angolare che converte la posizione angolare del suo asse rotante in un segnale elettrico digitale.

Collegato ad opportuni circuiti elettronici e con appropriate connessioni meccaniche, l'encoder consente di misurare spostamenti angolari, movimenti rettilinei e circolari nonché velocità di rotazione ed accelerazioni[11].

Esistono varie tecniche per rivelare un movimento angolare: capacitiva, induttiva, potenziometrica e foto elettrica. I trasduttori sugli encoder impiegano tutti il rilevamento fotoelettrico.

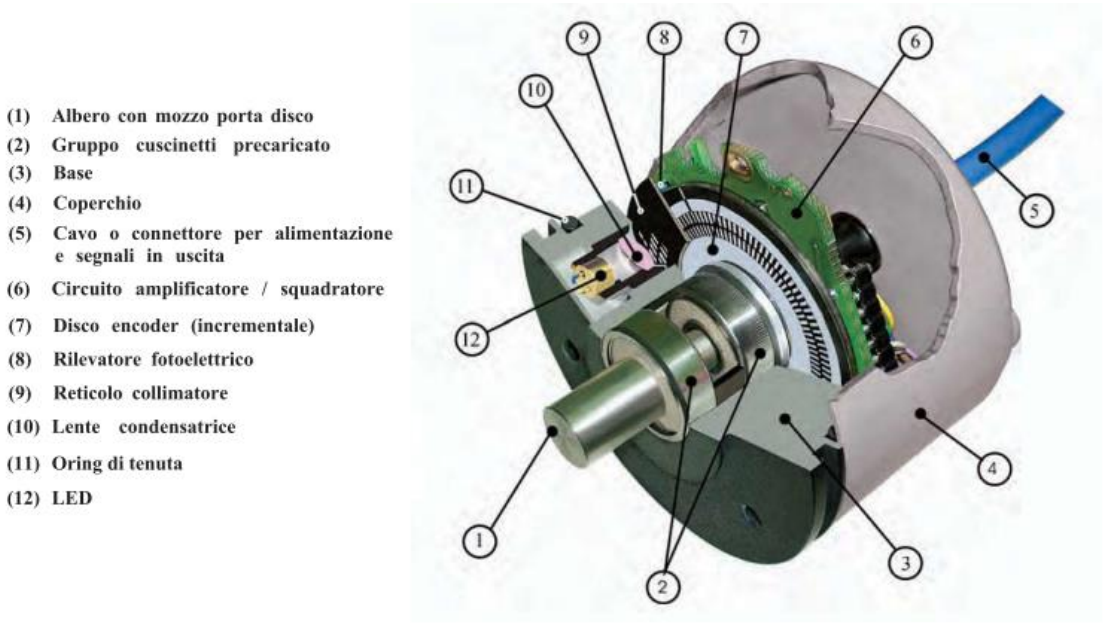

Figura 3.13: Costruzione di un encoder. Un fascio di luce collimato è usato per illuminare i reticoli radiali, quello fisso e quello mobile (disco). La luce che riesce a passare attraverso i due reticoli è raccolta da un rivelatore fotoelettrico posto immediatamente sopra il disco.

Gli encoder possono essere di due tipi:

1. **Incrementali:** in questo caso i segnali in uscita sono proporzionali in modo incrementale allo spostamento effettuato.
2. **Assoluti:** in questo caso ad ogni posizione dell'albero corrisponde un valore ben preciso.

Il principio di funzionamento di un encoder assoluto è molto simile a quello di un encoder incrementale nel quale un disco rotante, con zone trasparenti ed opache, interrompe un fascio di luce acquisito da fotoricevitori i quali trasformano gli impulsi luminosi in impulsi elettrici che vengono trattati e trasmessi dall'elettronica in uscita.

Rispetto agli encoder incrementali, gli encoder assoluti presentano importanti differenze dal punto di vista funzionale. Infatti mentre negli encoder incrementali la posizione è determinata dal conteggio del numero degli impulsi rispetto alla traccia di zero, negli encoder assoluti la posizione è determinata mediante la lettura del codice di uscita, il quale è unico per ciascuna delle posizioni all'interno del giro.

Di conseguenza gli encoder assoluti non perdono la posizione reale quando viene tolta l'alimentazione (anche in caso di spostamenti) e ad una successiva accensione (grazie alla codifica diretta sul disco) la posizione è aggiornata e disponibile senza dover eseguire, come per gli encoder incrementali, la ricerca del punto di zero.

Gli encoder assoluti hanno un disco codificato con più piste, le quali sono lette simultaneamente e danno un'uscita parallela in codice (Gray, binario, BCD, ecc.) per ogni posizione angolare.

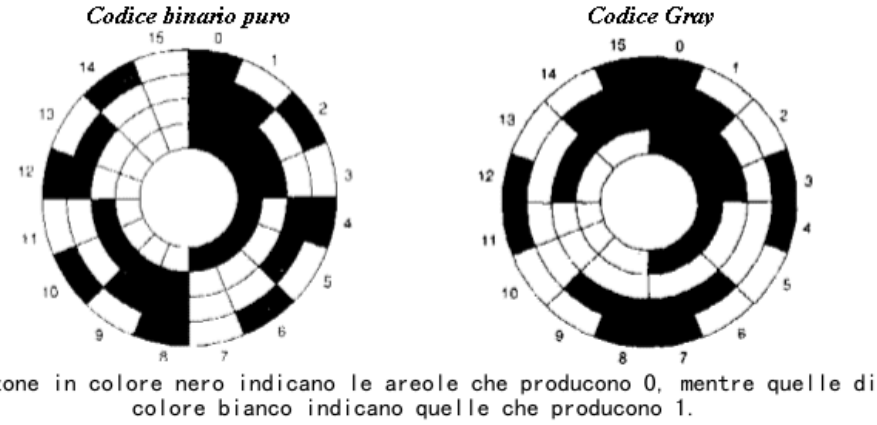

Figura 3.14: A sinistra è riportato un esempio di disco con codifica binaria, a destra invece è riportato un esempio di disco con codifica Gray.

Il disco è diviso in n (numero di bit) corone circolari e in $2 \cdot n$ spicchi. Ogni settore (spicchio) ha n areole che a seconda se opacizzate o trasparenti corrispondono a 1

logico o 0 logico. Ogni areola ha quindi il valore di un bit. Il bit più significativo è quello della corona più interna.

Per evitare errori di lettura invece del codice binario puro vengono utilizzati altri codici, tra i quali il più importante è il codice Gray. Nel codice Gray il passaggio da un numero al successivo avviene sempre variando un'unica cifra binaria, evitando così che nel passaggio tra la lettura di un numero e del successivo possano aversi letture casuali, assicurando cosi una elevata sicurezza e affidabilità per quanto riguarda la generazione e la decodificazione del codice.

Negli encoder assoluti si utilizza un fotoemettitore e un corrispondente fotorivelatore per ogni corona circolare del disco[12].

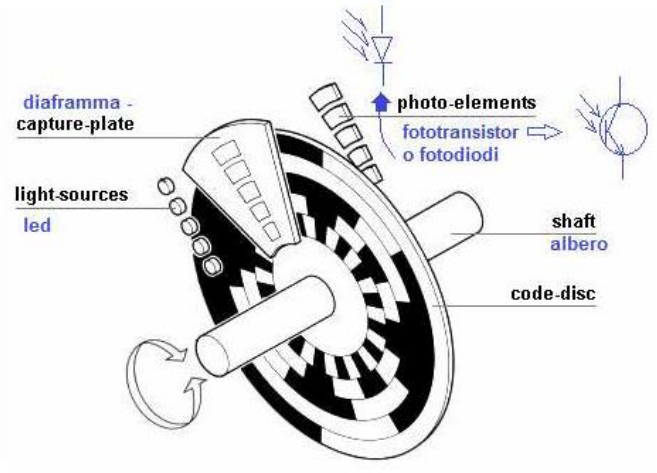

Figura 3.15: Principio di funzionamento di un encoder assoluto.

L'encoder assoluto ha il vantaggio di dare informazioni che non vengono perdute in caso di mancanza di alimentazione, ma richiede una particolare cura nella collimazione dello zero logico con lo zero macchina, è più costoso di quello incrementale e non consente di effettuare delle misure della velocità.

Gli encoder assoluti mono-giro sono realizzati per impieghi industriali e professionali. La meccanica realizzata interamente in metallo offre una buona tenuta agli agenti esterni.

Esistono anche encoder assoluti multi-giro che presentano fra un disco e l'altro un riduttore con rapporto uguale alle divisioni del disco.

Gli encoder per la loro vastissima gamma di modelli, qualità e robustezza, sono validamente applicati in tutto il mondo su: controlli di processo industriale, robot industriali, macchine utensili, strumenti di misura, plotters, divisori, laminatoi, macchine per lamiera, bilance e bilici, antenne e telescopi, macchine per la lavorazione del vetro, marmo, cemento, legno, impianti ecologici, macchine tessili, conciarie, gru, carri ponte, presse, macchine da stampa, imballaggio, ecc.

I 2 encoder montati sulla stazione di test per il controllo di precisione del movimento del piano sono 2 encoder assoluti mono-giro della casa produttrice ELCIS (modello LA390-G4096-5-CM5-R) con risoluzione di 12 bit in grado quindi di misurare uno spostamento angolare minimo di $\frac{360°}{4096} = 0{,}0879°$.

Si tratta di encoder con interfaccia di uscita parallela NPN open collector con codice Gray, alimentati alla tensione di 5 V con un assorbimento massimo di corrente di 200 mA e frequenza massima di funzionamento di 25 kHz.

Capitolo 4

Il microcontrollore PIC24FJ128GA010

Il microcontrollore utilizzato nella stazione di test per il controllo del piano motorizzato, della sorgente di luce e del sistema di interruttori, è il PIC24FJ128GA010. Si tratta di un microcontrollore RISC a 16-bit realizzato dalla casa produttrice Microchip e contenuto in un package TQFP da 100-pin. Adotta un architettura di tipo Harvard a basso costo e Low Power, grazie alla tecnologia XLP sviluppata della Microchip (9 nA in sleep mode e 30 μA/MHz in run mode), ed è dotato di una CPU in grado di eseguire fino ad un massimo di 16 MIPS a 32 MHz[13].

Il microcontrollore è dotato inoltre di una memoria di programma non volatile (FLASH) di 128 KB e di una memoria dati (SRAM) di 8 KB.

In figura 4.1 si può osservare lo schema a blocchi dell'architettura del microcontrollore.

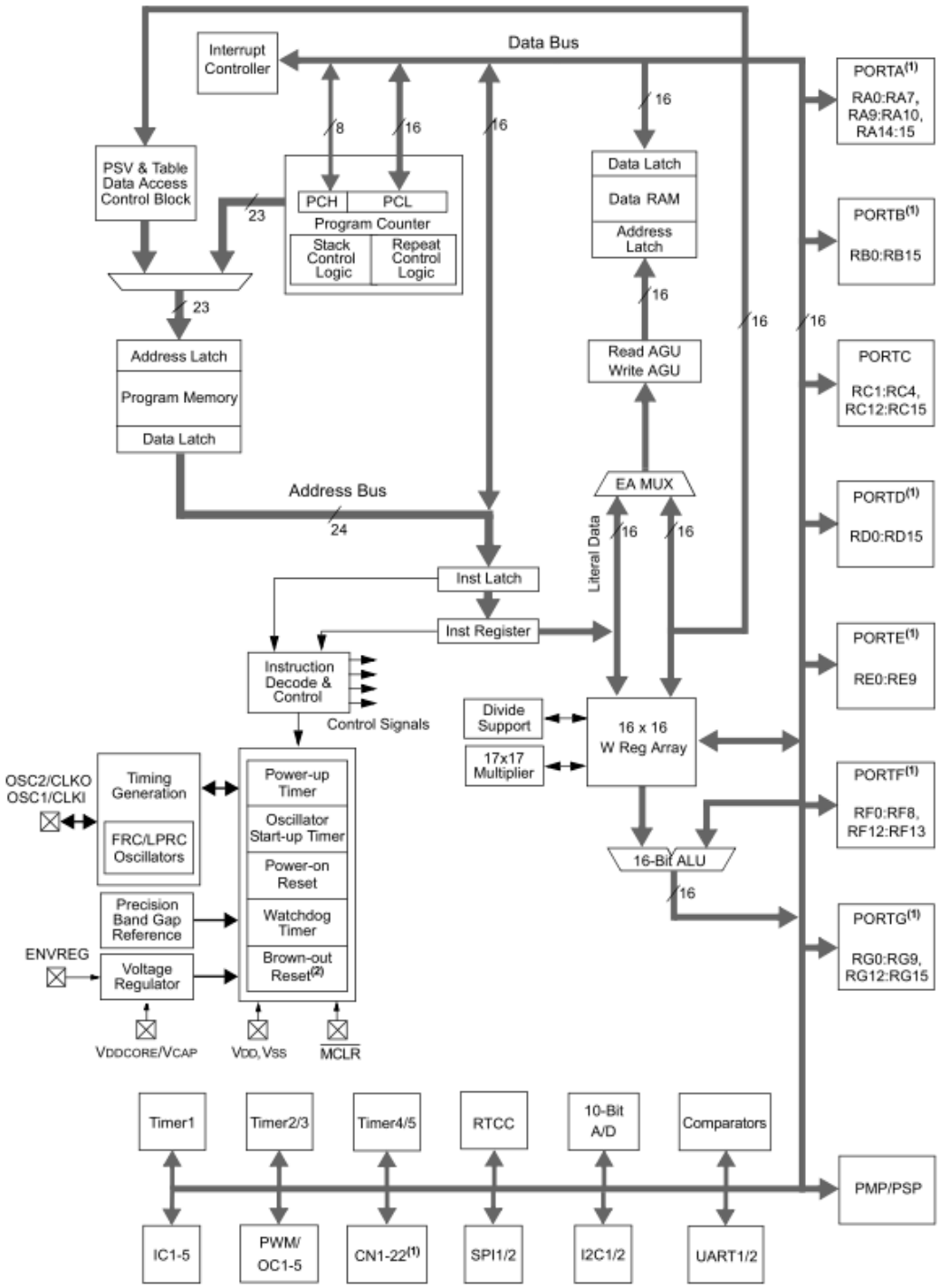

Figura 4.1: Schema a blocchi del architettura logica del microcontrollore PIC24FJ128GA010.

Nel progetto sviluppato in questa Tesi ho utilizzato solo alcune delle periferiche disponibili del PIC. Ne diamo nei paragrafi seguenti una descrizione dettagliata. Per le periferiche da noi non utilizzate, rimandiamo ai manuali di istruzione del dispositivo.

4.1 Porte I/O

Il PIC24FJ128GA010 dispone di 7 porte, per un totale di 53 pin di I/O di uso generale[15]:

- PORTA[RA0:RA7; RA9:RA10; RA14:RA15]
- PORTB[RB0:RB15]
- PORTC[RC1:RC4; RC12:RC15]
- PORTD[RD0:RD15]
- PORTE[RE0:RE9]
- PORTF[RF0:RF8; RF12:RF13]PORTG[RG0:RG9; RG12:RG15]

A ciascuna porta sono associati 4 registri, mediante i quali il programmatore può controllare il funzionamento della porta stessa. I registri sono:

- **Registri TRIS (Data Direction Control register)**: I registri TRISx (TRISA, TRISB, TRISC ecc) determinano se un pin di I/O deve svolgere la funzione di ingresso o di uscita. Se un bit del registro TRISx viene posto ad 1, il pin corrispondente della porta x[1] viene impostato come Ingresso (ovvero: il pin va in alta impedenza), se invece il bit viene posto a 0, il pin relativo funziona come uscita. Normalmente tutti i pin delle porte sono configurabili come ingressi.

- **Registri PORT (I/O Port register):** I registri PORT servono ad accedere al dato presente su un pin di I/O. Quando un pin è configurato come ingresso, la lettura del bit del registro PORT associato al pin, restituisce il livello presente sul pin. Se un pin è configurato come uscita e sul bit relativo nel registro PORT è stato scritto un 1, il pin è portato a livello logico alto, nel caso sia stato scritto uno zero il pin è portato a livello logico basso. Quando un pin viene utilizzato come uscita e quindi si intende impostare il suo livello logico di uscita, il dato scritto viene prima posizionato in un latch, ovvero una cella elementare di memoria, successivamente trasferito al pin, mentre la lettura del registro PORT viene eseguita direttamente sul pin.

- **Registri LAT (Data Latch register):** I registri LAT sono utilizzati per impostare il valore dei pin di I/O ed evitare i problemi dovuti a operazioni di lettura-modifica-scrittura che si avrebbero nel caso fossero utilizzati solo i registri PORT sia per

[1] La x sta per A, B, C, D, E oppure F in tutti i registri;

leggere che per scrivere sui pin di I/O. L'operazione di lettura del registro LATx esegue la lettura del latch di uscita anziché lo stato del pin stesso, mentre l'operazione di scrittura scrive nel latch di uscita e poi successivamente il valore viene trasferito al pin.

- **Registri ODC (Open-Drain Control register):** Questi registri svolgono l'importante funzione di permettere ad un pin configurato come uscita di essere un'uscita Open – Drain. Mettendo ad 1 il bit del registro ODCx, il pin di tale porta è configurato come uscita Open – Drain, se, invece, il bit è a zero (condizione di default) il pin agisce come normale uscita. Un'uscita Open – Drain può fornire unicamente il livello logico basso ma non quello alto. Il vantaggio di avere un'uscita Open – Drain è di poter pilotare dei carichi funzionanti a tensioni più elevate (ma anche più basse) di Vdd: si mette una resistenza di pull – up per fornire il livello logico alto. Ovviamente ci sono dei limiti, il valore massimo di tensione applicabile è definito dal parametro VIH, mentre il minimo vale Vss.

In figura 4.2 è riportato lo schema a blocchi della architettura logica di una porta di I/O.

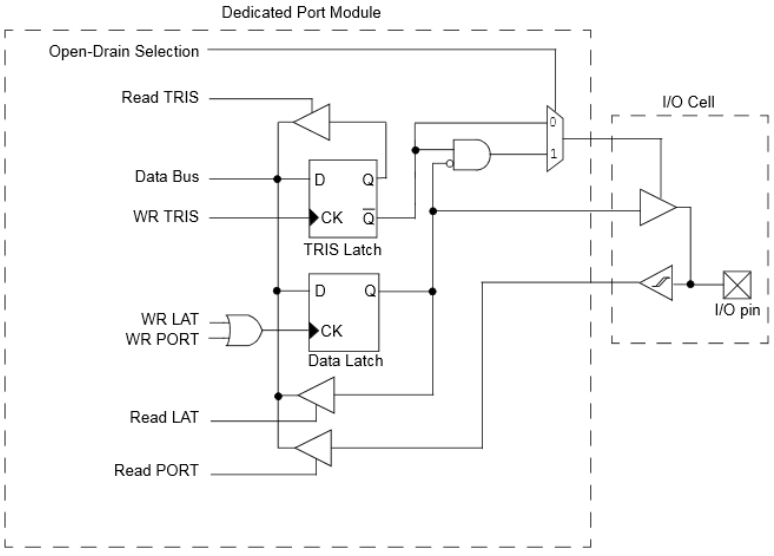

Figura 4.2: Schema a blocchi della architettura logica di una porta di I/0.

