UNIVERSITÀ DEGLI STUDI DI CATANIA
DIPARTIMENTO DI FISICA E ASTRONOMIA
DOTTORATO DI RICERCA IN FISICA - XXIX CICLO

DAVIDE AGATINO LICCIARDELLO

SVILUPPO DI UN SISTEMA DI CONTROLLO PER LO SPETTROGRAFO CAOS

PHD THESIS

SUPERVISOR:
CHIAR.MO PROF. FRANCO LEONE

Alla mia famiglia…

Indice

Indice

Indice

Introduzione

Lo scopo della mia attività è quello di realizzare una nuova interfaccia tra il telescopio da 91 cm e lo spettropolarimetro CAOS, situati all'osservatorio M.G. Fracastoro di Serra La Nave (SLN).

La nuova interfaccia al telescopio dello spettropolarimetro CAOS nasce da due diverse esigenze:

1. Rendere più efficiente il sistema di autoguida.
2. Aumentare il grado di automazione dell'interfaccia al telescopio.

I passi seguiti per la realizzazione del progetto sono i seguenti:

1) Valutazione dei requisiti da raggiungere in termini di precisione da parte di ogni singolo motore e conseguente scelta dell'hardware sia relativo ai moduli Beckhoff sia relativo ai motori stessi.
2) Implementazione delle procedure "ad hoc" che permettano ad ogni singolo sottosistema di espletare le funzioni definite per la nuova interfaccia.
3) Realizzazione di un'interfaccia grafica in ambiente TwinCAT 3 o pannello di visualizzazione interattivo per ogni singolo motore.
4) Implementazione di un'interfaccia grafica utente (GUI) in linguaggio Java per la realizzazione di un Client per ogni singolo motore.

Nel **primo capitolo** vengono descritte le caratteristiche di uno spettrografo in generale e di seguito le specifiche dello spettrografo CAOS, entrando nei dettagli per ogni singola parte di cui lo spettrografo è composto. Poi viene descritta l'attuale interfaccia tra il telescopio e lo spettrografo CAOS e le sue funzioni, infine viene proposta la descrizione di una nuova interfaccia con i suoi sottosistemi.

Nel **secondo capitolo** vengono illustrate le funzioni delle movimentazioni della nuova interfaccia e l'hardware utilizzato per ogni singolo sottosistema.

Inoltre verranno dati dei cenni sull'ambiente TwinCAT 3 utilizzato per lo sviluppo del software Server e sul protocollo ADS, messo a disposizione della ditta Beckhoff per mettere in comunicazione un Client, realizzato con un linguaggio ad alto livello, con il Server TwinCAT.

Nel **terzo capitolo** vengono descritte le tecniche adoperate per la configurazione dei parametri elettrici e meccanici relativi ad ogni singola movimentazione che compone un sottosistema.

A partire da tali configurazioni è stata realizzata, per ogni singolo motore, un' interfaccia grafica ingegneristica in ambiente TwinCAT 3, che permette di conoscere le informazioni sullo stato del motore e di verificare le funzioni implementate con il linguaggio IEC 61131-3. In un scenario reale di comunicazione a distanza, in rete, tale software rappresenta la parte "Server".

Infine è stata realizzata un Client grafico in Java, per ogni singolo motore, con lo scopo di testare il corrispondente Server e fornire un primo interfacciamento con il Server stesso.

CAPITOLO 1

Lo spettrografo

1 Introduzione

Nella prima parte di questo capitolo si introdurranno le generalità di uno spettrografo seguite dalla descrizione delle singole componenti dello spettropolarimetro CAOS e della loro configurazione.

Nella seconda parte si analizzerà l'attuale interfaccia al telescopio, descrivendo le modifiche da apportare e proponendo una nuova interfaccia.

1.1 Descrizione di uno spettrografo

L'analizzatore astronomico di base della luce è la spettrografo, un dispositivo ottico in grado di creare immagini di una sorgente a diverse lunghezze d'onda. Come si può vedere in figura 1.1, un tipico spettrografo consiste di una fenditura (slit) di ingresso collocata sul fuoco del telescopio o su un'immagine di esso, un collimatore che intercetta il fascio divergente del telescopio, un elemento dispersore che può essere un prisma o un reticolo (grating) e una camera che concentra la luce dispersa sul rivelatore. La distanza tra la fenditura e il collimatore è la lunghezza focale del collimatore mentre la distanza tra la camera e lo spettro focalizzato è la lunghezza focale della camera. L'elemento disperdente più efficiente è il reticolo di diffrazione schematizzato in figura 1.2. Esso è costituito solitamente da una lastra di vetro sulla cui superficie, rivestita di alluminio, è incisa una trama di linee parallele, uguali ed equidistanti, a distanze dello stesso ordine con la lunghezza d'onda della luce. La distanza tra le linee è detta "passo del reticolo". Nella pratica i reticoli sono solitamente caratterizzati dal numero di incisioni per unità di lunghezza; tale parametro in genere viene espresso in linee per millimetro (lines/mm).

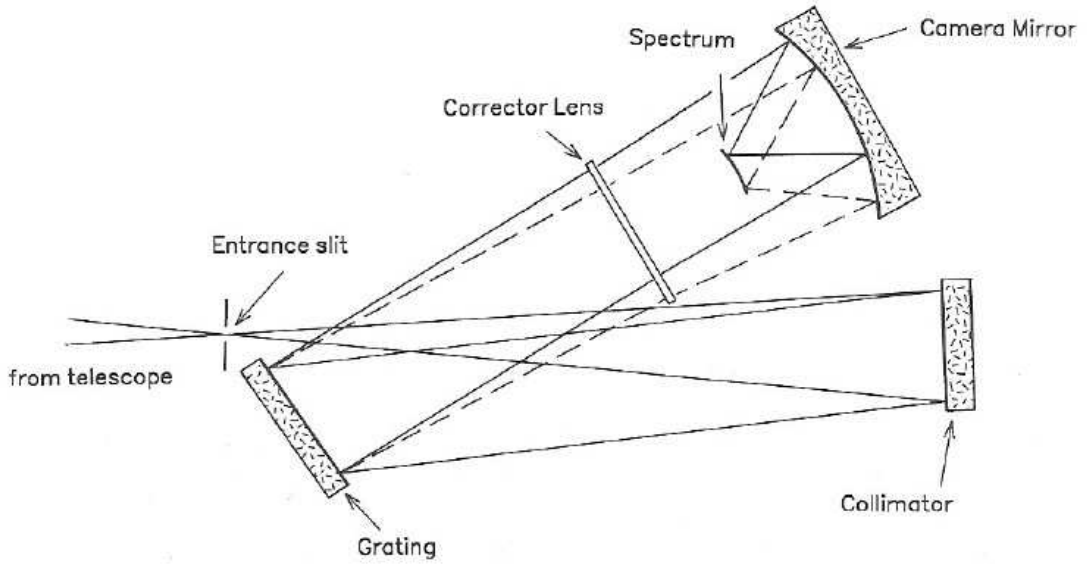

Figura 1.1 – Layout per un generico spettrografo

Si può dimostrare che le onde risultanti dietro il reticolo possono essere scritte come la trasformata di Fourier della funzione di trasmissione del reticolo $G(x)$:

$$G(\theta) = F_0 \int G(x) e^{2\pi i x\theta}\, dx \tag{1.1}$$

dove

$$G(x) = B_1(x) * III(x) B_2(x) \tag{1.2}$$

$B_1(x)$ è una funzione che rappresenta la trasmissione attraverso una singola fenditura, con larghezza *b*. La funzione $B_2(x)$ rappresenta la larghezza totale del reticolo *W*. La funzione $III(x)$ è una funzione di Shah e rappresenta la distanza dei tratti. I differenti ordini, *n*, nascono da varie linee e la larghezza della singola fenditura, *b*, imposta l'inviluppo

della diffrazione mentre *W* imposta la larghezza del valore massimo della singola interferenza. Assumendo che

$$III(\theta) = \sum \delta\left(\theta - \frac{n}{d}\right) \tag{1.3}$$

si ha:

$$g(\theta) = \sum_n \frac{W\ sin\pi\left(\theta - \frac{n}{d}\right)W}{\pi(\theta - n/d)W} \frac{b\ sin\pi\theta b}{\pi\theta b} \tag{1.4}$$

Il quadrato di questa ampiezza d'onda è proporzionale all'intensità della luce. Si può dimostrare che ogni valore massimo corrisponde ad un valore di ordine *n*, seguendo l'equazione del reticolo:

$$\frac{n\lambda}{d} = \theta = sin\alpha + sin\beta \tag{1.5}$$

Si può dimostrare anche che la risoluzione cromatica può essere scritta:

$$\Delta\lambda = \frac{\lambda d}{Wn} \tag{1.6}$$

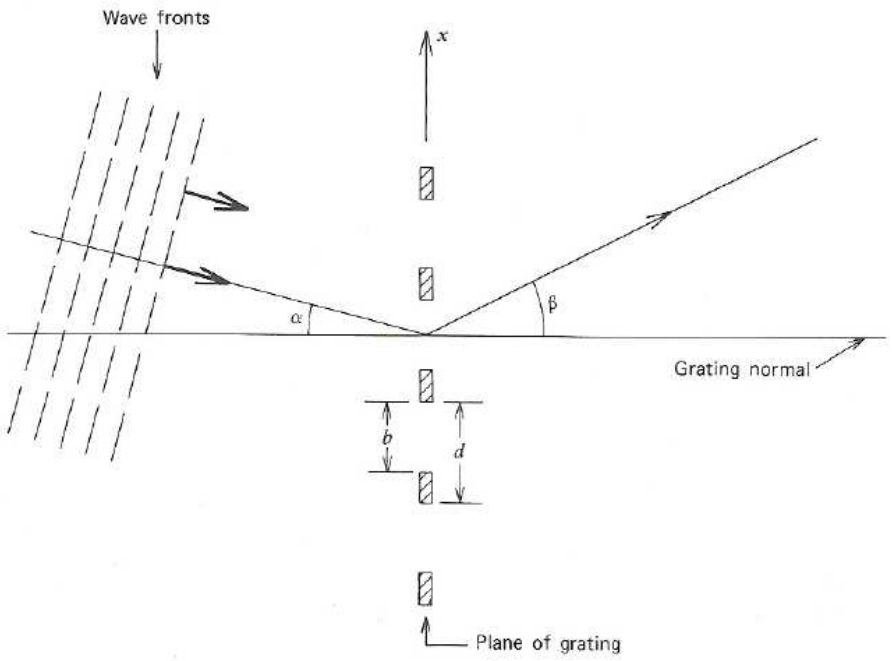

Figura 1.2: Reticolo semplice di diffrazione

Si consideri l'angolo β che è l'angolo sotto cui guardare la radiazione emergente. Questo tipo di reticolo viene utilizzato per basse risoluzioni spettrali o per bassi ordini. Per alte risoluzioni non conviene utilizzarlo per due opposte esigenze, infatti da un lato l'angolo β deve essere grande per poter avere una elevata risoluzione ma, allo stesso tempo, per avere la massima luminosità occorre che β sia nullo.

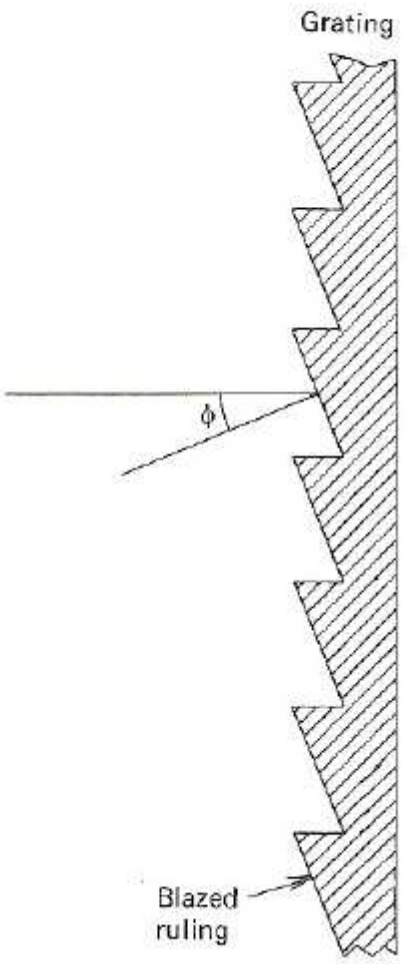

Figura 1.3: Reticolo di blaze

Il rapporto $\lambda/\Delta\lambda$ è chiamato potere risolutivo o risoluzione ed è indipendente dalla lunghezza d'onda.

Fissato α=0, si noti che i reticoli semplici di trasmissione hanno il valore massimo più grande, per β=0, che comporta dall'equazione 1.5, $\theta=0$, dove non c'è dispersione cromatica pertanto non può essere eseguito nessun tipo di studio spettroscopico. Per evitare questo, l'inviluppo della diffrazione è traslato rispetto al modello di interferenza con una rotazione, infatti per avere il massimo della luminosità non a β=0 ma dove la risoluzione è maggiore, occorre traslare la figura di diffrazione o interferenza. Poiché tale figura rappresenta la trasformata di Fourier del reticolo a trasmissione, è possibile sfruttare una proprietà della trasformata di Fourier secondo la quale, apportando una rotazione in un

dominio si ottiene una traslazione nel dominio corrispondente. Con tale trasformata si sposta il massimo valore della luminosità nella direzione desiderata.