Alcuni pin di I/O possono svolgere anche delle funzioni speciali, come ad esempio la generazione di un interrupt in corrispondenza del cambio di stato del pin stesso (Change Notification).

Nel PIC24FJ128GA010 sono disponibili 22 pin che possiedono questa funzione (CN0 – CN21) ed i registri associati a questi pin sono:

- CNENx : Questo registro abilita la funzionalità di interrupt per i pin. In pratica in questo registro ci sono i vari bit CNxIE che permettono di abilitare (bit posto a 1) / disabilitare (bit posto a zero) la generazione dell'interrupt sul cambio di stato per i pin aventi il contrassegno CNx. I bit vengono abilitati singolarmente ma per poter essere rilevati da una funzione di interrupt è necessario settare il flag di abilitazione globale per il change notification (CNIE situato nel registro IEC1).

- CNPUx : Questo registro serve per abilitare le resistenze di pull-up integrate sui pin CNx, una funzione comodissima soprattutto se su tali pin sono stati collegati dei pulsanti. In questo registro sono contenuti i bit CNxPUE, che servono ad abilitare (1) o disabilitare (0) la resistenza di pull-up sul pin CNx.

Per quanto riguarda la tolleranza sulla tensione dei pin configurati come ingressi, ci sono alcuni pin che supportano i 5.5 V mentre tutti gli altri i 3.3 V.

Nel nostro progetto abbiamo utilizzato le porte I/O per interfacciare il microcontrollore con il mondo esterno. Quindi sono state utilizzate per la lettura dei 4 finecorsa presenti nella stazione di test e per controllare i 2 driver che pilotano i 2 motori passo – passo.

4.2 Timer

I moduli Timer sono dei contatori a 16 bit che possono essere utilizzati come temporizzatori, per misure di tempo o per generare clock, oppure come contatori di eventi.

Il PIC24FJ12GA010 dispone di 5 Timer (Timer1, Timer2, Timer3, Timer4, Timer5) che possono essere utilizzati singolarmente oppure in coppia, Timer2/3 e Timer4/5, per realizzare due timer a 32 bit[16].

Ciascun timer viene gestito mediante i seguenti 3 registri:

- **TMRx**[2] **(16-Bit Timer Count register):** questo registro contiene il valore del conteggio.
- **PRx (16-Bit Timer Period Register):** in questo registro viene impostato il valore del periodo del timer
- **TxCON (16-Bit Timer Control Register):** è il registro di controllo del timer

Ogni timer ha anche dei bit associati per il controllo degli interrupt, che sono :

- **TxIF (Interrupt Flag Status Bit):** è un bit di flag che viene settato ad uno quando si ha un match tra il valore di conteggio del timer e il valore contenuto nel registro PRx. È compito del programmatore settare a zero tale bit dopo la rilevazione.
- **TxIP (Interrupt Priority Control Bits):** sono 3 bit utilizzati per impostare il livello di priorità dell'interrupt che viene lanciato quando si è verificato un match e il bit di abilitazione del interrupt è ad 1.
- **TxIE (Interrupt Enable Control Bit):** questo bit è utilizzato per abilitare l'interrupt del timer, deve essere impostato ad 1 se si vuole abilitare l'interrupt del timer.

I 5 timer sono di 3 tipi diversi (Tipo A , Tipo B, Tipo C):

4.2.1 Timer tipo A

Il Timer 1 è di tipo A ed ha le seguenti caratteristiche che lo distinguono da tutti gli altri timer:

- Può essere utilizzato con l'oscillatore a bassa potenza a 32 kHz.
- Può essere utilizzato in modalità asincrona tramite una sorgente di clock esterna.

In particolare, le caratteristiche proprie del Timer di tipo A permettono di utilizzarlo per funzioni di cronometro o come sorgente di clock di sistema secondario.

[2] La x sta per 1, 2, 3, 4 oppure 5 in tutti i registri

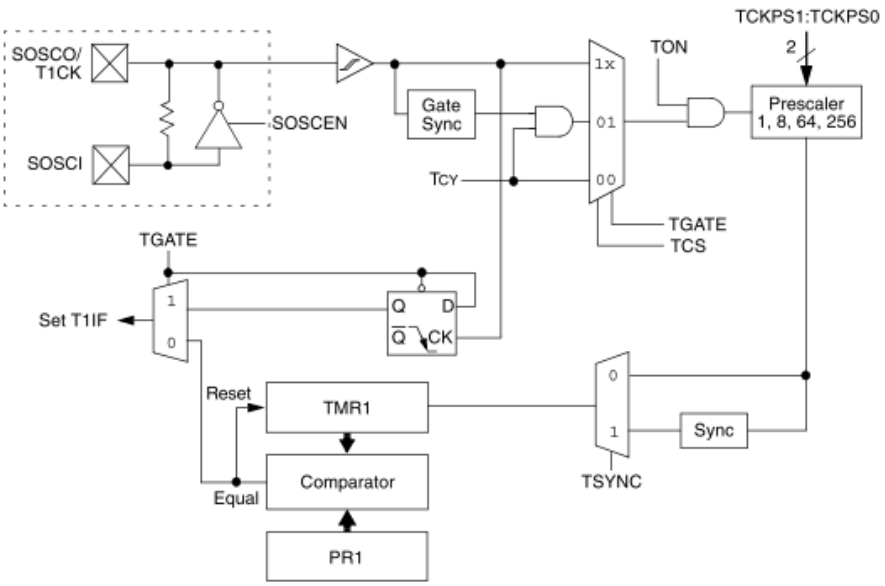

Figura 4.3: Schema a blocchi della architettura logica del Timer tipo A.

4.2.2 Timer tipo B

I Timer 2 e 4 sono di tipo B ed hanno le seguenti caratteristiche che lo distinguono da tutti gli altri timer:

- Un timer di tipo B può essere concatenato con uno di tipo C per realizzare un timer a 32 bit.
- La sincronizzazione del clock per un timer di tipo B viene eseguita dopo il prescaler.

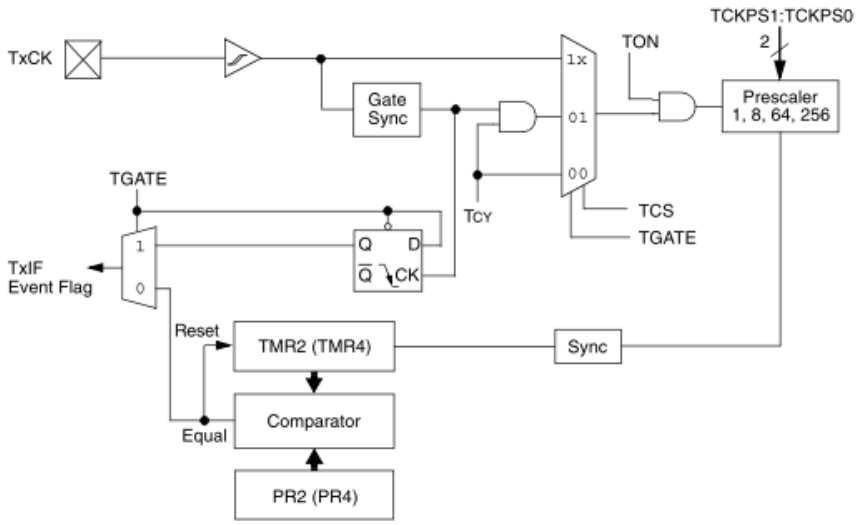

Figura 4.4: Schema a blocchi della architettura logica del Timer tipo B.

4.2.3 Timer tipo C

I timer 3 e 5 sono di tipo C ed hanno le seguenti caratteristiche:

- Può essere concatenato con uno di tipo B per formare un timer a 32 bit.
- Può innescare una conversione A / D.

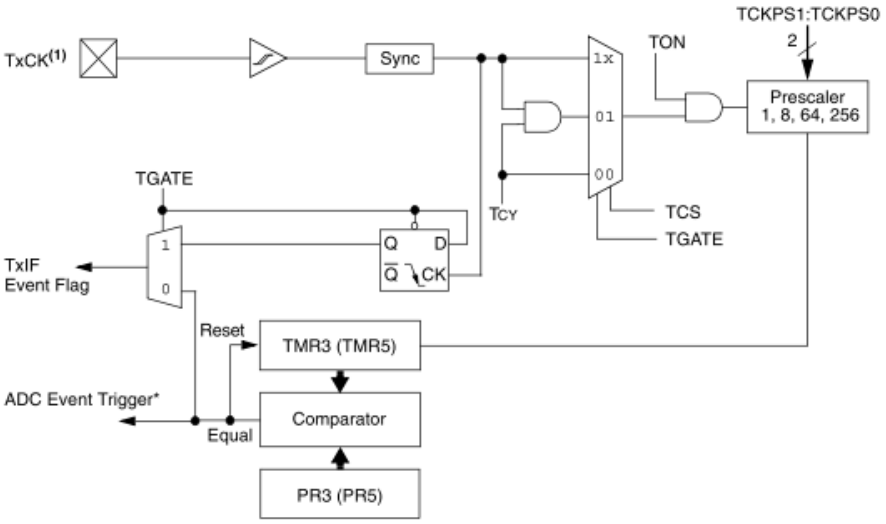

Figura 4.5: Schema a blocchi della architettura logica del Timer tipo C.

Ciascun timer può essere programmato per operare in 4 diversi modi:

- Come timer
- Come contatore sincrono
- Come Gated timer
- Come contatore asincrono

I modi di operare dei timer vengono decisi tramite i seguenti bit del registro di controllo TxCON: TCS, TSYNC, TGATE.

I timer sono utilizzati insieme al modulo PWM per generare il segnale impulsato da inviare al driver per il pilotaggio dei motori passo – passo.

4.3 Compare / PWM Output

Il PIC24FJ128GA010 dispone di 5 moduli Compare/PWM Output identici la cui funzione è quella di comparare il valore del timer selezionato (Timer2 o Timer3) con il valore di uno o due registri di comparazione (OCxR[3], OCxRS) e di generare un singolo impulso in uscita o una sequenza di impulsi quando si verifica un evento di "compare math" e un segnale di interrupt se abilitati[17].

In figura 4.6 è riportato lo schema a blocchi del modulo Compare/PWM Output.

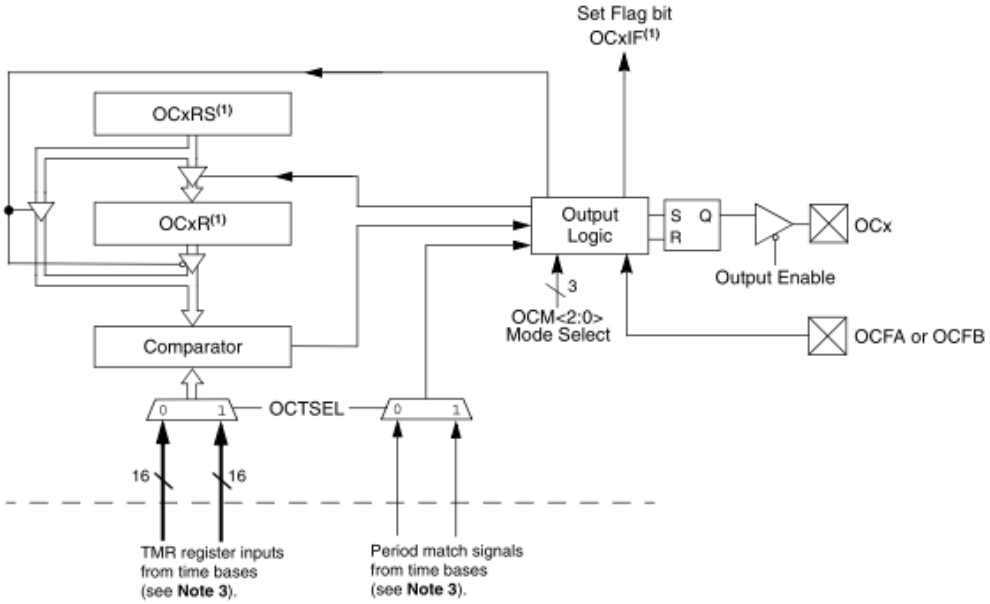

Figura 4.6: Schema a blocchi della architettura logica del modulo Compare/PWM Output.

Ogni modulo può usare come timer di appoggio o il timer2 (TMR2, PR2) o il timer3 (TMR3, PR3) selezionabili tramite il bit OCTSEL.

Ad ogni modulo è associato un set di 3 registri tramite i quali il programmatore può usare questi moduli:

- **OCxCON (Control register):** è il registro di controllo del modulo
- **OCxR (Data Register):** è il registro che contiene il valore da comparare
- **OCxRS (Secondary Data Register):** è il registro di comparazione secondario

Ogni modulo può operare nei seguenti 3 modi:

[3] La x sta per 1, 2, 3, 4 oppure 5 in tutti i registri

1. Single Compare Match Mode

Si ottiene imponendo i seguenti valori nei 3 bit meno significativi del registro di controllo, OCxCON<2:0> = 001 , 010 oppure 011.

In questa modalità viene confrontato il solo valore del registro OCxR con il valore di conteggio del timer, quando avviene un match, uno dei seguenti tre eventi si verifica:

- Se OCxCON = 001, il pin di uscita del modulo (OCx), inizialmente a livello basso (0 logico), viene portato a livello altro (1 logico) quando si verifica l'evento di "compare match". Si genera cosi un fronte alto sul pin OCx.
- Se OCxCON = 010, il pin OCx, inizialmente a livello alto (1), viene portato a livello basso (0) quando si verifica l'evento di "compare match". Si genera cosi un fronte basso sul pin OCx.
- Se OCxCON = 011, il pin OCx cambia valore ad ogni evento di "compare match".

2. Dual Compare Match Mode

Si ottiene imponendo i seguenti valori nei 3 bit meno significativi del registro di controllo, OCxCON<2:0> =100 oppure 101.

In questa modalità il valore del contatore viene confrontato sia con il registro OCxR, per generare un fronte positivo sul pin OCx, che con il registro OCxRS, per generare un fronte negativo sul pin OCx, in questo modo si riescono a generare impulsi.

- Se OCxCON<2:0> = 100 il comparatore è configurato in modalità *Single Output Pulse*.
- Se OCxCON<2:0> = 101 il comparatore è configurato in modalità *Continuos Output Pulse*.

3. Simple Pulse-Width Modulation Mode (PWM)

Questa modalità di funzionamento è particolarmente importante per il fatto che viene utilizzata per pilotare i due motori passo-passo. Infatti permette di realizzare un segnale impulsato con il duty cycle e la frequenza desiderata.

La modalità PWM si ottiene imponendo i seguenti valori nei 3 bit meno significativi del registro di controllo, OCxCON<2:0> = 110 oppure 111.

In questa modalità il registro OCxR diventa solo un registro di lettura, il valore del duty cycle del segnale PWM che si vuole generare viene scritto sul registro OCxRS. Ad ogni inizio del periodo di conteggio il valore del registro OCxRS viene trasferito sul registro OCxR dal hardware del modulo.

Quindi le operazioni necessarie per generare un segnale PWM in uscita sul pin OCx sono:

1. Impostare il periodo del segnale PWM che si vuole generare sul registro PRy[4].
2. Impostare il valore del duty cycle del segnale PWM nel registro OCxRS.
3. Prima di attivare il modulo comparatore, scrivere sul registro OCxR il valore iniziale del duty cycle.
4. Abilitare gli interrupt se necessari.
5. Attivare il modulo comparatore in modalità PWM ed avviare il timer.

Nella figura 4.7 si può osservare la forma d'onda del segnale PWM generato dal modulo.

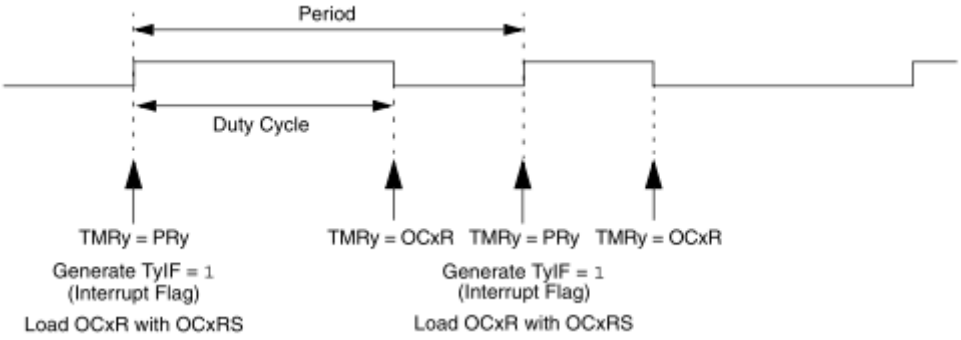

Figura 4.7: Forma d'onda del segnale PWM generato dal modulo.

Il valore da scrivere sul registro PRy, per ottenere un segnale di un determinato periodo si calcola attraverso la seguente formula:

[4] La y sta per 2 o 3

$$PRy = \frac{PWM\ Period}{Tcy\ \cdot TMRy\ Prescale\ Value} - 1$$

Dove Tcy è il periodo del clock del microcontrollore.

In figura 4.8 viene riportato il diagramma temporale relativo alla generazione del segnale PWM da parte del modulo.

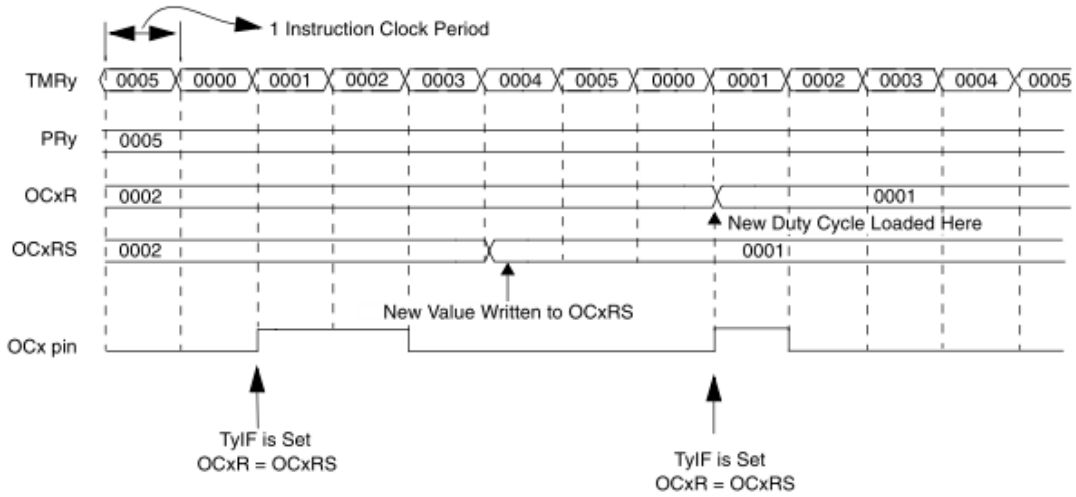

Figura 4.8: Diagramma temporale che descrive il funzionamento del modulo nella modalità PWM.

4.4 Inter – Integrated Circuit (I^2C)

Il modulo I^2C è un interfaccia per comunicazioni seriali bifilari del PIC con altre periferiche o un altro microcontrollore.

Nel nostro progetto viene usato per realizzare una comunicazione seriale tra la scheda Explorer 16 e la scheda di interfaccia per gli encoder di posizione, lo vedremo in dettaglio nel capitolo 5.

Il PIC24FJ128GA010 dispone di 2 moduli I^2C identici, ciascuno dei quali contiene al suo interno sia la logica $\boldsymbol{I^2C}$ **master** che la logica $\boldsymbol{I^2C}$ **slave** indipendenti l'una dall'altra[18].

Questo modulo può operare nei sistemi di comunicazione I^2C come:

- SLAVE DEVICE

- MASTER DEVICE
- MASTER/SLAVE DEVICE in un sistema multi master

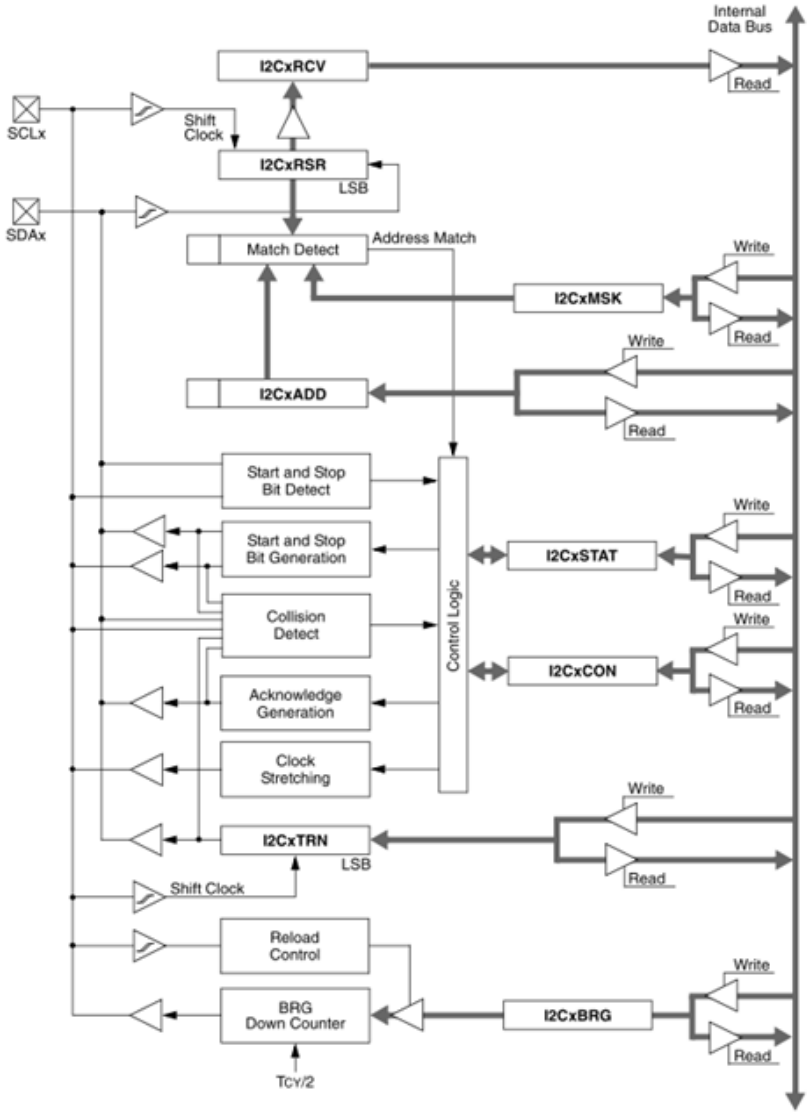

Figura 4.9: Schema a blocchi della architettura logica del modulo I^2C.

4.4.1 Caratteristiche del Bus I^2C

Si tratta di un Bus molto semplice, costituito da due linee bidirezionali, la linea del clock (SCLx) e la linea dati (SDA), mostrato nella figura 4.10.

Durante la comunicazione si ha un dispositivo che svolge la funzione di master il quale inizia il trasferimento sul Bus e genera il segnale del clock, e dall'altra parte si ha un dispositivo che svolge la funzione di slave che risponde al trasferimento.

Poiché le 2 linee SCLx e SDAx sono bidirezionali, gli stadi di uscita dei dispositivi che guidano queste 2 linee devono essere open – drain.

I 2 resistori di pull-up esterni sono usati per assicurare il livello alto quando nessun dispositivo tira giù la linea.

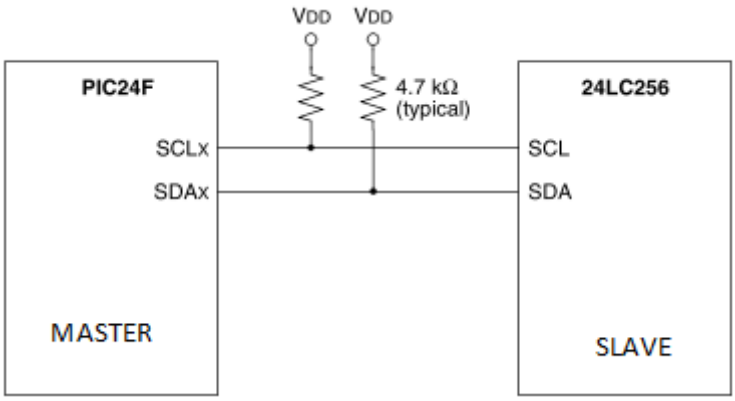

Figura 4.10: Esempio di connessione tra due dispositivi tramite Bus I^2C.

Nel protocollo I^2C ogni dispositivo (Master o Slave) ha un indirizzo. Quando il master vuole iniziare una trasmissione di dati con uno slave per prima cosa invia l'indirizzo del dispositivo con il quale vuole comunicare. Tutti i dispositivi sono in ascolto e controllano il valore dell'indirizzo.

Il master e lo slave lavorano sempre in modo opposto, uno trasmette e l'altro riceve e viceversa. Questo significa che uno solo alla volta ha il controllo della linea.

Il trasferimento dati può essere avviato solo quando il Bus è libero (IDLE : SLCx = 1 ed SDAx = 1).

Durante il trasferimento dei dati la linea SDAx deve rimanere stabile ogni qual volta la linea SCLx è alta. Cambiamenti nella linea dati mentre il SCLx è alto saranno interpretati come una condizione di START o di STOP.

Di conseguenza sono state definite le seguenti condizioni sul Bus che sono mostrate nella figura 4.11.

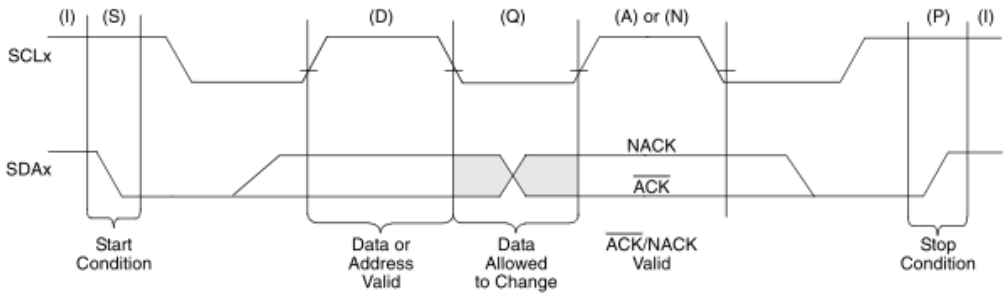

Figura 4.11: Diagramma temporale dove è mostrato il protocollo da rispettare per il Bus I^2C.

Supponiamo di voler realizzare una comunicazione I^2C tra il PIC24F (Master) e una EEPROM (Slave).

È compito del master controllare e sequenziare il protocollo per lo scambio dei messaggi tra i due dispositivi, lo slave guida il Bus solo ad istanti di tempo specifici dettati dal master.

Il protocollo che deve essere rispettato per il corretto scambio di messaggi tra i due dispositivi prevede i seguenti passi mostrati sul diagramma temporale di figura 4.12.

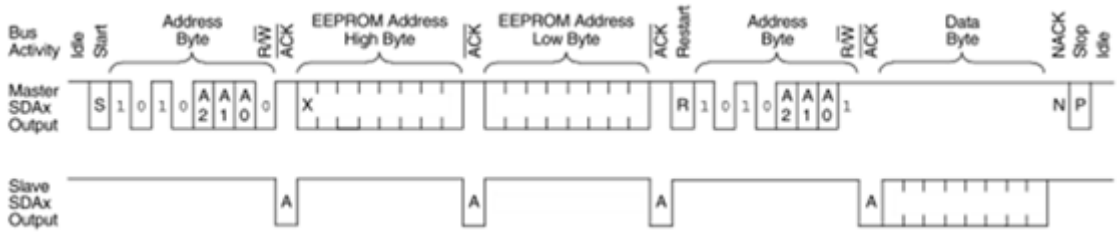

Figura 4.12: Diagramma temporale nel quale è mostrato un tipico messaggio inviato tramite I^2C.

4.4.2 Registri del modulo I^2C

Il modulo I^2C ha 7 registri che permettono al programmatore di utilizzare correttamente questo modulo, che sono:

- **I2CxCON**[5] **(Control Register) :** è il registro di controllo del modulo.
- **I2CxSTAT (Status Register) :** contiene i flag di stato che indicano lo stato del modulo durante le operazioni.

[5] La x sta per 1 o 2 in tutti i registri

- **I2CxMSK (Address Mask Register) :** è utilizzato per abilitare il mascheramento dei bit di indirizzo contenuti nel registro I2CxADD utile in un sistema multi master.
- **I2CxRCV (Receive Buffer Register) :** è un registro di sola lettura dal quale vengono letti i bytes di dati ricevuti.
- **I2CxTRN (Transmit Register) :** è un registro di scrittura/lettura nel quale vengono scritti i bytes da trasmettere.
- **I2CxADD (Address Register):** contiene l'indirizzo dello slave.
- **I2CxBRG (Boudrate Generator Reload Register) :** contiene il valore del Baud Rate (cioè della velocita di trasmissione di un byte) del generatore Baud Rate del modulo I^2C.

Come accennato in precedenza, il modulo I^2C contiene al suo interno sia la logica master che la logica slave. Master e Slave sono indipendenti l'una dall'altra e possono essere attivi contemporaneamente.

Il modulo I^2C si attiva settando ad 1 il bit I2CEN del registro di controllo.

Esso genera 2 sorgenti di interrupt, se abilitati : uno assegnato ali eventi del master (MI2CxIF) e l'altro assegnato agli eventi dello slave (SI2CxIF).

Quando opera come master il modulo I^2C deve generare il clock del sistema. Per generare tale clock il modulo dispone di un Baud Rate Generator, il quale, in base al contenuto del registro I2CxBRG genera un clock di una determinata frequenza, calcolabile secondo la seguente formula:

$$Fscl = \frac{Fcy}{2 \cdot (I2CxBRG + 1)}$$

Dove Fcy è la frequenza del clock del microcontrollore.

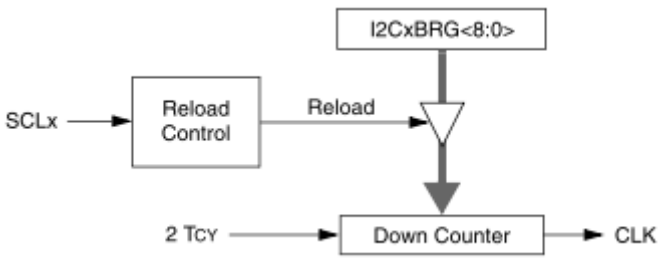

Figura 4.13: Diagramma a blocchi del Baud Rate Generator.

Quando si vuole realizzare una comunicazione I^2C corretta tra due dispositivi è necessario tenere in conto una cosa molto importante: è il software (il programma) il responsabile della corretta costruzione dei messaggi rispettando il protocollo I^2C, cioè garantire il sequenziamento dei componenti del protocollo per costruire un messaggio completo.

Il modulo I^2C controlla solo le singole porzioni del protocollo.

Il software può utilizzare il metodo del polling o degli interrupt durante l'utilizzo del modulo per gestire il corretto sequenziamento dei componenti del protocollo I^2C.

4.5 The Universal Asynchronous Receiver Transmitter (UART)

Il modulo UART (Universal Asynchronous Receiver Transmitter) è un ricevitore – trasmettitore asincrono universale cioè un canale di comunicazione full – duplex asincrono che permette al microcontrollore di comunicare serialmente con un PC o altre periferiche utilizzando protocolli come RS-232, RS-485, LIN 1.2 e IrDA ®. Il modulo supporta anche l'opzione di controllo di flusso hardware attraverso i pin UxCTS e UxRTS, e comprende anche l'encoder e decoder IrDA[19].

Nel nostro progetto la UART è utilizzata per realizzare una comunicazione seriale tra il PC e il microcontrollore in modo tale da poter inviare degli opportuni comandi al microcontrollore tramite terminale da PC.

Il PIC24FJ128GA010 dispone di 2 moduli UART identici.

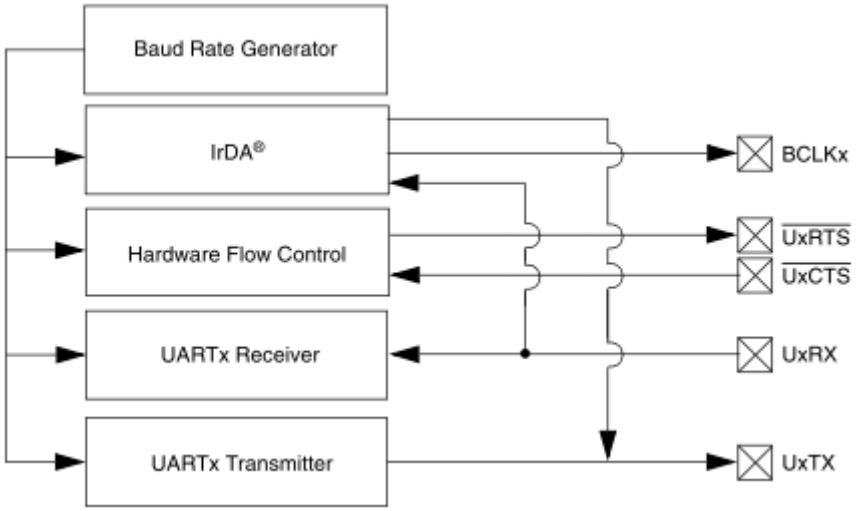

Figura 4.14: Diagramma a blocchi del modulo UART.

L'utilizzo del modulo è molto semplice, una volta inizializzata opportunamente la periferica secondo le proprie specifiche, per leggere i byte ricevuti basta leggere il registro UTXxREG (dove la x sta per 1 nel caso della UART1 o 2 per la UART2), mentre se si vuole trasmettere è sufficiente scrivere un byte alla volta nel registro UxRXREG.

Per maggiori informazioni sulle varie periferiche del PIC24FJ128GA010 è necessario fare riferimento al data sheet.

Capitolo 5

Il progetto della scheda di interfaccia per gli encoder di posizione

Dal momento che non tutti i pin del microcontrollore PIC24FJ128GA010 supportano la tensione di 5 V, utilizzata per i segnali di uscita degli encoder, per non impegnare eccessivamente le risorse del dispositivo, ho progettato una semplice scheda di interfaccia per effettuare la lettura dei 24 bit di uscita dei due encoder. La scheda di interfaccia in questione utilizza il protocollo di comunicazione seriale I^2C per trasmettere i dati al microcontrollore PIC24FJ128GA010 che, nel seguito, chiameremo "master".

I 2 encoder utilizzati nella stazione di test sono dei modelli vecchi con interfaccia di uscita parallela a 12 bit composta da transistor NPN open collector alimentati a 5 V. Questo significa che la lettura dei pin di uscita dell'encoder necessita di una resistenza di pull-up per ogni pin.

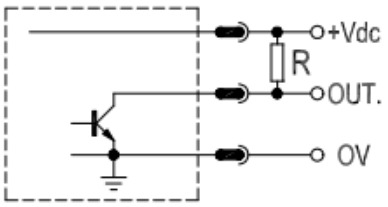

Figura 5.1: Stadio di uscita del encoder LA390-G4096-5-CM5-R.

Il progetto, la realizzazione fisica ed, infine, il collaudo di questa scheda per la lettura dei 24 bit degli encoder, è stata una parte significativa del mio lavoro di Tesi.

Nel seguito di questo capitolo viene descritto in dettaglio il progetto della scheda di interfaccia che ho realizzato.

5.1 Schema elettrico del PCB

La funzione della scheda di interfaccia è quella di leggere i 24 pin di uscita a 5 V dei due encoder e trasmetterli tramite una comunicazione seriale al master.

Abbiamo deciso di utilizzare un secondo microcontrollore PIC e precisamente il modello dsPIC30F4013, il quale opera alla tensione di alimentazione di 5 V, ed è simile al master, ma con una quantità di risorse molto inferiori. Per questo motivo ha anche un costo di circa 4 euro[14]. Utilizzare un microcontrollore nella scheda di interfaccia semplifica l'elaborazione dei dati relativi agli encoder da parte del master. Lo si vedrà più nel dettaglio quando descriverò il firmware installato nel dsPIC30F4013. Nel seguito faremo riferimento a questo microcontrollore con il termine "slave".