Per ottenere tale risultato si utilizza una superficie riflettente; le fenditure del reticolo di trasmissione sono sostituite da superfici inclinate, come si può vedere in figura 1.3. Le singole linee (rulings) riflettenti sono chiamate superfici (facets). L'angolo delle superfici è impostato dal produttore e assume valore standard; chiamiamo R4 l'angolo di inclinazione ϕ rispetto al fascio incidente la cui tangente è pari a 4 con $\phi = 76°$ ($tan\ \phi = 4$) e R2 quello con $\phi = 63.5°$ ($tan\ \phi = 2$). La luce all'uscita dello spettrografo è descritta dalla funzione di Blaze, [2]:

$$I(\beta) = \left[\frac{\sin\left[\frac{n\pi b}{d}\left(cos\phi - \frac{sin\phi}{\tan\frac{1}{2}(\alpha+\beta)}\right)\right]}{\left[\frac{n\pi b}{d}\left(cos\phi - \frac{sin\phi}{\tan\frac{1}{2}(\alpha+\beta)}\right)\right]} \right]^2 \tag{1.7}$$

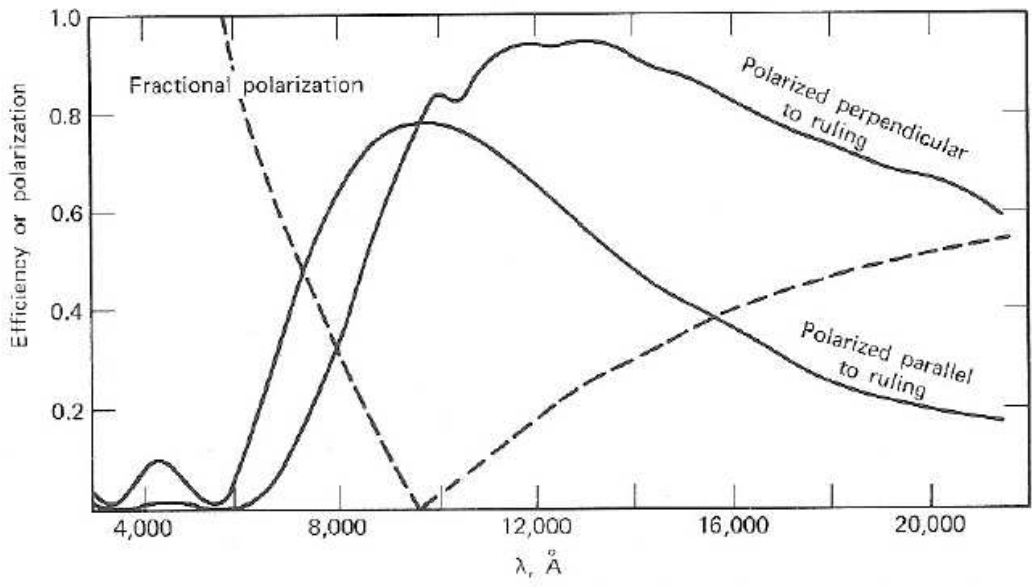

Figura 1.4: Reticolo di blaze - Efficienza di polarizzazione

L'efficienza dipende dalla polarizzazione, come si può vedere dalla figura 1.4.

1.1.1 Gli spettrografi echelle

Negli anni passati gli spettrografi echelle cross-dispersed sono stati usati per certi tipi di lavori. Il vantaggio offerto dal reticolo echelle è il suo elevato angolo di dispersione, $d\beta/d\lambda$, derivante dall'utilizzo di un numero elevato di ordini, $n \sim 100$.

Questo particolare tipo di spettrografo utilizza una combinazione di prisma e reticolo per la dispersione della luce. Il sistema è detto echelle, dal francese "scala", in quanto l'elemento disperdente ha una sezione a forma di scaletta. Il fascio incidente, dopo aver attraversato un collimatore, viene diffratto dall'elemento disperdente, e quindi viene indirizzato sul CCD attraverso un particolare sistema ottico. In questo modo, lo spettro sul CCD non viene a trovarsi lungo una sola riga o colonna, ma viene diviso per colonne o per righe in una serie di spettri detti ordini, in cui l'estremità finale dell'ordine precedente si sovrappone in lunghezza d'onda all'estremità iniziale dell'ordine successivo. In questo modo si ha uno spettro diviso lungo n colonne (o righe) del CCD, come si vede in figura 1.5, pertanto si ottiene un aumento della dispersione dello spettro stesso.

I vantaggi scientifici dello spettrografo echelle includono una elevata purezza spettrale, (si veda l'equazione 1.6 con n elevato) e una larga copertura di lunghezze d'onda su ciascuna esposizione. Il secondo punto è importante per effettuare studi molto dettagliati, relativi all'abbondanza di una specie chimica dell'oggetto osservato dove devono essere misurate un elevato numero di linee largamente separate.