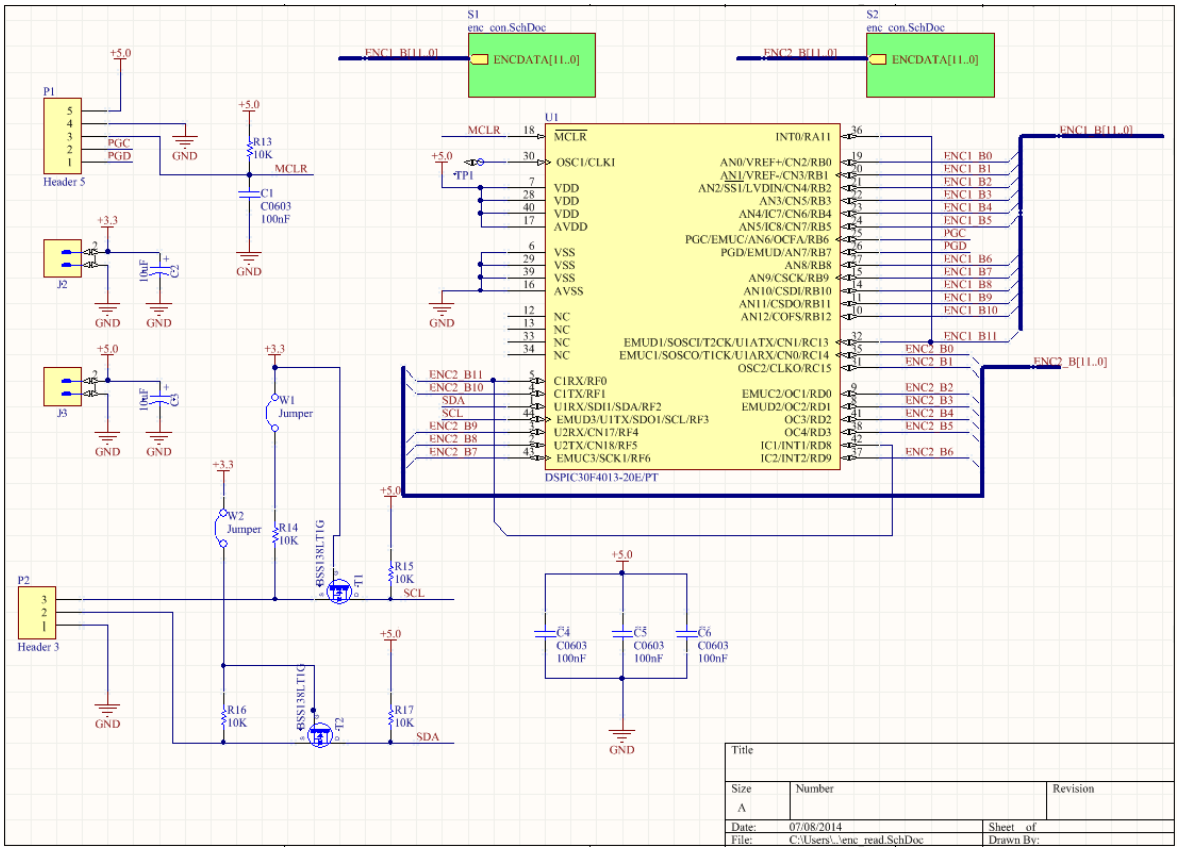

Figura 5.2: Schema elettrico della scheda di interfaccia per gli encoder di posizione.

Il connettore P1 è utilizzato per programmare il microcontrollore.

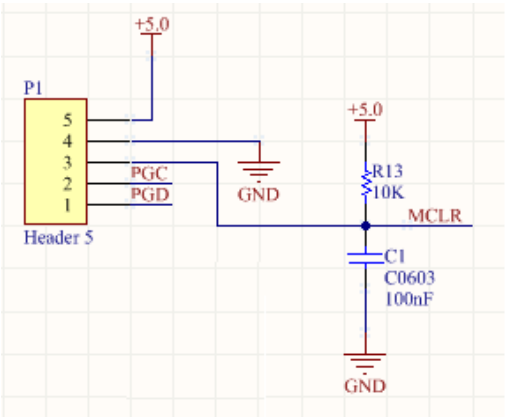

Figura 5.3: Schema elettrico del connettore P1 utilizzato per programmare il dsPIC30F4013.

I connettori J2 e J3 sono utilizzati per le tensioni di alimentazione di 3.3 V e 5 V. La tensione 5 V è l'alimentazione per il dsPIC30F4013 e per gli encoder, mentre la tensione 3 V è necessaria per il pull-up del Bus I^2C lato master.

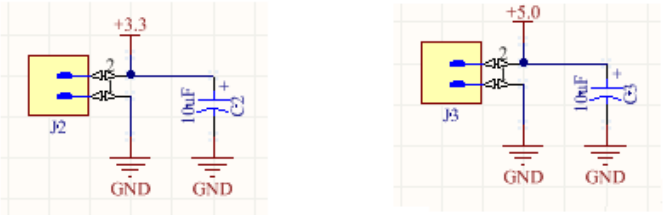

Figura 5.4: A sinistra è mostrato lo schema elettrico del connettore J2 per la tensione di alimentazione di 3.3 V; a destra lo schema elettrico del connettore J3 per la tensione di alimentazione di 5 V.

I blocchi ENCDATA[11..0] contengono al loro interno il seguente circuito.

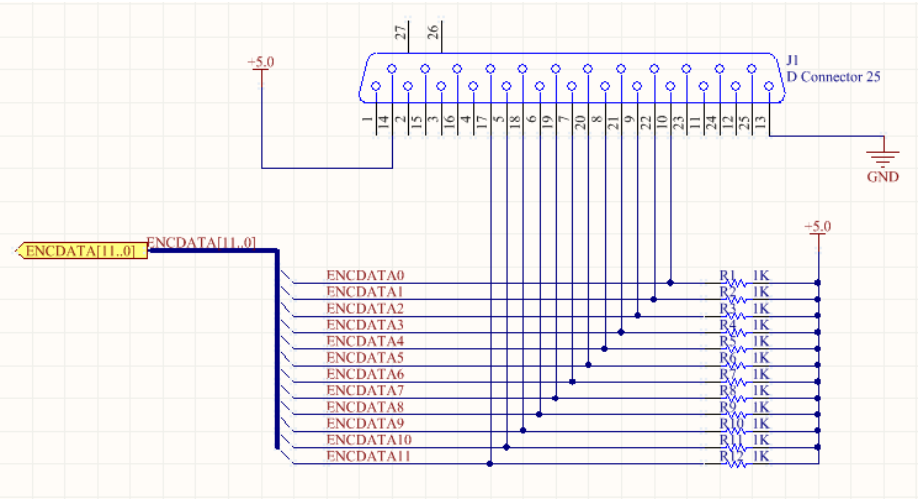

Figura 5.5: Schema elettrico del connettore DSUB 25 utilizzato per alimentare e leggere i 12 bit del encoder.

Il connettore P2 è il connettore al Bus I^2C tramite il quale il master comunica con lo slave.

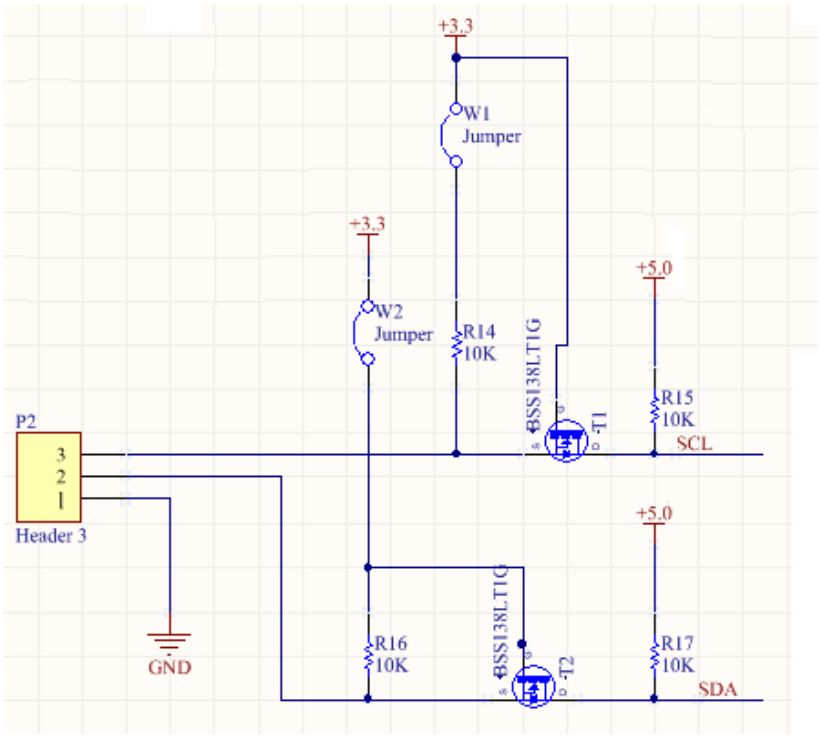

Figura 5.6: Schema elettrico del connettore P2 utilizzato per comunicare in I^2C con il master.

Dal momento che il master utilizza la tensione di alimentazione di 3.3 V e lo slave la tensione di 5 V, per realizzare il Bus I^2C , in modo che i due microcontrollori possano comunicare correttamente tra di loro, si è pensato di utilizzare il circuito mostrato in figura 5.6, dove T1 e T2 sono due N – MOSFET di potenza utilizzati tipicamente per conversioni DC – DC[20].

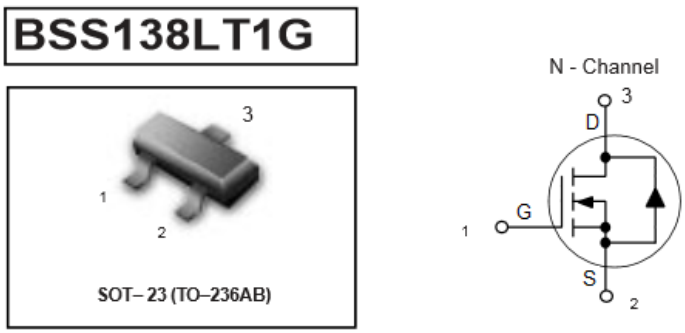

Figura 5.7: A sinistra è riportata l'immagine del N-MOSFET BSS138LT1G, a destra il simbolo circuitale.

Il funzionamento di questo circuito è molto semplice. Quando il Bus I^2C è in condizioni di inattività (idle mode) la linea dati (SDA) e il clock (SCL) si trovano alla tensione 3.3 V nel lato master e alla tensione 5 V nel lato slave grazie alle resistenze di pull-up e al fatto che i due MOSFET non sono in conduzione, visto che le V_{GS} sono alla tensione 0 V. Quando invece il master tira giù a 0 V una delle due linee (oppure entrambe), ad esempio SDA, il MOSFET T2 conduce, perché la V_{GS} vale ora 3.3 V e quindi impone a 0 V anche la linea SDA dello slave.

Quando, invece, è lo slave (il quale può tirare giù sola la linea SDA quando deve inviare dati al master) a tirare giù la linea dati SDA, il MOSFET T2 non conduce perché $V_{GS} = 0\ V$, a condurre invece è il diodo di ricircolo presente tra Source e Drain , tirando giù la linea SDA del master.

5.2 Layout e realizzazione del PCB

Dopo aver scelto i componenti ed aver creato lo schematico dell'intero progetto, abbiamo realizzato il progetto del circuito stampato. La scelta del numero di piani conduttori, della geometria della scheda e, infine, il piazzamento e il collegamento di tutti i componenti, sono passaggi necessari per completare il progetto.

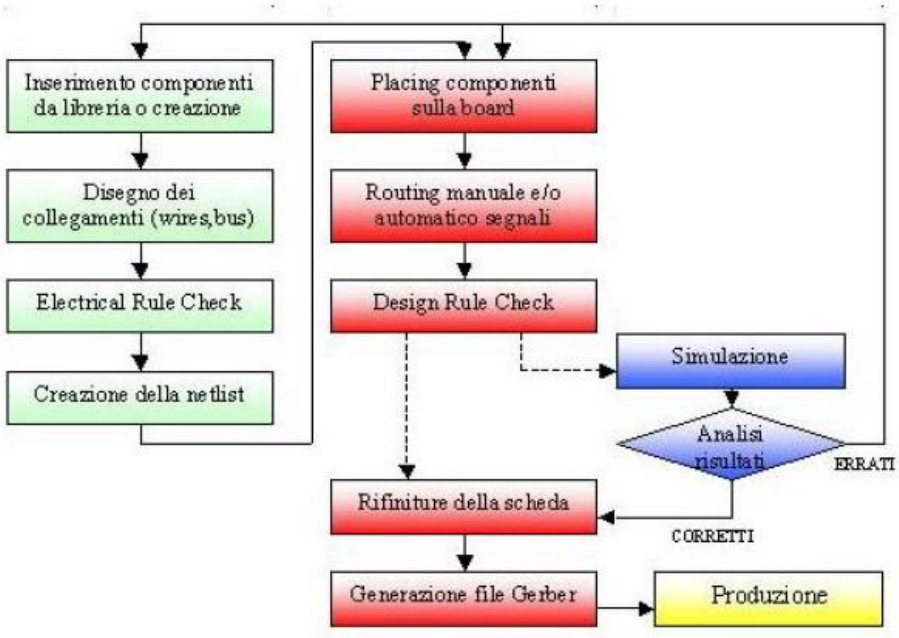

Figura 5.8: Rappresentazione schematica del flusso di progetto del PCB.

Per il disegno del circuito stampato ho utilizzato il CAD Altium Designer 10, un ambiente di sviluppo ampio e potente, ricco di librerie di componenti e di footprint, che permette di realizzare una progettazione completa ed affidabile.

Dal momento che lo schema elettrico della scheda è abbastanza semplice, con tre sole tensioni di alimentazione (3 V , 5 V, GND) , ho deciso di realizzare il layout del PCB su due piani, Top e Bottom.

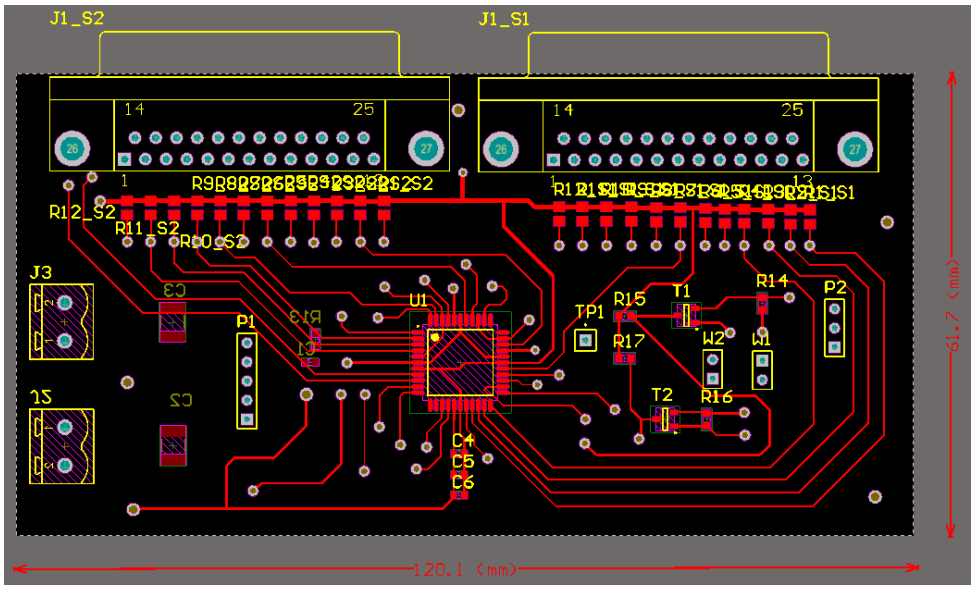

Figura 5.9: Vista lato Top del layout del PCB.

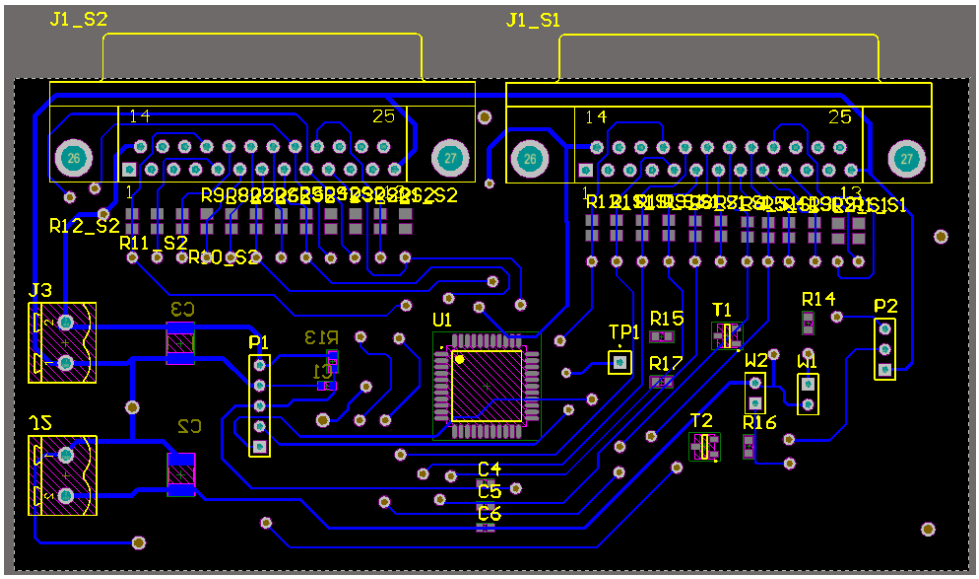

Figura 5.10: Vista lato Bottom del Layout del PCB.

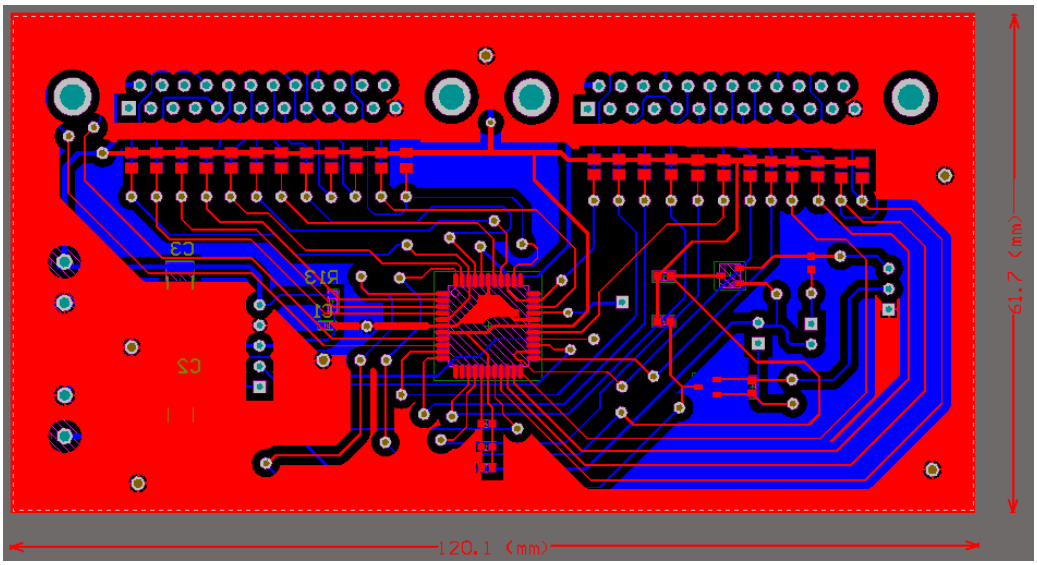

Figura 5.11: PCB completo della scheda. L'immagine mostra il lato Top sovrapposto al lato Bottom.

Per la realizzazione fisica del PCB è stata adottata la tecnica "domestica" basata sulla stampa del master su un foglio di carta lucida, seguito dalla fotoincisione del layout sulla basetta, tramite bromografo e sviluppo in soda caustica, ed infine lavaggio attraverso acido cloridrico e perossido di idrogeno per eliminare la parte esposta alla luce. Il processo termina con la foratura, montaggio e saldatura manuale dei componenti, ottenendo cosi la scheda che si può osservare nelle figure 5.12 e 5.13.

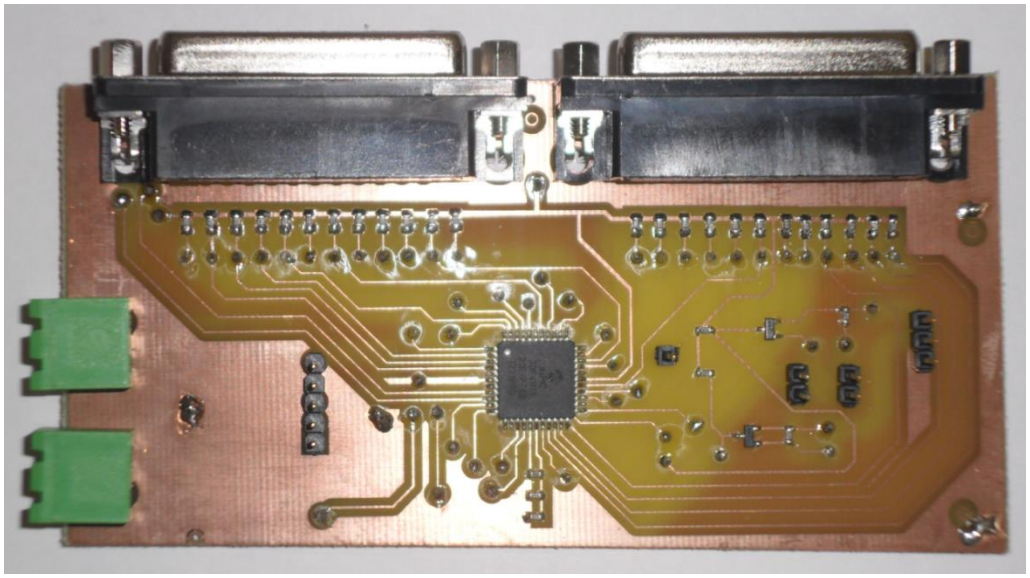

Figura 5.12: Immagine lato Top della scheda di interfaccia per la lettura degli encoder.

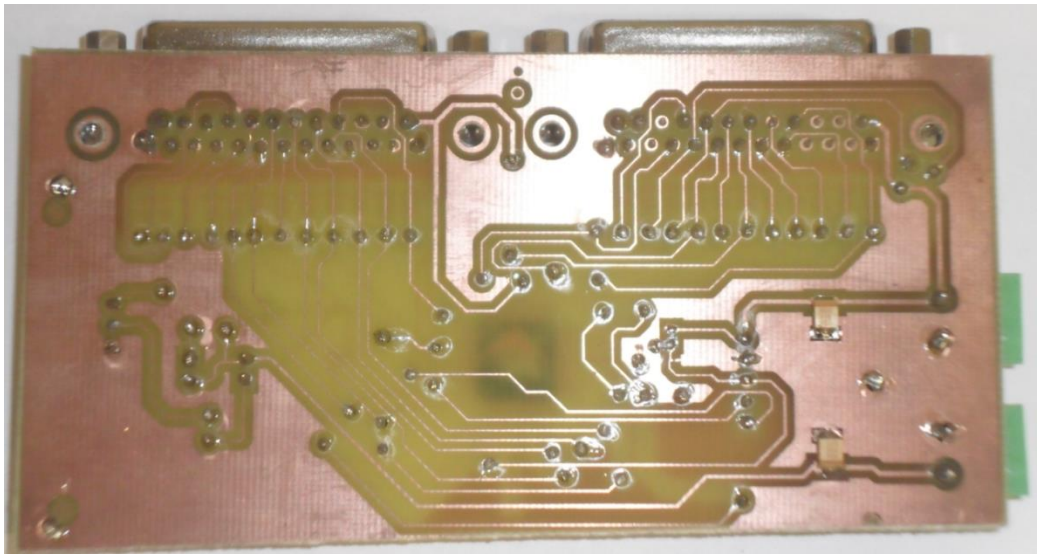

Figura 5.13: Immagine lato Bottom della scheda di interfaccia per la lettura degli encoder.

Un volta realizzata la scheda e verificata l'integrità di tutte le connessioni elettriche, ho programmato il dsPIC30F4013 con dei semplici programmi di test per verificare il corretto funzionamento della scheda. Controllato il corretto funzionamento della scheda, sono passato allo sviluppo del firmware da installare sul dsPIC30F4013 della scheda.

5.3 Sviluppo del firmware installato nel dsPIC30F4013

Si è detto che la scheda Explorer 16, che monta il PIC24FJ12GA010, comunica con la scheda di interfaccia mediante il protocollo seriale I^2C, dove il PIC24FJ12GA010 funge da master e il dsPIC30F4013 da slave.

Il compito della scheda di interfaccia è quello di rimanere sempre in attesa di un comando dal master e, ricevuta la richiesta, trasmettergli la lettura dell'encoder e del numero di giri.

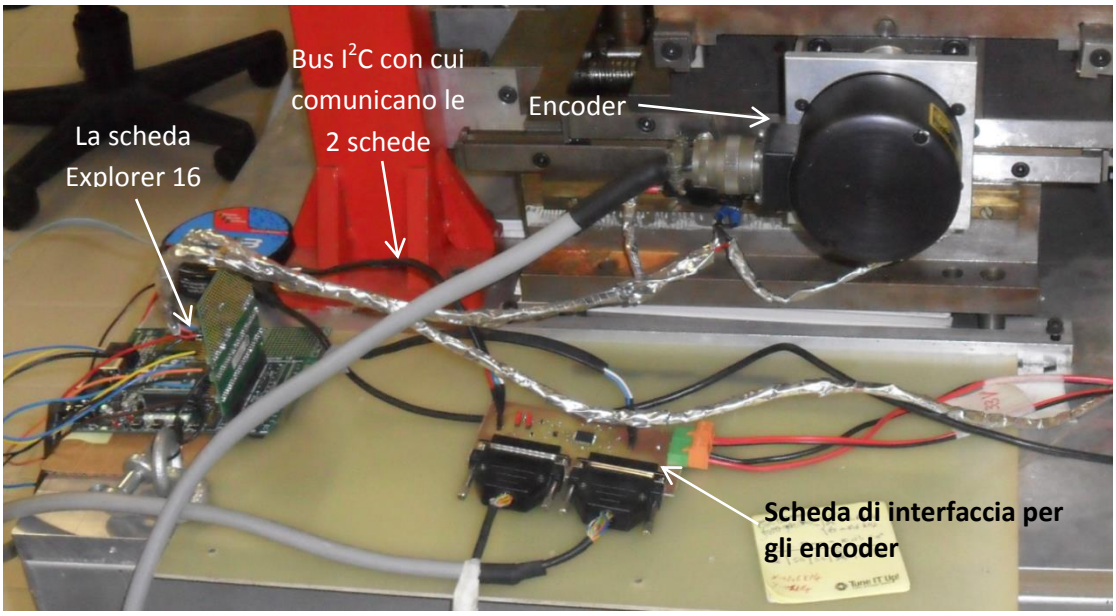

Figura 5.14: Set-up per lo sviluppo del firmware installato nel dsPIC30F4013. Si può notare la scheda di interfaccia alla quale sono collegati i 2 connettori delle uscite dei 2 encoder, la scheda Explorer 16 che gestisce l'intero sistema e infine il Bus I^2C che permette la comunicazione seriale delle due schede.

I comandi possibili che il master può inviare allo slave sono 4:

- **init_ENC1, init_ENC2**: prima di ogni spostamento del piano il master invia questi due comandi allo slave per richiedere la trasmissione del valore corrente dei due encoder, che rappresentano quindi le posizioni iniziali degli encoder prima di uno spostamento del piano motorizzato. Una volta ricevuto questo comando e trasmesso il valore dei due encoder al master, lo slave conta il numero di volte che tale valore iniziale del encoder ricompare quando il piano si muove, per sapere quanti giri completi dell'albero dell'encoder vengono compiuti finché il piano non si ferma.

- **read_ENC1, read_ENC2**: al termine di ogni spostamento del piano il master invia questi due comandi allo slave per chiedere la trasmissione del numero di giri compiuti dai due encoder nell'ultimo spostamento e il valore attuale dei due encoder. Con queste informazioni il master è in grado di calcolare in modo preciso lo spostamento del piano motorizzato, dopo di che viene azzerato il contatore del numero di giri degli encoder.

Essendo l'uscita degli encoder in codifica Gray, ovviamente il valore letto dagli encoder viene prima convertito in codice binario dalla funzione *gray_to_bin(int valore)* e poi inviato al master.

Dalla conoscenza di queste informazioni ricevute dallo slave, il master può determinare in modo molto preciso ogni spostamento del piano della stazione di test.

Di sotto è riportato il firmware installato nel dsPIC30F4013:

```
/*
 * File:   main_I2C_Slave.c
 * Author: Ergys
 *
 * Created on 6 luglio 2014, 15.52
 */

#include <stdio.h>
#include <stdlib.h>
#include <p30F4013.h>
#include <i2c.h>

/***************************Setup configuration bits***************/
#pragma config FOSFPR = FRC             // Oscillator (Internal Fast
RC (No change to Primary Osc Mode bits))
#pragma config FCKSMEN = CSW_FSCM_OFF   // Clock Switching and Monitor
(Sw Disabled, Mon Disabled)
_FWDT(WDT_OFF);                 /* Turn off the Watch-Dog Timer.  */
_FBORPOR(MCLR_EN & PWRT_OFF);   /* Enable MCLR reset pin and turn off
the power-up timers. */
_FGS(CODE_PROT_OFF);            /* Disable Code Protection */

/*****************************************************************/

/*****************  DEFINITIONS ***********************************/
#define ENC1
((PORTCbits.RC13<<11)|(PORTBbits.RB12<<10)|(PORTBbits.RB11<<9)|(PORTBb
its.RB10<<8)|(PORTBbits.RB9<<7)|(PORTBbits.RB8<<6)|(PORTBbits.RB5<<5)|
(PORTBbits.RB4<<4)|(PORTBbits.RB3<<3)|(PORTBbits.RB2<<2)|(PORTBbits.RB
1<<1)|(PORTBbits.RB0))
#define ENC2
((PORTFbits.RF0<<11)|(PORTFbits.RF1<<10)|(PORTFbits.RF4<<9)|(PORTFbits
.RF5<<8)|(PORTFbits.RF6<<7)|(PORTDbits.RD9<<6)|(PORTDbits.RD3<<5)|(POR
TDbits.RD2<<4)|(PORTDbits.RD1<<3)|(PORTDbits.RD0<<2)|(PORTCbits.RC15<<
1)|(PORTCbits.RC14))
#define read_ENC1 0x01
#define read_ENC2 0x02
#define init_ENC1 0x03
#define init_ENC2 0x04
/*****************************************************************/

/***************** Globally Defined Variables ********************/
volatile unsigned char stato_I2C;
volatile unsigned int n_giri_enc1;
volatile unsigned int n_giri_enc2;
volatile unsigned int enc_gray;
volatile unsigned int pos_enc;
volatile unsigned int pos_init_enc1;
volatile unsigned int pos_init_enc2;
volatile unsigned char flag_byte_tx;
volatile unsigned char a;
```

```
volatile unsigned char b;
/*****************************************************************/

/*******Initialize I/O pins*****************************************/
void IO_init(void){
    ADPCFG = 0xFFFF; /* Configure ANx pins used by ICD for digital
i/o*/
    TRISB = 0xFFFF; //all input
    TRISC = 0xFFFF;
    TRISD = 0xFFFF;
    TRISF = 0xFFFF;
}
/*****************************************************************/

void I2CInit()
{
    I2CCON = 0x9040; //Enable I2C1 module, enable clock stretching
    I2CADD = 0x50;  // 7-bit I2C slave address must be initialized
here.
    IFS0bits.SI2CIF = 0; //clear flag
    IEC0bits.SI2CIE = 1; //enable Slave I2C interrupt
}

void INTx_IO_Init(void)
{
        INTCON2bits.INT0EP = 1; /*Setup INT0, INT1 pins to interrupt
on falling edge*/
        INTCON2bits.INT1EP = 1;

        IFS0bits.INT0IF = 0;    /*Reset INT0 interrupt flag */
        IEC0bits.INT0IE = 1;    /*Enable INT0 Interrupt Service
Routine */

        IFS1bits.INT1IF = 0;    /*Reset INT1 interrupt flag */
        IEC1bits.INT1IE = 1;    /*Enable INT1 Interrupt Service
Routine */

}


/*****Conversione da Gray a Binario*********************************/
unsigned int gray_to_bin (unsigned int num){
    unsigned int mask;
    for(mask = (num >> 1); mask != 0; mask = (mask >> 1)){
        num = num ^ mask;
    }
    return (num);
}
/*****************************************************************/

/************ This is the ISR for I2C1 Slave interrupt **************/
void __attribute__((interrupt,no_auto_psv)) _SI2CInterrupt(void)
{
      IFS0bits.SI2CIF = 0; //clear flag
      unsigned char Temp;     //used for dummy read

      if((I2CSTATbits.R_W == 0)&&(I2CSTATbits.D_A == 0)){ //Address
matched
            Temp = I2CRCV;          //dummy read
            return;
```

```
      }

      if((I2CSTATbits.R_W == 0)&&(I2CSTATbits.D_A == 1)){ //ricezione
byte dal master
            stato_I2C = I2CRCV;

            if((stato_I2C == read_ENC1) || (stato_I2C == read_ENC2))
flag_byte_tx = 3;
            else    flag_byte_tx = 2;

            I2CSTATbits.I2COV = 0;  /* clear OV flag */
            I2CCONbits.SCLREL = 1; //Release SCL line
            return;
        }

      if((I2CSTATbits.R_W == 1)&&(I2CSTATbits.D_A == 0)) //invio primo
byte al master
      {
            Temp = I2CRCV;
                switch(stato_I2C){

                    case read_ENC1:
                    {
                        if(flag_byte_tx == 3){
                            I2CTRN = n_giri_enc1;//il primo byte che
invio al master è il numero di giri compiuti dall'encoder1
                            flag_byte_tx --;
                        }
                        else if(flag_byte_tx == 2){
                                enc_gray =(unsigned int) ENC1;
                                pos_enc = gray_to_bin(enc_gray);
                                I2CTRN = pos_enc; // 2 byte trasmesso
al master
                                flag_byte_tx --;
                             }
                             else if(flag_byte_tx == 1){ // 3 byte
trasmesso al master
                                    I2CTRN = (pos_enc >> 8);
                                    flag_byte_tx --;
                                    n_giri_enc1 = 0;
                                    a = 0;
                                }
                        break;
                    }

                    case read_ENC2:
                    {
                        if(flag_byte_tx == 3){
                            I2CTRN = n_giri_enc2;//il primo byte che
invio al master è il numero di giri compiuti dall'encoder2
                            flag_byte_tx --;
                        }
                        else if(flag_byte_tx == 2){
                                enc_gray =(unsigned int) ENC2;
                                pos_enc = gray_to_bin(enc_gray);
                                I2CTRN = pos_enc; // 2 byte trasmesso
al master
                                flag_byte_tx --;
                             }
```

```
                            else if(flag_byte_tx == 1){ // 3 byte
trasmesso al master
                                    I2CTRN = (pos_enc >> 8);
                                    flag_byte_tx --;
                                    n_giri_enc2 = 0;
                                    b = 0;
                                }
                        break;
                    }

                    case init_ENC1: //in questo caso il master a
chiesto allo slave di inviarli solo la sua posizione angolare, che
verrà considerata come posizione iniziale
                    {
                        if(flag_byte_tx == 2){
                            enc_gray =(unsigned int) ENC1;
                            pos_init_enc1 = gray_to_bin(enc_gray);
                            I2CTRN = pos_init_enc1; //invio prima il
byte meno significativo
                            flag_byte_tx --;
                        }
                        else if(flag_byte_tx == 1){
                                I2CTRN = (pos_init_enc1 >> 8);
                                flag_byte_tx --;
                                n_giri_enc1 = 0;
                                a = 0;
                            }
                        break;
                    }

                    case init_ENC2:
                    {
                        if(flag_byte_tx == 2){
                            enc_gray =(unsigned int) ENC2;
                            pos_init_enc2 = gray_to_bin(enc_gray);
                            I2CTRN = pos_init_enc2; //invio prima il
byte meno significativo
                            flag_byte_tx --;
                        }
                        else if(flag_byte_tx == 1){
                                I2CTRN = (pos_init_enc2 >> 8);
                                flag_byte_tx --;
                                n_giri_enc2 = 0;
                                b = 0;
                            }
                        break;
                    }

                }

            I2CCONbits.SCLREL = 1;  //Release SCL1 line
                return;
        }
}

//************************ Interrupt service routine for INT0 ********
void __attribute__((interrupt,no_auto_psv)) _INT0Interrupt(void)
{
    IFS0bits.INT0IF = 0;    /*Reset INT0 interrupt flag */
    a = 1;
```

```
}

//*********************** Interrupt service routine for INT1 ********
void __attribute__((interrupt,no_auto_psv)) _INT1Interrupt(void)
{
    IFS1bits.INT1IF = 0;    /*Reset INT1 interrupt flag */
    b = 1;
}

int main() {
    stato_I2C = 0;
    n_giri_enc1 = n_giri_enc2 = 0;
    pos_init_enc1 = pos_init_enc2 = 0;
    flag_byte_tx = 0;
    a = b = 0;
    IO_init();
    I2CInit();
    INTx_IO_Init();

    while(1){
        if((pos_init_enc1 == gray_to_bin((unsigned int) ENC1)) && (a
==1)){
            a = 0;
            n_giri_enc1 ++;
        }

        if((pos_init_enc2 == gray_to_bin((unsigned int) ENC2)) && (b
==1)){
            b = 0;
            n_giri_enc2 ++;
        }
    }
    return (0);
}
```

Capitolo 6

Sviluppo del firmware per il controllo del piano motorizzato della stazione di test

Il grosso del lavoro svolto nella mia Tesi è stato lo sviluppo del firmware installato sul microcontrollore PIC24FJ12GA010 per il movimento e il controllo di precisione del piano motorizzato, dove sarà posizionata la matrice di fotosensori da testare.