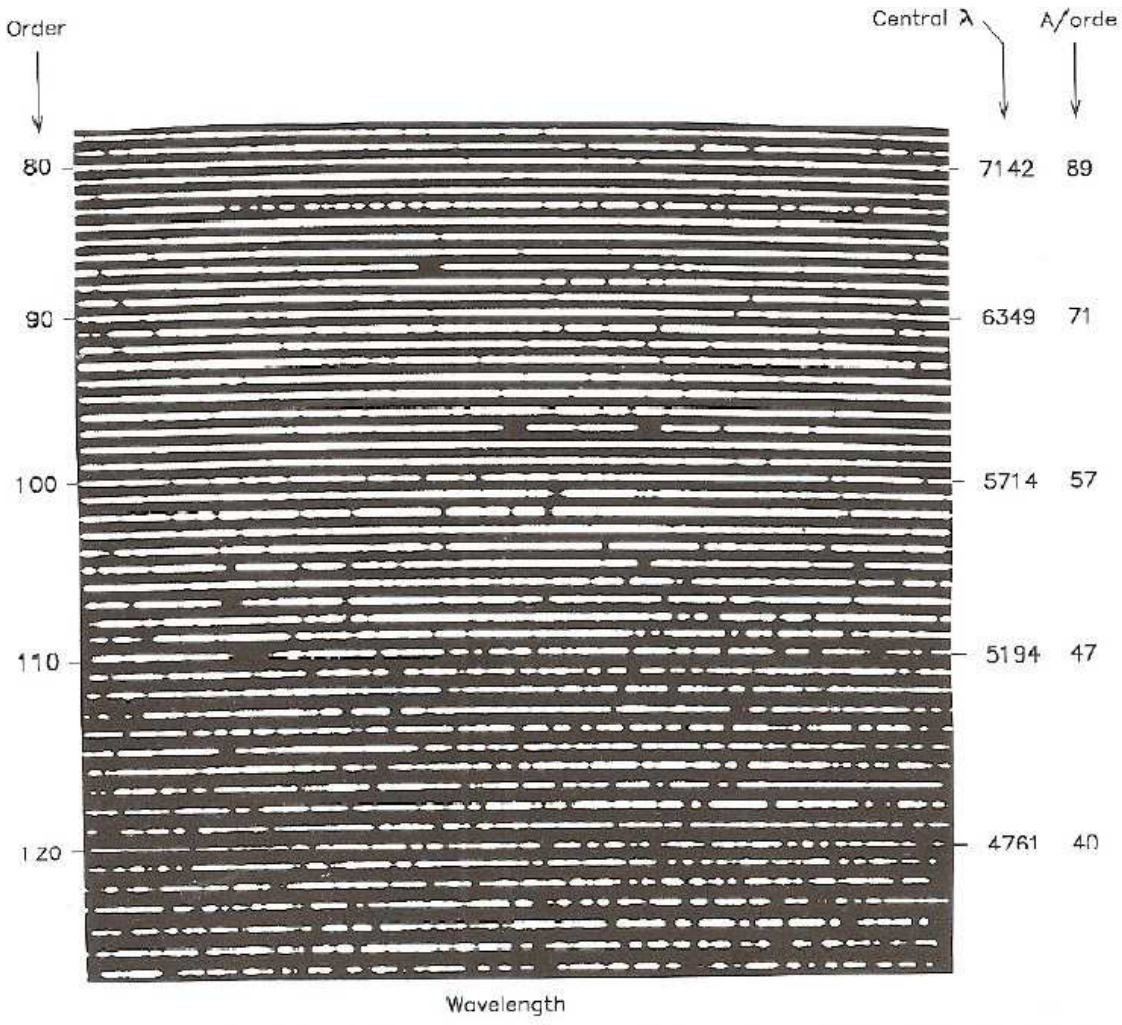

Figura 1.5: Echellogramma

1.1.2 I polarimetri

Per misurare lo stato di polarizzazione della luce, sono presenti vari metodi tutti basati su misure di intensità a differenti orientazioni relative dei componenti del polarimetro o a differenti orientazioni della strumentazione relativamente alla sorgente di luce. In un polarimetro generalizzato, una lamina di ritardo Δ, il cui asse principale forma un angolo β rispetto ad un sistema di riferimento, è combinata con un polarizzatore il cui asse di trasmissione forma un angolo α con il sistema di riferimento. La lamina di ritardo introduce un ritardo nella fase dell'onda cambiando il piano di polarizzazione, mentre il polarizzatore può scomporre le due componenti di un'onda, dette ordinaria e straordinaria, o può non fare emergere una componente.

1.1.2.1 Il polarizzatore

Come esempio di polarizzatore si consideri la plate di Savart.

La plate di Savart è un dispositivo in genere costituito da calcite o quarzo, il quale separa spazialmente due stati di polarizzazione lineare ortogonali. Tale dispositivo viene realizzato mediante due piastre di uguale spessore di materiale birifrangente, cioè dotato di un asse ottico; ciascuna piastra presenta un suo asse ottico orientato a 45° rispetto alla normale alla superficie e tali assi sono ruotati di 90° l'uno rispetto all'altro. Il fascio di luce che entra nella prima piastra è diviso nei due suoi stati di polarizzazione, dati dai due raggi polarizzati, uno detto ordinario e l'altro detto straordinario. Il raggio ordinario si propaga lungo la normale, mentre il raggio straordinario viene deviato su di un piano parallelo ad un bordo della lamina di un certo angolo che dipende dall'indice di rifrazione. Tali raggi entrano nella seconda piastra. Il raggio ordinario nel primo cristallo diventa il raggio straordinario nel secondo cristallo, pertanto tale raggio viene deviato all'interno della seconda piastra in una direzione perpendicolare rispetto alla direzione di deviazione nel primo cristallo. Il raggio straordinario nel primo cristallo diventa il raggio ordinario nel secondo, pertanto tale raggio si propaga lungo la direzione della normale alla superficie della seconda piastra.

In questo modo i due fasci emergenti sono fisicamente separati ma entrambi paralleli al raggio d'entrata e possono essere utilizzati sullo stesso piano focale. Si ottengono due fasci emergenti separati della distanza d pari alla diagonale di un quadrato avente per lato lo spostamento introdotto dalla singola lamina. La differenza di cammino ottico fra i due fasci è nulla in quanto la differenza di cammino nella prima lamina viene bilanciata da una differenza di cammino complementare nella seconda lamina, si veda la figura 1.6.

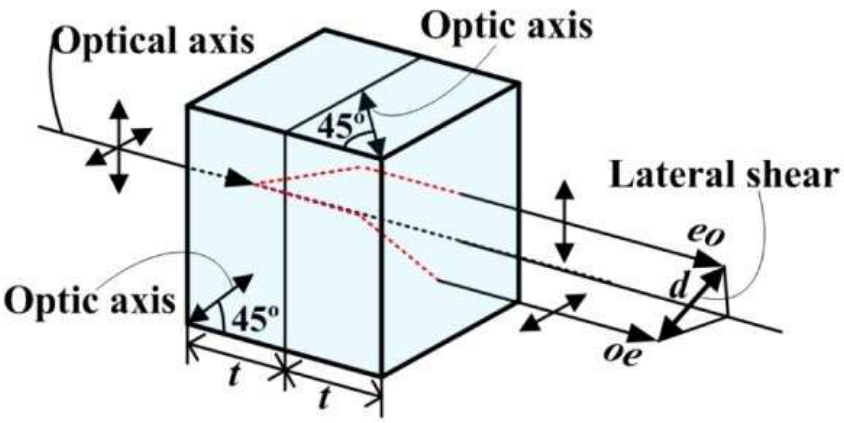

Figura 1.6: Schema ottico di una semplice plate di Savart

1.1.2.2 Le lamine di ritardo

Le lamine di ritardo sono costituite da materiali birifrangenti che presentano un indice di rifrazione differente nei due assi ortogonali. Questa proprietà di birifrangenza introduce una differenza di velocità tra la luce polarizzata lungo l'asse principale veloce della lamina, e la componente di luce polarizzata lungo l'asse principale lento. L'asse principale veloce della lamina ha un indice di rifrazione inferiore, dando luogo ad una maggiore velocità per la luce polarizzata in questa direzione. Al contrario, l'asse lento ha un indice di rifrazione più elevato, con conseguente minore velocità per la luce polarizzata in questa direzione. Quando la luce passa attraverso una lamina, questa differenza di velocità comporta una differenza di fase tra le due componenti di polarizzazione ortogonali. Lo sfasamento effettivo dipende dalle proprietà del materiale, dallo spessore della lamina, dalla lunghezza d'onda del segnale, e può essere descritto come[1]:

$$\Delta\phi = \frac{2\pi d\ (n_1 - n_2)}{\lambda} \tag{1.6}$$

dove è n_1 l'indice di rifrazione lungo l'asse lento, n_2 è l'indice di rifrazione lungo l'asse veloce, d è lo spessore della lamina e λ è la lunghezza d'onda del segnale.