L'obiettivo è realizzare un sistema automatizzato che permetta di spostare il piano motorizzato alle coordinate desiderate, inviate tramite terminale da PC.

I fotosensori che devono essere testati hanno area attiva di $100\,mm^2$, per la loro caratterizzazione è richiesto che vengano sollecitati con un'opportuna sorgente di luce in almeno un paio di punti della superficie dell'area attiva. Di conseguenza sono richiesti spostamenti minimi del piano dell'ordine di 0.5 mm.

Per il collaudo del progetto ho utilizzato la **Explorer 16 Development Board**, che monta il PIC24FJ12GA010, un LCD, 4 tasti e 7 LED.

Figura 6.1: Immagine della scheda Explorer 16 con a fianco il programmatore ICD 3.

Per il movimento del piano della stazione di test vengono utilizzati due motori passo – passo pilotati mediante il PIC attraverso il driver G203V. Il PIC24FJ12GA010 dispone di 5 moduli PWM realizzati per offrire un modo molto semplice per il controllo dei motori passo – passo. Infatti, settando opportunamente i parametri del modulo PWM come la frequenza, il duty – cycle e il numero di impulsi, viene generato un treno di impulsi che inviato al driver fa girare l'albero motore di un determinato numero di passi e alla velocità voluta.

6.1 Determinazione dei rapporti di trasmissione dei motori

La prima fase del lavoro è stata misurare con precisione, tramite l'ausilio di un calibro e un comparatore, lo spostamento lungo gli assi X e Y del piano corrispondenti ad un giro completo dell'albero motore. Questo è stato necessario in quanto sarebbe stato complicato calcolare a priori i rapporti di trasmissione relativi ai due motori.

Figura 6.2: Nell'immagine di sopra si può osservare il meccanismo di trasmissione dall'albero motore alla vite senza fine relativo all'asse X, nell'immagine di sotto quello relativo all'asse Y.

Per fare queste misure ho dovuto sviluppare un programma che fa compiere un giro completo dell'albero motore ogni volta che si preme un tasto sulla scheda **Explorer 16**.

```
/*
 * File:   main.c
 * Author: Ergys
 *
 * Created on 18 marzo 2014, 9.37
 */

#include <stdio.h>
#include <stdlib.h>
#include <p24fj128ga010.h>
#include "lcdPmp.h"
#include "ports.h"

// CONFIG2
#pragma config POSCMOD = HS     // Primary Oscillator Select (HS
Oscillator mode selected)
#pragma config OSCIOFNC = ON    // Primary Oscillator Output Function
(OSC2/CLKO/RC15 functions as port I/O (RC15))
#pragma config FCKSM = CSDCMD  // Clock Switching and Monitor (Clock
switching and Fail-Safe Clock Monitor are disabled)
#pragma config FNOSC = PRI      // Oscillator Select (Primary
Oscillator (XT, HS, EC))
#pragma config IESO = OFF       // Internal External Switch Over Mode
(IESO mode (Two-Speed Start-up) disabled)
// CONFIG1
#pragma config WDTPS = PS32768 // Watchdog Timer Postscaler (1:32,768)
#pragma config FWPSA = PR128    // WDT Prescaler (Prescaler ratio of
1:128)
#pragma config WINDIS = ON      // Watchdog Timer Window (Standard
Watchdog Timer enabled,(Windowed-mode is disabled))
#pragma config FWDTEN = OFF     // Watchdog Timer Enable (Watchdog
Timer is disabled)
#pragma config ICS = PGx2       // Comm Channel Select
(Emulator/debugger uses EMUC2/EMUD2)
#pragma config GWRP = OFF       // General Code Segment Write Protect
(Writes to program memory are allowed)
#pragma config GCP = OFF        // General Code Segment Code Protect
(Code protection is disabled)
#pragma config JTAGEN = OFF     // JTAG Port Enable (JTAG port is
disabled)

#define S6 PORTDbits.RD7
#define S5 PORTAbits.RA7
#define S4 PORTDbits.RD13
#define S3 PORTDbits.RD6
#define dir_motor_X LATAbits.LATA0 //RA0 <-- dir motore asse X
#define dir_motor_Y LATAbits.LATA1 //RA1 <-- dir motore asse y
#define TCY 0.00000025 //Clock period TCY=2/FOSC (FOSC = 8MHz)
#define PWM_freq_x 1.5*1e03 //frequenza del segnale PWM del motore x
#define PWM_freq_y 0.9*1e03
#define PRx (1.0/(PWM_freq_x * TCY)-1.0) //Valore del Piriod Register
per ottenere il segnale PWM della frequenza desiderata
#define PRy (1.0/(PWM_freq_y * TCY)-1.0)
```

```
void initIO(void){
 //Pin configurati come ingressi
    TRISDbits.TRISD7 = 1;
    TRISAbits.TRISA7 = 1;
    TRISDbits.TRISD13 = 1;
    TRISDbits.TRISD6 = 1;

 //Pin configurati come uscite
    TRISAbits.TRISA0 = 0; //dir_motor_X
    TRISAbits.TRISA1 = 0; //dir_motor_Y
}

//Initialize PWM for Motor X
void init_PWM_X(void){
    T2CON = 0; //Turn off timer 2
    OC1CON = 0; //Turn off Output Compare 1 Module
    OC1RS = PRx/2; //Duty cycle = 50%
    OC1R = 0;
    PR2 = PRx;
    IPC1bits.T2IP = 4; //interrupt priority 4
    IFS0bits.T2IF = 0; //Clear Output Compare 1 interrupt flag
    IEC0bits.T2IE = 0; //Turn off TMR2 Interrupt Enable
    OC1CONbits.OCM = 6; //PWM mode on OC1 pin; Fault pin Disable
}

//Initialize PWM for Motor Y
void init_PWM_Y(void){
    T3CON = 0; //Turn off timer 3
    OC2CON = 0; //Turn off Output Compare 2 Module
    OC2RS = PRy/2; //Duty cycle = 50%
    OC2R = 0;
    PR3 = PRy;
    IPC2bits.T3IP = 4; //interrupt priority 4
    IFS0bits.T3IF = 0; //Clear Output Compare 2 interrupt flag
    IEC0bits.T3IE = 0; //Turn off TMR3 Interrupt Enable
    OC2CONbits.OCTSEL = 1; //Timer3 is the clock source for output
compare 2
    OC2CONbits.OCM = 6; //PWM mode on OC2 pin; Fault pin Disable
}

int main(void) {
    LCDInit();
    initIO();
    init_PWM_X();
    init_PWM_Y();
    char s1[15];
    char s2[15];
    int n_giri_x = 0;
    int n_giri_y = 0;
    int count;
    while(1){
        if( !S3 ){
            count = 0;
            dir_motor_X = 1;
            OC1CON = 0x0006;// select PWM mode, fault pin is disable
            T2CONbits.TON = 1;
            while(1){
                if(IFS0bits.T2IF){
```

```
                    IFS0bits.T2IF = 0; //Clear Output Compare 1
interrupt flag
                    count++;
                    if ( count == 2000 ){
                        OC1CON = 0;
                        T2CONbits.TON = 0;
                        n_giri_x++;
                        break;
                    }
                }
            }
        }
        sprintf(s1 ,"n_giri_x : %d " ,n_giri_x);
        LCDwriteLine(LCD_LINE1, (unsigned char *)s1);

        if( !S6 ){
            break;
        }
    }

    while(1){
        if( !S5 ){
            count = 0;
            dir_motor_Y = 1;
            OC2CON = 0x000E;
            T3CONbits.TON = 1;
            while(1){
                if(IFS0bits.T3IF){
                    IFS0bits.T3IF = 0; //Clear Output Compare 1
interrupt flag
                    count++;
                    if ( count == 2000 ){
                        OC2CON = 0;
                        T3CONbits.TON = 0;
                        n_giri_y++;
                        break;
                    }
                }
            }
        }
        sprintf(s2 ,"n_giri_y : %d " ,n_giri_y);
        LCDwriteLine(LCD_LINE2, (unsigned char *)s2);

        if( !S4 ){
            break;
        }
    }

    while(1);
    return 0;
}
```

Misurando gli spostamenti del piano corrispondenti ad un determinato numero di giri dell'albero motore, ed effettuando numerose misure, ho potuto stimare che:

- **Motore asse X** : un giro dell'albero motore corrisponde ad uno spostamento del piano di circa 1 mm.
- **Motore asse Y** : un giro dell'albero motore corrisponde ad uno spostamento del piano di circa 3.2 mm.

Non conoscere con esattezza i rapporti di trasmissione dei due motori non pone un grosso limite. Nonostante siano richiesti spostamenti minimi molto piccoli dell'ordine di 0,5 mm, grazie alla presenza dei due encoder assoluti, è possibile misurare lo spostamento effettivo del piano con precisione di $1{,}2\ \mu m$. Tale valore si ottiene considerando il fatto che l'encoder è a 12 bit e quindi ha una risoluzione di 4096; sapendo che l'albero dell'encoder è collegato alla vite senza fine, la quale presenta una filettatura standard con un passo di 5 mm, che fa muovere il piano della stazione di test, si calcola che lo spostamento minimo misurato dall'encoder è :

$$Risoluzione\ = \frac{5000\ \mu m}{4096} \cong 1.2\ \mu m$$

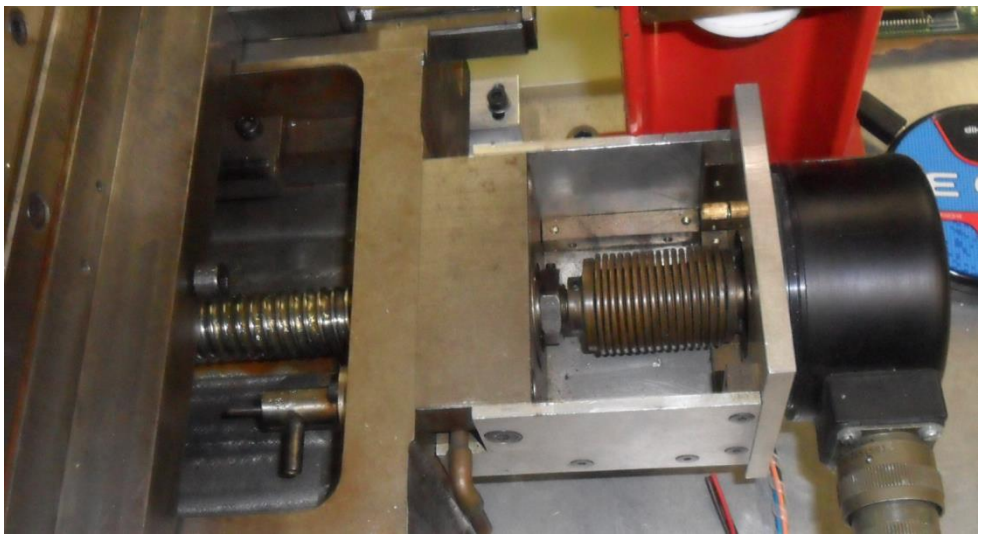

Figura 6.3: Immagine del aggancio dell'albero dell'encoder alla vite senza fine per effettuare la misura di precisione della posizione del piano della stazione di test.

Quindi con l'ausilio degli encoder è possibile realizzare un sistema ad anello chiuso che consente di spostare il piano esattamente alla coordinata voluta.

6.2 La funzione init_motor()

Determinati i rapporti di trasmissione dei due motori, ho sviluppato la funzione *init_motor()*. Tale funzione serve per inizializzare il sistema e riportare il piano all'origine.

La funzione verrà chiamata solo all'inizio, quando si avvia il sistema. Le prime istruzioni che esegue sono l'inizializzazione delle varie periferiche del sistema e, successivamente, si abilitano i due motori e il piano si sposta verso l'origine. L'origine viene determinata dalla posizione dei finecorsa. Dal momento che, per ciascun asse ci sono due finecorsa, uno rappresenta l'origine e l'altro la fine.

Essendo i finecorsa attualmente utilizzati nella stazione di test dei dispositivi meccanici, soffrono del problema dei rimbalzi elettrici. Per ovviare a questo problema, una volta che il motore è giunto al finecorsa, viene fatto avanzare in avanti di un paio di millimetri in modo tale che la leva a rotella del finecorsa si sollevi e la sua uscita ritorni a livello logico 0. Questo punto rappresenta l'origine.

Ovviamente la soluzione ideale sarebbe l'uso di finecorsa ottici, in quanto non soffrono del problema dei rimbalzi elettrici. Purtroppo sul macchinario erano montatati questi meccanici.

Figura 6.4: Immagine in cui vengono mostrati due finecorsa della stazione di test e il posizionamento del piano nel punto di origine.

Una volta che il piano è giunto nell'origine, i motori vengono disattivati e vengono abilitati gli interrupt sui segnali dei finecorsa. Questo perché, una volta che il sistema è stato inizializzato, il piano deve muoversi solo all'interno della zona delimitata dai finecorsa, se il piano tenta di andare oltre tale zona vorrebbe dire che qualcosa non sta funzionando e l'intero sistema si deve immediatamente arrestare.

Chiameremo **FC_X_1** e **FC_Y_1** i finecorsa che rappresentano l'origine nei due assi X ed Y.

Il codice della funzione *init_motor()* è riportato nel seguito.

```
/********* Funzione che sposta il piano nell'origine ***************/
void init_motor(void){
    flag_term = 1;
    init_PWM_X(3.5*1e03);
    init_PWM_Y(1.5*1e03);
    unsigned long n_x = 14000;
    unsigned long n_y = 6000;

//Spostamento all'origine del piano lungo l'asse X
    if(FC_X_1 == 0){ //se il motore non si trova in posizione iniziale
riportalo nella posizione iniziale
        disable_motor_X = off; //Abilito il driver del motore asse X
        dir_motor_X = DOWN;
        T2CONbits.TON = 1;
        while( FC_X_1 == 0){
            ; //attendo finche il piano non giunge all'origine
        }

//per evitare i problema dei rimbalzi dei finecorsa faccio avanzare di
un paio di mm in avanti il carrello lungo l'asse x
        dir_motor_X = UP;
        count_n_pulse(n_x , X);
        wait(100000);
        disable_motor_X = on; //Disabilito il driver del motore asse X
    }
    else {
        disable_motor_X = off;
        dir_motor_X = UP;
        count_n_pulse(n_x , X);
        wait(100000);
        disable_motor_X = on;
    }

//Spostamento all'origine del piano lungo l'asse Y
    if(FC_Y_1 == 0){ //se il motore non si trova in posizione iniziale
riportalo nella posizione iniziale
        disable_motor_Y = off; //Abilito il driver del motore asse Y
        dir_motor_Y = DOWN;
        T3CONbits.TON = 1;
        while( FC_Y_1 == 0){
            ; //attendo finche il piano non giunge all'origine
```

```
        }

        dir_motor_Y = UP;
        count_n_pulse(n_y , Y);
        wait(100000);
        disable_motor_Y = on;
    }
    else {
       disable_motor_Y = off;
       count_n_pulse(n_y , Y);
       wait(100000);
       disable_motor_Y = on;
    }
//x_pos e y_pos sono 2 variabili che tengono conto della posizione
attuale del piano
    x_pos = 0;
    y_pos = 0; //il piano si trova nell'origine

    I2CSend(init_ENC1);
    I2CSend(init_ENC2);

    ExtInt();
    flag_term = 0;
}
/*****************************************************************/
```

6.3 La funzione go_to(x , y)

La funzione *go_to(x , y)* consente di spostare il piano della stazione di test alla coordinata desiderata rispetto al punto definito come origine.

Tale funzione riceve come parametri le coordinate X ed Y espresse in micron, dopo di che confronta se le coordinate inserite coincidono con la posizione attuale del piano. Se le coordinate coincidono non succede nulla, vengono semplicemente stampate sul display della Explorer 16 le coordinate del piano; se invece le coordinate non coincidono, vengono abilitati i motori e il piano viene spostato alla nuova coordinata. Per fare questo si calcola la differenza tra la posizione nella quale si vuole spostare il piano e la posizione attuale e si invia al driver del motore un numero opportuno di impulsi (calcolati conoscendo il rapporto di trasmissione precedentemente ricavato), per spostare il piano della differenza tra le due coordinate.

Quando il motore si ferma, e quindi si suppone che il piano sia giunto alla coordinata desiderata, si controlla, misurando lo spostamento effettivo del piano con l'encoder, se il piano si trova effettivamente alla coordinata voluta. Se ciò non è vero si sposta il piano della differenza necessaria per giungere alla coordinata desiderata. Questo ultimo processo viene ripetuto finché la differenza tra la posizione effettiva e quella

desiderata del piano non sia inferiore a 10 μm. Si tratta di un errore accettabile rispetto a spostamenti minimi di 500 μm richiesti.

Il codice della funzione *go_to(X, Y)* è riportato nel seguito.

```
/**** Funzione che sposta il piano in una determinata coordinata ****/
void go_to( unsigned long x, unsigned long y ){
    flag_term = 1;
    unsigned long d, s, n_step, ris1, ris2, ris3;
    char direction;

//movimentazione motore asse X se il motore non si trova già in quella
coordinata e se x è più piccolo di x_max
    if ( (x != x_pos) && (x < x_max) ){
        disable_motor_X = off; //Abilito il driver del motore asse X

        if ( x_pos > x ){
            dir_motor_X = DOWN;
            direction = CCW;
            d = (x_pos - x);
        }
        else {
            dir_motor_X = UP;
            direction = CW;
            d = (x - x_pos);
        }
        n_step =(unsigned long) (2 * d);
        if(d <= 20000){
            init_PWM_X(0.8*1e03);
        }
        else   init_PWM_X(3.0*1e03);

        count_n_pulse(n_step , X);
        wait(100000);

        int i = 10;
        while(i){

            if(flag_int){
                flag_int = 0;
                string = "ERRORE: sei ";
                LCDwriteLine(LCD_LINE1, (unsigned char *)string);
                string = "andato oltre ";
                LCDwriteLine(LCD_LINE2, (unsigned char *)string);
                CloseINT1(); CloseINT2(); CloseINT3(); CloseINT4();
                return;
            }

            I2CSend(read_ENC1);
            i --;
            if(direction == CW){
               if(pos_enc >= pos_init_enc1){
                  ris1 = (unsigned long) (n_turns_enc * 4.096*1e03);
                  ris2 = (unsigned long) (pos_enc - pos_init_enc1);
                  ris3 = (unsigned long) (ris1 + ris2);
                  s = (unsigned long) (ris3 * 1.220703125);
```

```
                }
                else {
                    ris1 = (unsigned long) ((n_turns_enc + 1) *
4.096*1e03);
                    ris2 = (unsigned long) (pos_init_enc1 - pos_enc);
                    ris3 = (unsigned long) (ris1 - ris2);
                    s =(unsigned long) (ris3 * 1.220703125);
                }
                x_pos = (x_pos + s);

                if(s != d){
                     if( s > d){
                         if((x_pos >= x) && (x_pos <= (x + 10)))
break;
                         dir_motor_X = DOWN;
                         direction = CCW;
                         d = (x_pos - x);
                     }
                     else {
                         if((x_pos >= (x - 10)) && (x_pos <= x))
break;
                         dir_motor_X = UP;
                         direction = CW;
                         d = (x - x_pos);
                     }
                 }
                 else    break;
            }
            else {
              if(pos_enc > pos_init_enc1){
                  ris1 = (unsigned long) ((n_turns_enc + 1) *
4.096*1e03);
                  ris2 = (unsigned long) (pos_enc - pos_init_enc1);
                  ris3 = (unsigned long) (ris1 - ris2);
                  s = (unsigned long) (ris3 * 1.220703125);
              }
              else {
                  ris1 = (unsigned long) (n_turns_enc * 4.096*1e03);
                  ris2 = (unsigned long) (pos_init_enc1 - pos_enc);
                  ris3 = (unsigned long) (ris1 + ris2);
                  s = (unsigned long) (ris3 * 1.220703125);
              }
              x_pos = (x_pos - s);

              if(s != d){
                     if( s > d){
                         if((x_pos >= (x - 10)) && (x_pos <= x))
break;
                         dir_motor_X = UP;
                         direction = CW;
                         d = (x - x_pos);
                     }
                     else {
                         if((x_pos >= x) && (x_pos <= (x + 10)))
break;
                         dir_motor_X = DOWN;
                         direction = CCW;
                         d = (x_pos - x);
                     }
               }
```

```
                else    break;
            }

            I2CSend(init_ENC1);

            n_step =(unsigned long) (2 * d);
            init_PWM_X(0.8*1e03);
            count_n_pulse(n_step , X);
            wait(100000);
        }
        wait(100000);
        disable_motor_X = on; //Disabilito il driver del motore asse X
    }
    sprintf(str1 ,"Pos X: %lu um" ,(unsigned long) x_pos);
    LCDwriteLine(LCD_LINE1, (unsigned char *)str1);

//movimentazione motore asse Y se il motore non si trova già in quella
coordinata e se y è più piccolo di y_max
    if ( (y != y_pos) && ( y < y_max) ){
        disable_motor_Y = off; //Abilito il driver del motore asse Y

        if ( y_pos > y ){
            dir_motor_Y = DOWN;
            direction = CW;
            d = (y_pos - y);
        }
        else {
            dir_motor_Y = UP;
            direction = CCW;
            d = (y - y_pos);
        }
        n_step =(unsigned long) (0.625 * d);
        if(d <= 20000){
            init_PWM_Y(0.4*1e03);
        }
        else  init_PWM_Y(1.2*1e03);

        count_n_pulse(n_step , Y);
        wait(100000);

        int i = 10;
        while(i){

            if(flag_int){
                flag_int = 0;
                string = "ERRORE: sei ";
                LCDwriteLine(LCD_LINE1, (unsigned char *)string);
                string = "andato oltre ";
                LCDwriteLine(LCD_LINE2, (unsigned char *)string);
                CloseINT1(); CloseINT2(); CloseINT3(); CloseINT4();
                return;
            }

            I2CSend(read_ENC2);
            i --;
            if(direction == CW){
               if(pos_enc >= pos_init_enc2){
                   ris1 = (unsigned long) (n_turns_enc * 4.096*1e03);
                   ris2 = (unsigned long) (pos_enc - pos_init_enc2);
                   ris3 = (unsigned long) (ris1 + ris2);
```

```
                    s = (unsigned long) (ris3 * 1.220703125);
                }
                else {
                    ris1 = (unsigned long) ((n_turns_enc + 1) *
4.096*1e03);
                    ris2 = (unsigned long) (pos_init_enc2 - pos_enc);
                    ris3 = (unsigned long) (ris1 - ris2);
                    s = (unsigned long) (ris3 * 1.220703125);
                }
                y_pos = y_pos - s;

                if(s != d){
                     if( s > d){
                         if((y_pos >= (y - 10)) && (y_pos <= y))
break;
                         dir_motor_Y = UP;
                         direction = CCW;
                         d = (y - y_pos);
                     }
                     else {
                         if((y_pos >= y) && (y_pos <= (y + 10)))
break;
                         dir_motor_Y = DOWN;
                         direction = CW;
                         d = (y_pos - y);
                     }
                }
                else    break;
            }
            else {
                if(pos_enc > pos_init_enc2){
                    ris1 = (unsigned long) ((n_turns_enc + 1) *
4.096*1e03);
                    ris2 = (unsigned long) (pos_enc - pos_init_enc2);
                    ris3 = (unsigned long) (ris1 - ris2);
                    s = (unsigned long) (ris3 * 1.220703125);
                }
                else  {
                     ris1 = (unsigned long) (n_turns_enc * 4.096*1e03);
                     ris2 = (unsigned long) (pos_init_enc2 - pos_enc);
                     ris3 = (unsigned long) (ris1 + ris2);
                     s = (unsigned long) (ris3 * 1.220703125);
                }
                y_pos = (y_pos + s);

                if(s != d){
                     if( s > d){
                         if((y_pos >= y) && (y_pos <= (y + 10)))
break;
                         dir_motor_Y = DOWN;
                         direction = CW;
                         d = (y_pos - y);
                     }
                     else {
                         if((y_pos >= (y - 10)) && (y_pos <= y))
break;
                         dir_motor_Y = UP;
                         direction = CCW;
                         d = (y - y_pos);
                     }
```

```
                }
                else    break;
            }

            I2CSend(init_ENC2);

            n_step =(unsigned long) (0.625 * d);
            init_PWM_Y(0.4*1e03);
            count_n_pulse(n_step , Y);
            wait(100000);
        }

        wait(100000);
        disable_motor_Y = on;
    }
    sprintf(str2 ,"Pos Y: %lu um" ,(unsigned long) y_pos);
    LCDwriteLine(LCD_LINE2, (unsigned char *)str2);
    I2CSend(init_ENC1);
    I2CSend(init_ENC2);
    flag_term = 0;
}
/*****************************************************************/
```

6.4 Sviluppo di un protocollo di comunicazione seriale tra il microcontrollore e il PC

Una volta scritte le due funzioni *init_motor()* e *go_to(X , Y)*, l'ultimo passo rimasto nel progetto del firmware, per il movimento e il controllo del piano della stazione di test, è stato realizzare una semplice comunicazione seriale tra il PC e la scheda Explorer 16 in modo tale che sia possibile comandare il sistema da terminale su PC.

Per realizzare la comunicazione seriale ho utilizzato uno delle due UART (Universal Asynchronous Receiver Transmitter) presenti nel PIC24FJ12GA010 e un cavo RS232 per collegare il PC alla Explorer 16.

Il protocollo di comunicazione seriale sviluppato è molto semplice, il primo carattere da inviare è il comando che può essere solo uno tra "i", "x", "y", "g":

- Il carattere ' **i** ' è il comando utilizzato per inizializzare il sistema e quindi per chiamare la funzione *init_motor()*.
- Il carattere ' **x** ' è il comando utilizzato per inserire la coordinata dell'asse X dove si vuole spostare il piano.
- Il carattere ' **y** ' è il comando utilizzato per inserire la coordinata dell'asse Y dove si vuole spostare il piano.

- Il carattere ‘ **g** ‘ è il comando utilizzato per muovere il piano all’ultima coordinata inserita e quindi per chiamare la funzione *go_to(x , y)*.

Tutti gli altri caratteri che vengono inviati vengono scartati.

Una volta inserito uno dei caratteri di comando, successivamente vengono accettati solo altri 6 caratteri, cioè il valore numerico della coordinata espresso in micron. Ad esempio se si vuole spostare il piano alla coordinata x = 100 μm si scrive sul terminale **x000100**.

Infine è previsto il carattere di termine comando ‘ **#** ‘. Quando si invia il carattere di termine comando ‘ # ’, se il sistema è libero e non sta eseguendo la funzione *go_to(x , y)* oppure *init_motor(),* risponde con una ACK stampando sul terminale la lettera ‘ **A** ’ ed il comando viene eseguito; se invece il sistema è impegnato allora risponde con un NACK stampando sul terminale la lettera ‘ **N** ‘ ed il comando non viene eseguito.

Naturalmente nel caso dei comandi **i** e **g** basta scrivere **i#** e **g#**.

Nel caso in cui si sbaglia a inserire la coordinata, oppure si vuole cambiare la coordinata appena scritta, oppure si è inserito un carattere non corretto dopo aver inserito il carattere di comando, si può semplicemente riscrivere il comando in forma corretta partendo dal carattere di comando e la stringa precedentemente scritta viene scartata.

Il codice relativo alla comunicazione seriale da me progettata è riportato nel seguito:

```
void __attribute__((interrupt, no_auto_psv)) _U2RXInterrupt()
{
        IFS1bits.U2RXIF = 0; // manually cleared U2RX Interrupt flag
        buffer[next_in] = ReadUART2();

        if((buffer[next_in] == 'x') || (buffer[next_in] == 'y') ||
(buffer[next_in]          == 'i') || (buffer[next_in] == 'g'))
good_char = BUFFER_SIZE;

        if (good_char){
            if(good_char == BUFFER_SIZE)   next_in = 0;
            good_char --;
            if (buffer[next_in] == '#') {
                command = buffer[0]; /* the input string is organized
as: Cyyyyyy# C=command  yyyyyy=coordinata #=termine comando*/

                if(good_char == 0){
                    value = (unsigned long)((unsigned long)100000 *
(unsigned long)(buffer[1] - '0')
```

```
                        + (unsigned long)10000 * (unsigned
long)(buffer[2] - '0')
                        + (unsigned long)1000 * (unsigned
long)(buffer[3] - '0')
                        + (unsigned long)100 * (unsigned
long)(buffer[4] - '0')
                        + (unsigned long)10 * (unsigned
long)(buffer[5] - '0')
                        + (unsigned long) (buffer[6] - '0'));
                }

                switch(command){
                    case 'x' :
                        coordinate_X = value;
                        X_was_sent = 1;
                        if(flag_term)   WriteUART2('N');
                        else   WriteUART2('A');
                        break;

                    case 'y' :
                        coordinate_Y = value;
                        Y_was_sent = 1;
                        if(flag_term)   WriteUART2('N');
                        else   WriteUART2('A');
                        break;

                    case 'i':
                        flag_init_motor = 1;
                        good_char = 0;
                        if(flag_term)   WriteUART2('N');
                        else   WriteUART2('A');
                        break;

                    case 'g':
                        good_char = 0;
                        if(X_was_sent && Y_was_sent){
                            X_was_sent = Y_was_sent = 0;
                            flag_go_to = 1;
                        }
                        if(flag_term)   WriteUART2('N');
                        else   WriteUART2('A');
                        break;
                }
                next_in=0;
            }
            else if(next_in < (BUFFER_SIZE-1)) next_in ++;

            if(good_char == 0)  next_in = 0;
        }

}
```

Capitolo 7

Test del firmware

L'ambiente di sviluppo MPLAB della casa produttrice Microchip fornisce uno strumento molto importante come il *debug project* che permette il debug del firmware sviluppato dall'utente.

Grazie al debug project e al set-up sperimentale, montato sulla stazione di test, ho potuto testare ciascuna funzione presente nel codice mentre continuavo lo sviluppo del progetto, prima del test finale dell'intero firmware. Ciò mi ha permesso di mettere in luce e quindi di risolvere i vari errori commessi nello sviluppo delle varie funzioni.

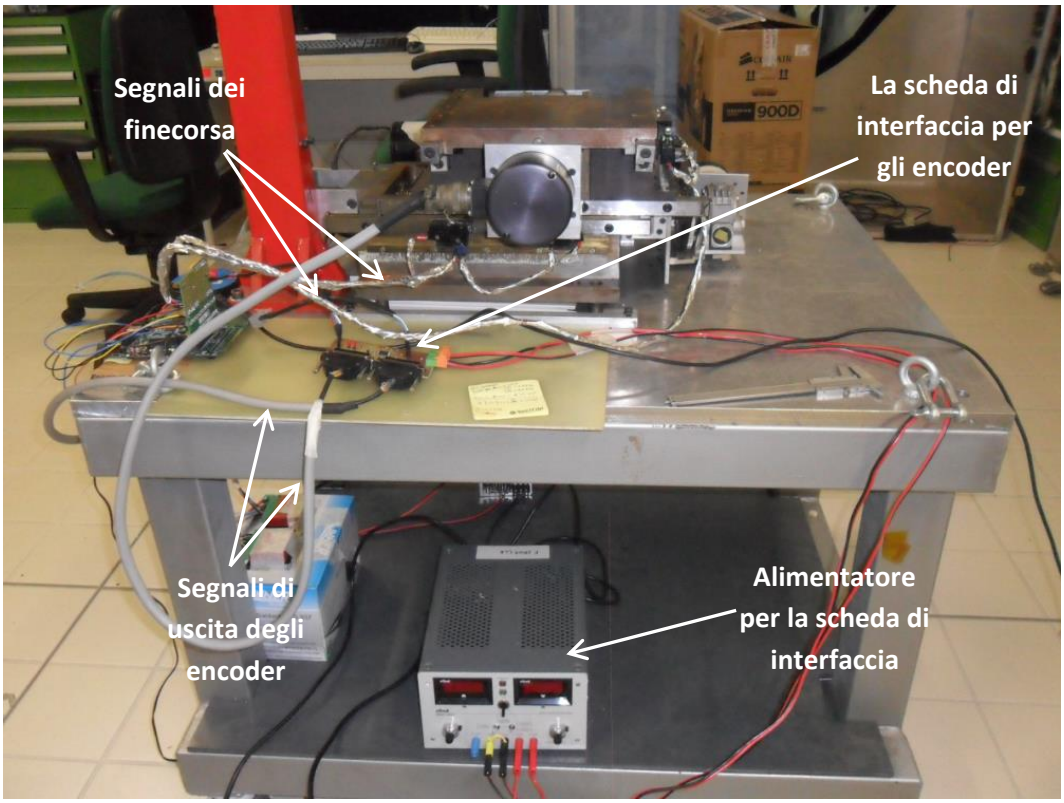

Figura 7.1: Set-up sperimentale utilizzato per il collaudo del firmware. Si possono osservare la scheda di interfaccia per gli encoder, la quale riceve in ingresso i segnali degli encoder e comunica in I^2C con la Explorer 16, l'alimentatore per la scheda di interfaccia che fornisce le tensioni di 5 V e 3.3 V, e infine i segnali dei quattro finecorsa inviati in ingresso al microcontrollore sulla Explorer 16.

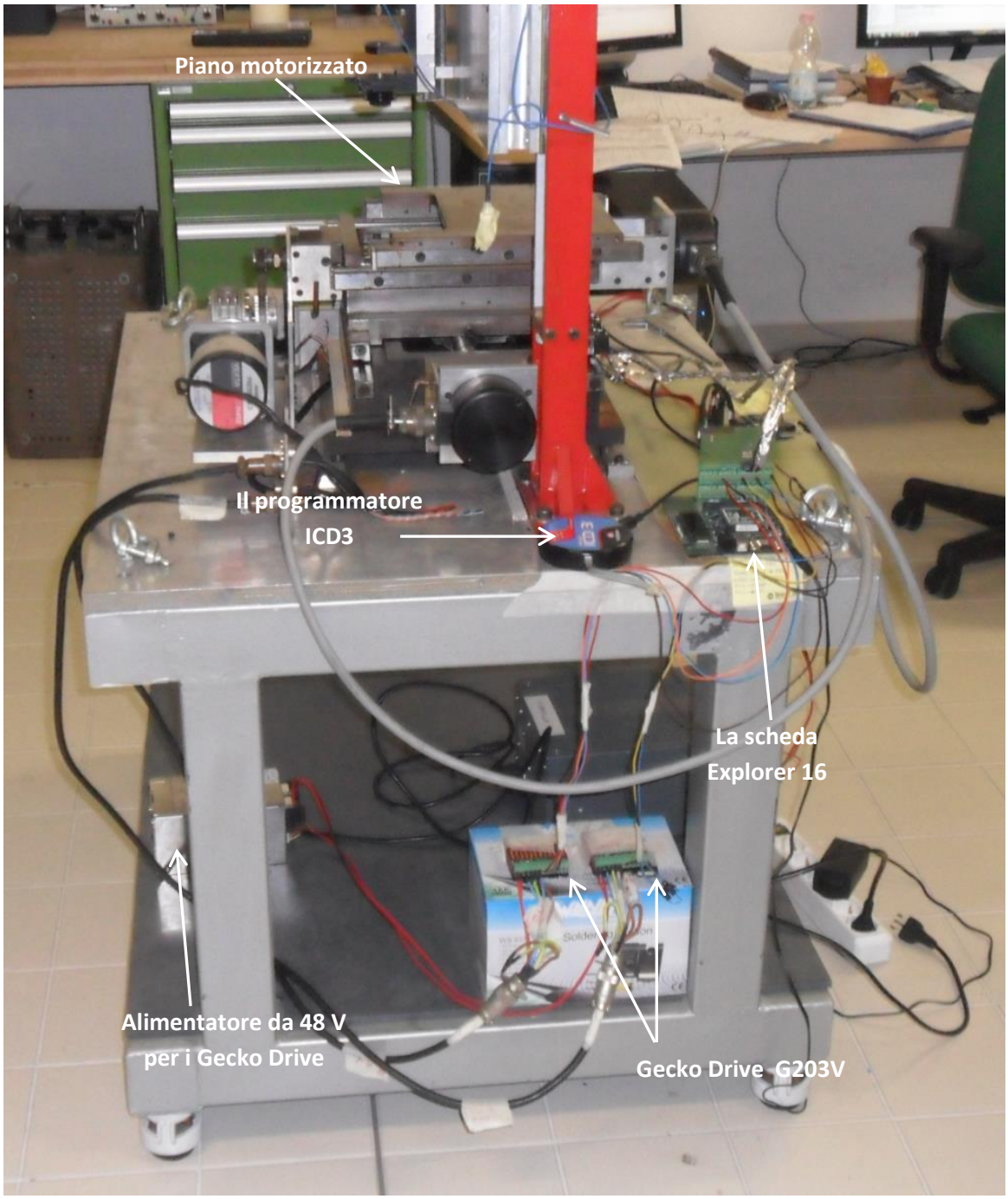

Figura 7.2: Set-up sperimentale utilizzato per il collaudo del firmware. Si può osservare la scheda Explorer 16 con a bordo il PIC24FJ12GA010 che controlla i due driver G203V mediante tre segnali: *STEP, DISABLE, DIR*; i quali a sua volta pilotano i due motori passo-passo facendo muovere il piano motorizzato. Si può osservare infine l'alimentatore da 48 V per i due driver G203V.