[1] https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=7234

Esistono differenti tipi di lamine di ritardo che si differenziano per i materiali di cui sono composte e per la copertura in lunghezze d'onda λ.

Le lamine acromatiche forniscono un ritardo di fase che relativamente non dipende dalla lunghezza d'onda in un largo intervallo spettrale, mentre le lamine super acromatiche forniscono un ritardo di fase quasi interamente indipendente dalla lunghezza d'onda per un intervallo spettrale più ampio rispetto alle acromatiche. Le lamine di ordine zero e quelle multi ordine forniscono un ritardo che è fortemente dipendente dalla lunghezza d'onda.

1.2 Lo spettrografo CAOS

CAOS (Catania Astrophysical Observatory Spectropolarimeter) è uno spettrografo echelle con cross-dispersore del tipo white-pupil capace di una risoluzione spettrale fino a R=55000 nell'intervallo 376-1121 nm [3]. Tale spettrografo, attualmente in uso, viene utilizzato per misure di abbondanza chimica degli elementi presenti nelle stelle [9], per la misura del campo magnetico stellare e, in generale, per gli studi della fisica stellare. La figura 1.7 mostra la rappresentazione opto-meccanica dello strumento.

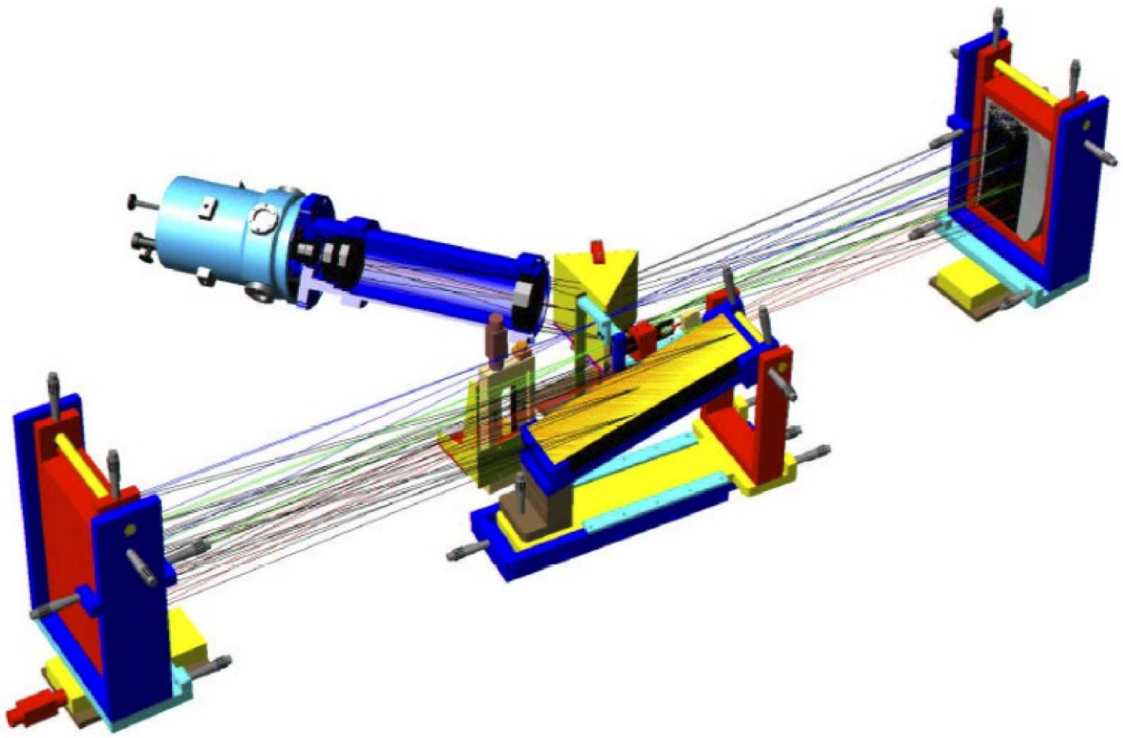

Figura 1.7: Rappresentazione schematica di CAOS

1.2.1 Schema ottico di CAOS

In un tipico spettrografo astronomico il diametro di apertura della camera è più grande rispetto alla dimensione del dispersore ed incrementa con la dimensione del campo, pertanto è impossibile avere un alto rapporto di apertura per la camera se viene utilizzato un reticolo molto grande e dispersivo. Per evitare questa limitazione, il dispersore è

posto sul fascio collimato e ogni fascio monocromatico interseca l'altro in una "white-pupil" all'ingresso della camera. Analogamente agli spettrografi più moderni, per esempio UVES (Ultraviolet and Visual Echelle Spectrograph) @ ESO (European Southern Observatory) [5], per ottenere la massima risoluzione possibile, lo schema ottico di CAOS si basa su una configurazione "white-pupil". La figura 1.8 mostra il layout ottico di CAOS.

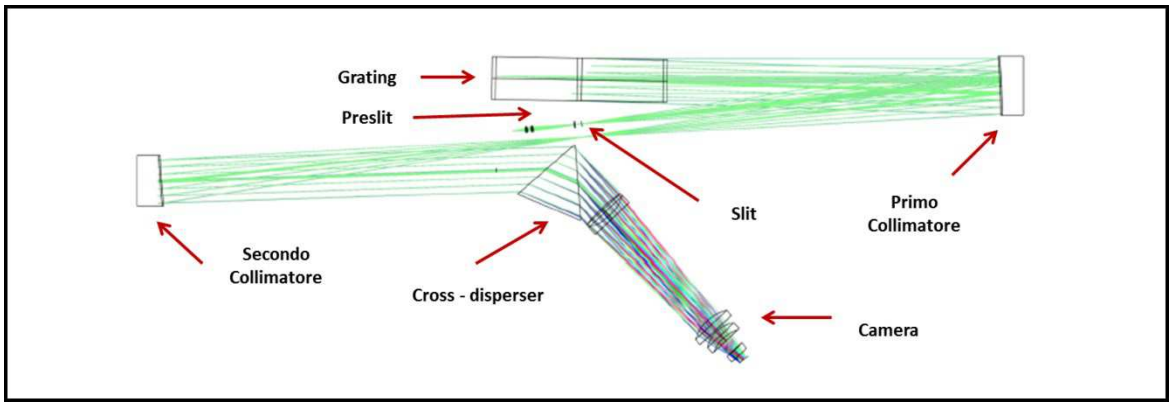

Figura 1.8: Schema ottico dello spettrografo CAOS

Lo spettrografo, il cui disegno ottico è mostrato in figura 1.8, è montato su di un banco ottico (figura 1.9) all'interno di un box stabilizzato in temperatura (rms di 0.01K). La luce dalla fibra ottica attraversa un'ottica di preslit che produce un fuoco sulla fenditura dello spettrografo. Il fascio in uscita dalla fenditura arriva su uno specchio, viene collimato sul reticolo disperso (cross-disperser) ed inviato nuovamente al collimatore. Lo specchio produce un fuoco ed il fascio viene ricollimato da un secondo specchio sul prisma dispersore. Da qui il fascio entra in una camera e viene messo a fuoco sul rivelatore.

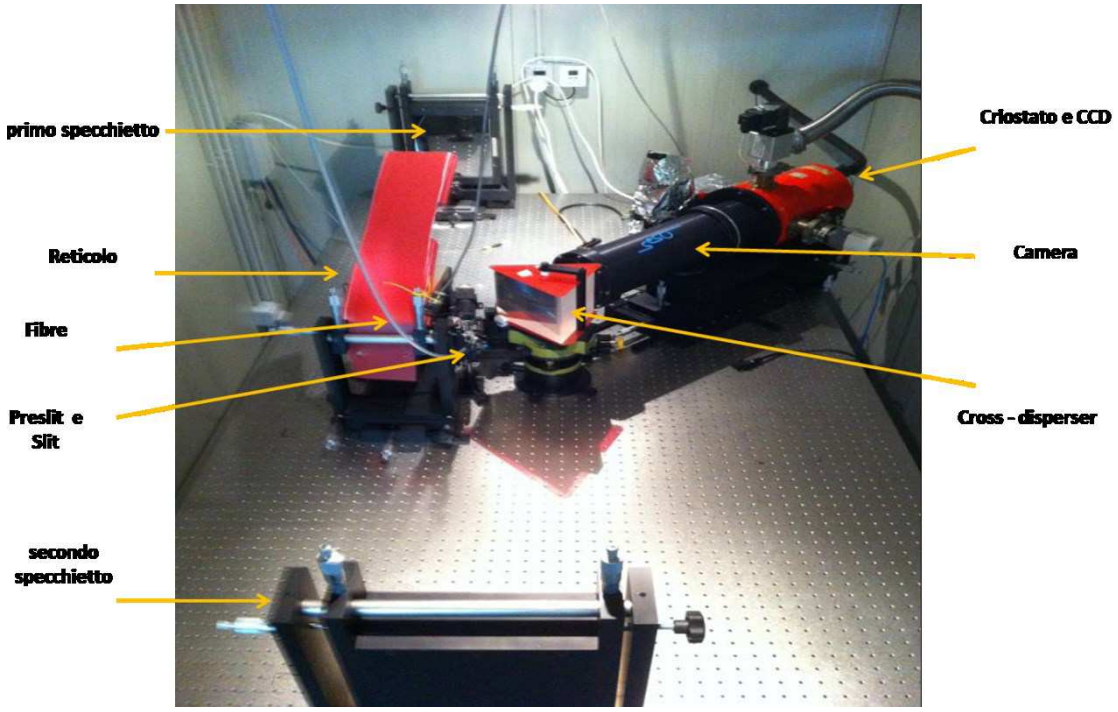

Figura 1.9: Lo spettrografo CAOS

1.2.2 L'ottica di preslit

CAOS è uno spettrografo alimentato con due fibre ottiche da 100 µm. Tali fibre sono poste l'una sull'altra ad una distanza di 1 mm, prima dell'ottica di preslit.

Per convertire la elevata apertura ottica (F/4) delle fibre in fasci F/10 vengono utilizzate una singola lente e una doppia lente (doppietto), si veda la figura 1.9. All'uscita dell'ottica di preslit le immagini delle due fibre appaiono con un diametro di 285 µm. Le fibre sono ruotate per avere una distanza proiettata, all'uscita dell'ottica di preslit, pari a 560 µm in modo da evitare la sovrapposizione degli ordini spettrali, si veda la figura 1.10.

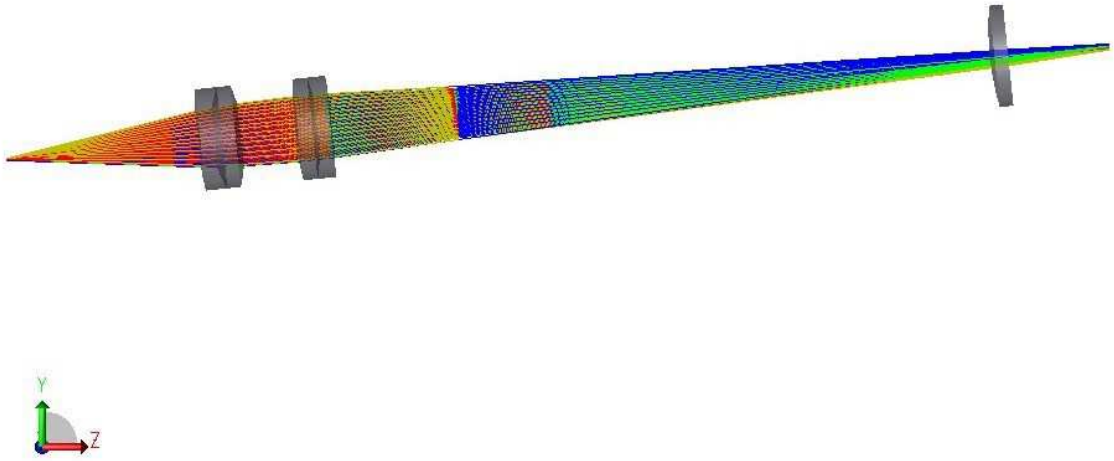

Figura 1.10: Ottica di preslit

1.2.3 Gli specchi collimatori

Uno specchio parabolico di F/2.4 realizzato mediante Astrosital, che è un materiale vetroso a basso coefficiente di espansione, con diametro di 400 mm ed una lunghezza focale di 1000 mm, è stato tagliato in due parti "fuori-asse" (off-axis) come mostrato in figura 1.10. Entrambe le porzioni sono state rivestite con argento per avere un'alta riflettività (> 97%).

La parte più grande è stata usata "fuori asse" come primo collimatore F/10, inviando il fascio verso il reticolo echelle. Lo specchio è utilizzato una seconda volta per la formazione di un piano focale intermedio ricevendo luce dispersa dal reticolo echelle.

La seconda parte è stata utilizzata per collimare la luce verso il cross-disperser. Occorre notare che, a differenza di altri spettrografi "white-pupil", non è stato adottato nessuno specchietto intermedio per riflettere la luce dal primo verso il secondo collimatore ed allineare più facilmente lo spettrografo. In figura 1.11 si può osservare che i collimatori sono stati estrusi da uno specchio parabolico di 400 mm di diametro, le misure sono date in mm e lo specchio più grande funziona come primo collimatore [3].