Una volta verificata la correttezza delle singole funzioni ho effettuato il test finale del sistema completo. Il test consiste nell'inserire il comando di inizializzazione (**i#**) da terminale PC e controllare se il piano si porta nell'origine. Successivamente, si

inseriscono le coordinate dove si vuole spostare il piano (Es. x008000# y008000#) ed infine il comando g#, per spostare il piano alle coordinate inserite. Per verificare il corretto spostamento del piano alle coordinate inserite, ho utilizzato un comparatore con precisione di 0.01 mm.

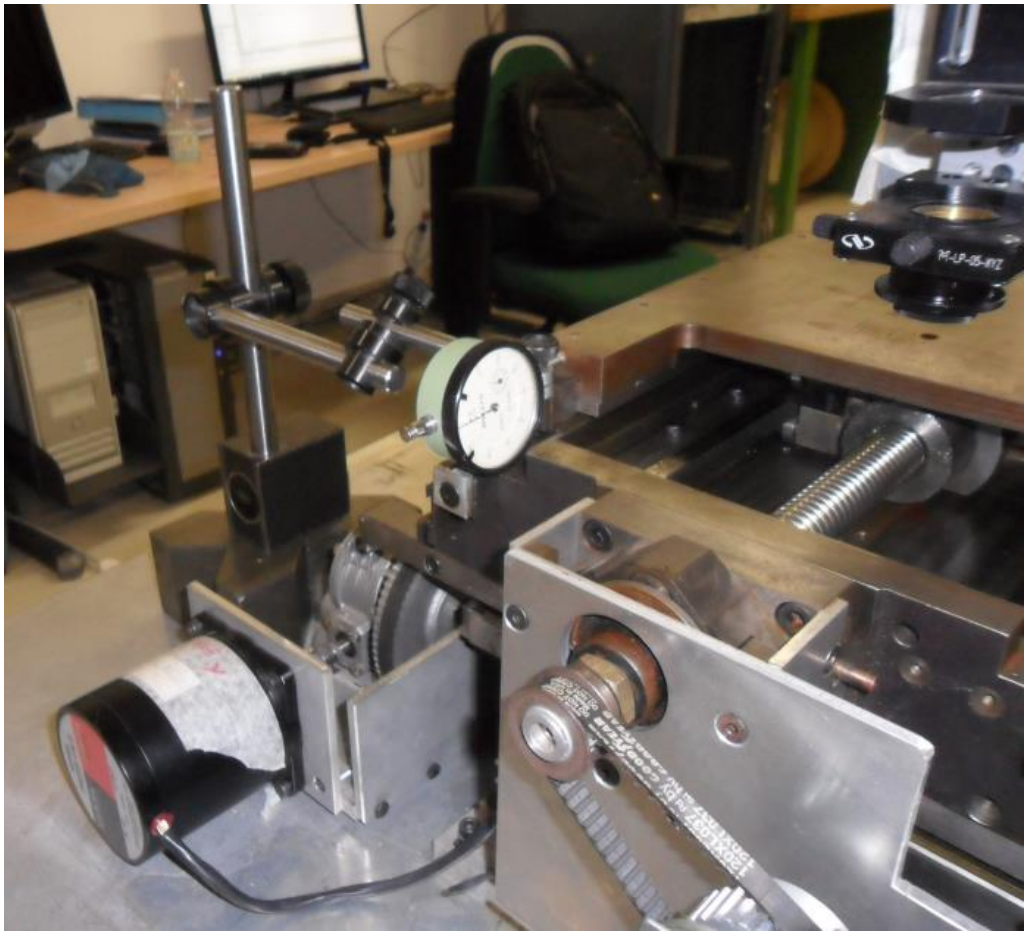

Figura 7.3: Misura dello spostamento lungo l'asse X del piano mediante il comparatore.

Una volta che il piano è nell'origine, con l'ausilio del braccio meccanico che sostiene il comparatore, come si vede dalla figura 7.3, ho posizionato la punta del comparatore perpendicolarmente al piano in modo tale che, la lancetta del comparatore vada a zero, vedi figura 7.4. Successivamente, digitando il comando g#, per lo spostamento del piano alla coordinata X = 8 mm, il piano inizia a muoversi spostandosi in direzione del comparatore fino a quando arriva alla coordinata inserita. Lo spostamento misurato dal comparatore rispetto all'origine ci dice a quale coordinata è giunto il piano e quindi permette di verificare se effettivamente è giunto alla coordinata X = 8 mm.

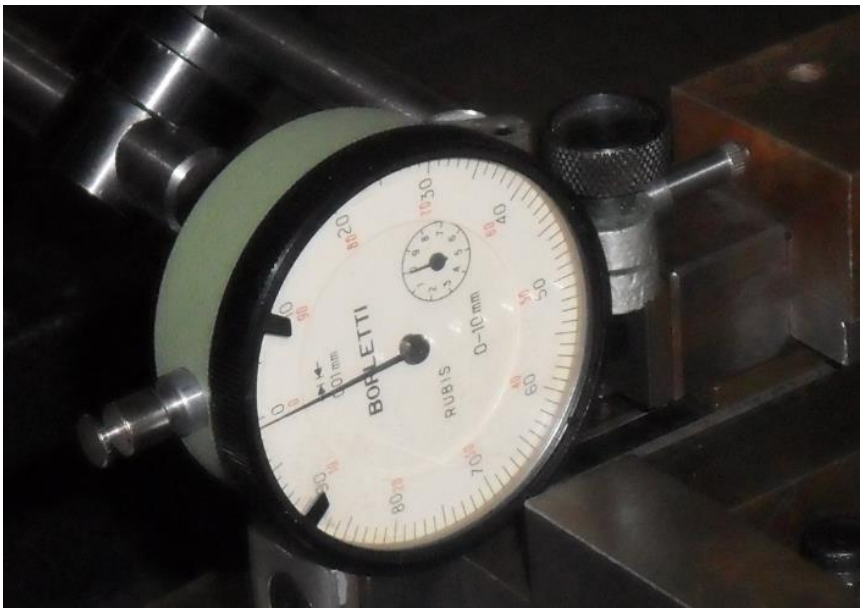

Figura 7.4: Posizione del comparatore prima dello spostamento. Si osserva la punta del comparatore posizionata perpendicolarmente al piano motorizzato e le due lancette del comparatore posizionate sullo zero.

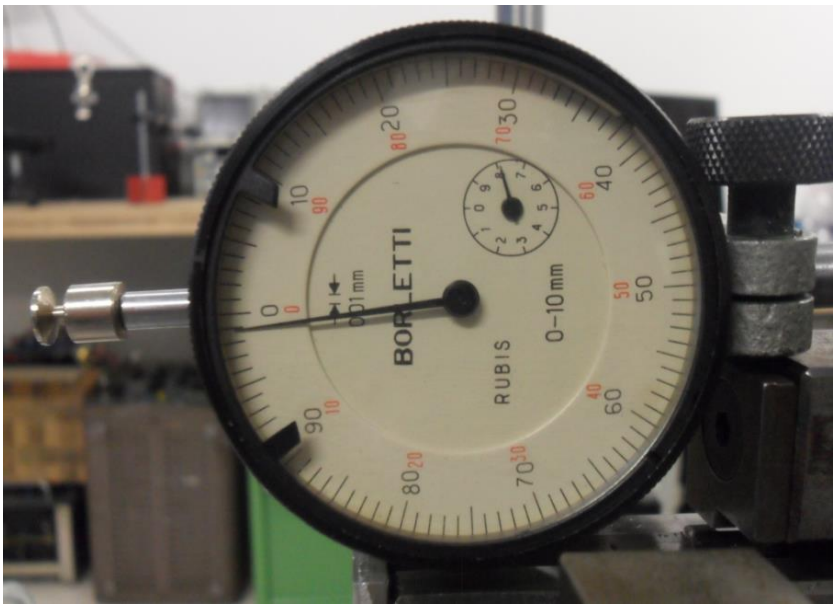

Figura 7.5: Posizione del comparatore dopo lo spostamento. Si osserva che la lancetta piccola del comparatore, che indica i millimetri, è posizionata sul numero 8, mentre la lancetta grande, che indica i centesimi di millimetro, è posizionata in prossimità dello zero. Quindi il comparatore ha misurato uno spostamento di 8 mm.

Come si può osservare dalla figura 7.5, il piano si è spostato dall'origine proprio di 8 mm come la coordinata inserita da terminale.

Nello stesso modo si è misurato lo spostamento lungo l'asse Y, ottenendo gli stessi identici risultati di sopra.

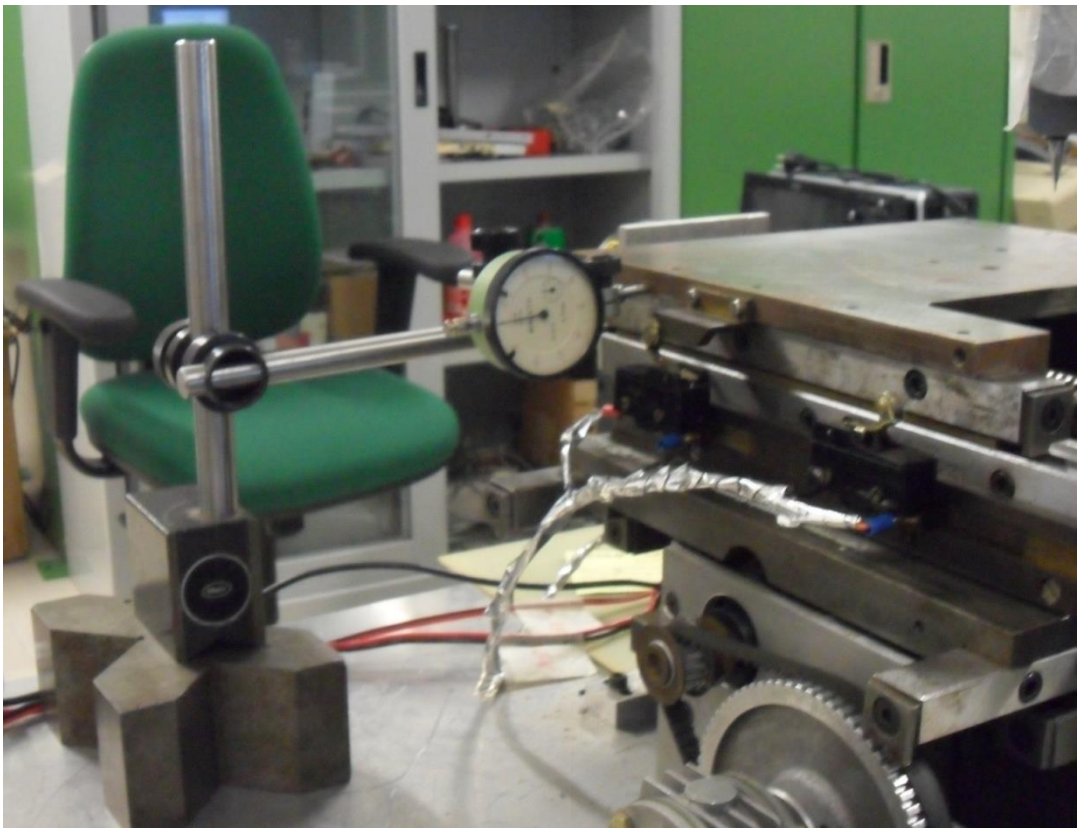

Figura 7.6: Misura dello spostamento lungo l'asse Y del piano mediante il comparatore

Naturalmente ho effettuato numerose prove spostando il piano da una coordinata ad un'altra e misurando con il calibro spostamenti superiori ad 1 cm (non misurabili con il comparatore perché ha un fondo scala di 10 mm). Abbiamo ottenuto in tutti i casi spostamenti molto precisi alle coordinate inserite da terminale.

Questo ci conferma il corretto funzionamento del firmware, quindi ci possiamo ritenere soddisfatti dei risultati ottenuti e soprattutto della precisione che si ottiene negli spostamenti.

Capitolo 8

Conclusioni e prospettive

Ho svolto il mio lavoro di Tesi presso la sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare nel periodo compreso tra marzo 2014 e settembre 2014.

Con il mio lavoro ho contribuito attivamente all'avvio della realizzazione di una stazione di test automatizzata per i fotosensori utilizzati nel calorimetro elettromagnetico dell'esperimento Mu2e, che è attualmente in costruzione al Fermi National Accelerator Laboratory (Fermilab) negli Stati Uniti.

Il calorimetro elettromagnetico Mu2e, che sarà costituito da una matrice di cristalli scintillanti letti da fotosensori (circa 3720) svolge un ruolo molto importante nell'esperimento, e cioè quello di misurare energia, posizione e tempo d'impatto delle particelle delle quali il sistema di tracciatura ha ricostruito la traiettoria.

Mi sono occupato dello sviluppo del firmware per il microcontrollore PIC24FJ12GA010 per la movimentazione e il controllo di precisione del piano sul quale sarà posizionato la matrice di fotosensori da testare.

All'inizio ho dovuto prendere familiarità con il PIC e con l'ambiente di sviluppo MPLAB.

Per realizzare la stazione di test si è utilizzata la parte meccanica di una vecchia stazione realizzata all'INFN di Pisa nell'anno 1999, per il test dei moduli in silicio per l'esperimento CDF di Fermilab. Quindi, ho dovuto rifare l'elettronica della stazione di test e montare un set-up sperimentale per lo sviluppo del firmware.

Ho dovuto anche progettare e realizzare una scheda di interfaccia per la lettura dei due encoder della stazione di test ed ho sviluppato il firmware per il microcontrollore della scheda, in modo tale che comunichi in I^2C con la Explorer 16.

I test del firmware ne hanno dimostrano il corretto funzionamento. Abbiamo dimostrato che con questo sistema si riescono ad ottenere spostamenti del piano molto precisi con un errore massimo di $10\ \mu m$ a fronte di spostamenti minimi richiesti di $500\ \mu m$. Nel

complesso ci possiamo ritenere soddisfatti dei risultati ottenuti. Ovviamente la realizzazione della stazione di test è solo all'inizio. Oltre alla parte della movimentazione da me realizzata, in futuro dovrà essere sviluppato il sistema per il pilotaggio della sorgente di luce utilizzata per stimolare il fotosensore.

Il guadagno dei fotosensori è fortemente dipendente dalla temperatura, per cui la stazione di test dovrà essere posizionata all'interno di una camera termica ed equipaggiata con sensori di temperatura utilizzati per il controllo ad anello.

I fotosensori sotto test saranno alimentati tramite uno strumento Keythley 6487 equipaggiato anche con un modulo picoamperometro, che è quindi in grado di tracciare la curva I-V di risposta al variare dei parametri di luce e di temperatura.

Dovrà infine essere sviluppato un software di controllo, utilizzando LabView, che permetta di gestire l'intero sistema in remoto da un PC.

Bibliografia

[1] Gianantonio Pezzulo, Calor 2014, Giessen, "*Progress status for the Mu2e Calorimeter system* ", April 2014.

[2] Alessandra Lucà, Tesi di Laurea Magistrale in Fisica, Università degli Studi di Roma Tor Vergata, "*Simulation and test of the electromagnetic calorimeter for the Mu2e experiment* ", 2011.

[3] Fermi National Accelerator Laboratory, Batavia, IL 60510. "Mu2e Technical Design Report", June 2014.

[4] Ivano Sarra, Tesi di Laurea Specialistica in Fisica, Indirizzo Elettronica e Cibernetica, Università degli Studi di Roma Tor Vergata, "*Caratterizzazione di fotosensori al silicio connessi a fibre ottiche e scintillatori per sviluppi di calorimetria elettromagnetica* ", 2008.

[5] http://nora.ing.unibs.it/riservato/com_ottiche/materiale/OptCom3c.pdf.

[6] http://www.iet.unipi.it/p.bagnoli/rivelatori.doc.

[7] http://www.ketek.net/products/sipm-technology/working-principle/.

[8] HONEYWELL S&C, Microinterruttore Serie BZ - 2RW822, "*Technical Data Sheet* "; http://www.farnell.com/datasheets/109065.pdf.

[9] Wikipedia, "*Motore passo – passo* "; http://it.wikipedia.org/wiki/Motore_passo-passo.

[10] GeckoDrive Motor Controls, "*G203V Stepper Drive Data Sheet* "; http://www.geckodrive.com/geckodrive-step-motor-drives/g203v.html.

[11] https://shop.omniray.ch/data/catalog/documents/uploads/410_datasheets/ELCIS.pdf.

[12] http://www.vfioraso.it/5anno/documenti_pdf/trasduttori/07_trasduttori_Encoder.pdf.

[13] Microchip, "*PIC24FJ128GA010 Family Data Sheet*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39747F.pdf.

[14] Microchip, "*dsPIC30F3014/4013 Data Sheet*"; http://ww1.microchip.com/downloads/en/DeviceDoc/70138G.pdf.

[15] Microchip, "*Section 12. I/O Ports with Peripheral Pin Select (PPS) - PIC24F FRM*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39711b.pdf.

[16] Microchip, "*Section 14. Timers - PIC24F FRM*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39704a.pdf.

[17] Microchip, "*Section 16. Output Compare - PIC24F FRM*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39706a.pdf.

[18] Microchip, "*Section 24. Inter-Integrated Circuit (I^2C^{TM}) - PIC24F FRM*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39702a.pdf.

[19] Microchip, "*Section 21. Universal Asynchronous Receiver Transmitter (UART)*"; http://ww1.microchip.com/downloads/en/DeviceDoc/39708B.pdf.

[20] ON Semiconductor, "*BS138LT1G data sheet*"; http://www.mouser.com/ds/2/308/BSS138LT1-D-104046.pdf.

[21] Referenza SINDRUM II: W.Beart et al., Eur.Phys. J. C47, 337 (2006)