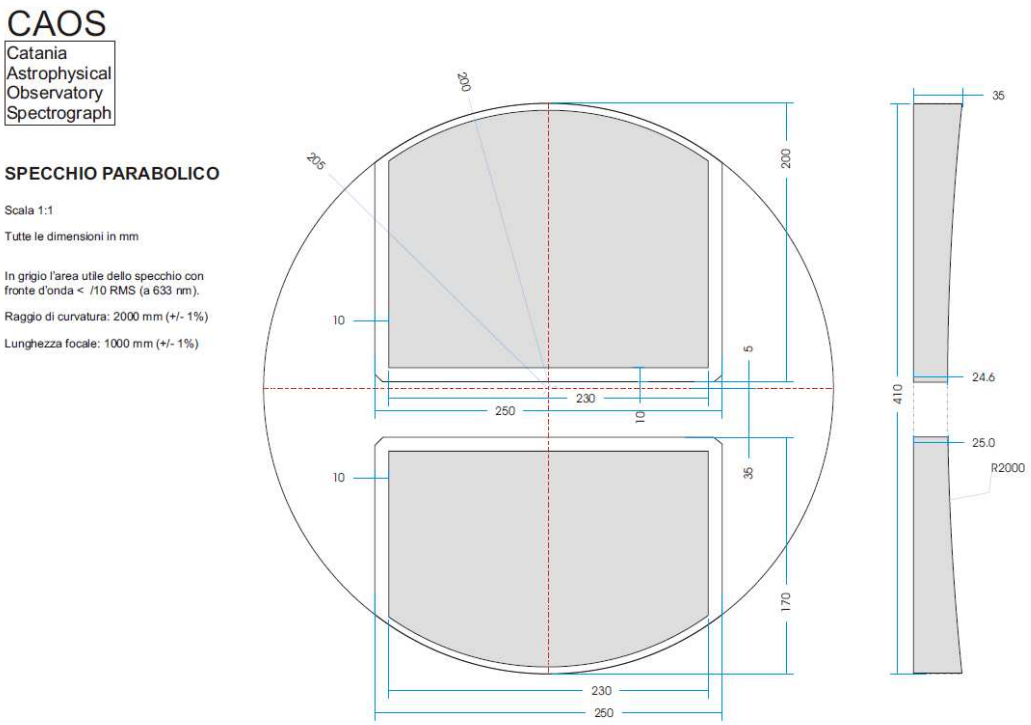

Figura 1.11: Lo specchio parabolico

1.2.4 Il reticolo

Il reticolo echelle fornito dalla SPECTRA PHYSICS Richardson Gratings presenta 41.59 linee per mm con un angolo di blaze di 76° (R4) che è l'angolo di massima efficienza del reticolo cioè l'angolo per cui il 40%-80% della luce viene riflessa nella direzione desiderata. L'area della linea è pari a 120x408 mm^2 con un rivestimento riflettente di Alluminio. Il funzionamento del reticolo si basa su una configurazione Littrow cioè gli angoli incidenti e riflessi del reticolo sono quasi coincidenti (con segno opposto) infatti $\alpha \sim -\beta$ per massimizzare il numero di fotoni in uscita dallo spettrografo cioè l'efficienza (si veda l'equazione 1.7).

Per evitare la sovrapposizione dell'immagine del reticolo con l'immagine della fibra sul piano focale, quindi per evitare che il fuoco prodotto all'uscita dell'ottica di preslit e il fuoco prodotto dal primo collimatore coincidano, il reticolo è stato ruotato di un angolo "fuori piano" (off-plane) pari a 0.9°.

1.2.5 Il cross-disperser

Un prisma avente dimensioni di 140x160x160 mm^3, realizzato dalla ditta SILO con materiale vetroso di tipo Schott SF1 e con un angolo di apice di 53.1°, è stato adottato come cross-disperser, si veda la figura 1.12. I rivestimenti A/R (antiriflesso) di cui vengono rivestiti i materiali vetrosi per minimizzare le riflessioni e le conseguenti perdite di luce, dovute al passaggio del raggio di luce all'interfaccia tra due mezzi con indici di rifrazione differenti, danno un'efficienza media prossima al 99%. Lo spettro echelle risultante copre l'intervallo spettrale di 3881-7250 Å in 56 ordini senza salti e senza sovrapposizioni per le due proiezioni della fibra, a valle dell'ottica di preslit.

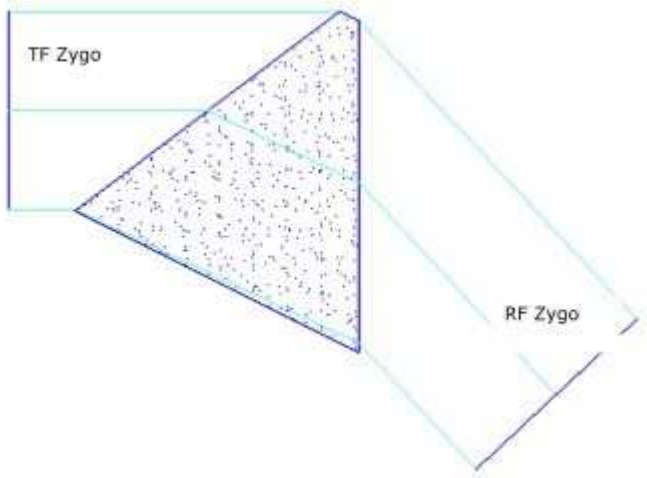

Figura 1.12: Schema di un prisma cross-disperser.

1.2.6 La camera

La camera F/2.1, rappresentata in figura 1.13, è stata prodotta dalla Sociétè Européenne de Systémes Optiques e consiste di 5 lenti montate in tre gruppi: una lente doppia (doppietto), due lenti separate, una lente piana che agisce come una finestra di Dewar, un criostato dove dentro di esso è posizionato il CCD. Come sarà illustrato nella prossima sezione, il CCD deve essere sotto vuoto e deve essere chiuso da un materiale trasparente come il vetro in modo da poterlo illuminare. La finestra di Dewar è raffigurata come l'ultima lente nella camera. In figura 1.13 si può osservare il piano focale inclinato [3].

Figura 1.13: Disegno ottico della camera F/2.1 di CAOS.

1.2.7 Il rivelatore CCD

Il sensore utilizzato è un "E2V42-40 Thin and Back Illuminated High Performance Charge Coupled Device" avente 2048x2048 pixel ciascuno dei quali con area pari a 13.5x13.5 μm^2 e una superficie d'immagine di 27.6 mm x 27.6 mm che è il risultato del prodotto tra il numero di pixel e l'area di ciascuno di essi. Esso è montato all'interno di un criostato e raffreddato ad una temperatura di -135 °C tramite LN_2 (Azoto Liquido). Il criostato è collegato attraverso una elettrovalvola al sistema di vuoto di CAOS. E' poi collegato, mediante un tubo coibentato, al serbatoio dell'azoto liquido esterno al box termostabilizzato. Sul criostato sono montati il controller CCD e un sensore di pressione. Questo sensore è stato scelto per la sua elevata efficienza quantica, data dal rapporto tra il numero di fotoni rivelati dal detector e il numero di fotoni incidenti sulla superficie, la quale presenta dei picchi anche fino a 85% a 450 nm, come si vede in figura 1.14. CAOS funziona

con un CCD E2V42-40, la cui efficienza quantica è riportata in rosso come funzione della lunghezza d'onda. Per confronto, l'efficienza quantica di un Thick CCD è riportata in blu.

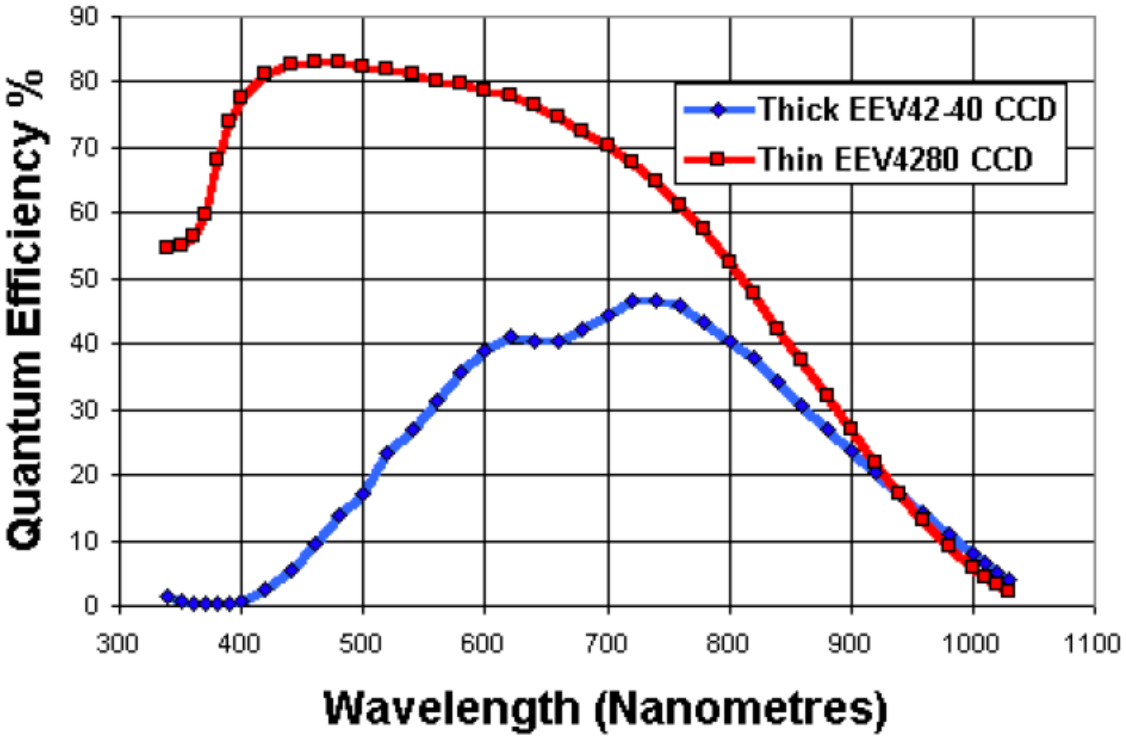

Figura 1.14: Efficienza quantica di CAOS e di un Thick CCD.

Dal grafico si evince che la migliore efficienza quantica è presente nei CCD sottili e retroilluminati. Come qualsiasi CCD sottile e retroilluminato, il sensore di CAOS è affetto dal fenomeno di fringing. All'interno degli strati sensibili del CCD rivestiti di materiale antiriflettente, utilizzato per migliorare l'efficienza, si presentano fenomeni di interferenza che dipendono dalla lunghezza d'onda. Tale fenomeno prende il nome di fringing. Come si vede in figura 1.15 ci sono delle zone più scure dove è presente una maggiore interferenza. Grazie ad un controller del CCD a 2^{19} bit, corrispondenti a 524228 livelli, non si sono verificati problemi di saturazione e non linearità durante le osservazioni.

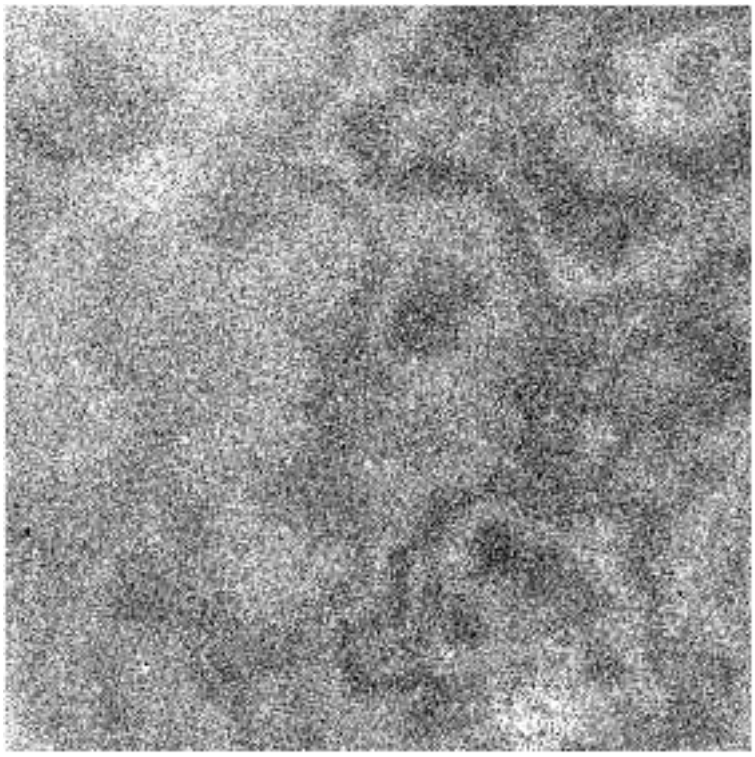

Figura 1.15: Effetto fringing

1.2.8 L'echellogramma

L'echellogramma rappresenta l'immagine della sorgente luminosa. Quello finale di CAOS dopo l'allineamento consiste di 82 ordini, si veda la figura 1.16, che copre lunghezze d'onda comprese tra 376 e 1121 nm, pertanto sono stati registrati gli ordini tra 44 e 125.

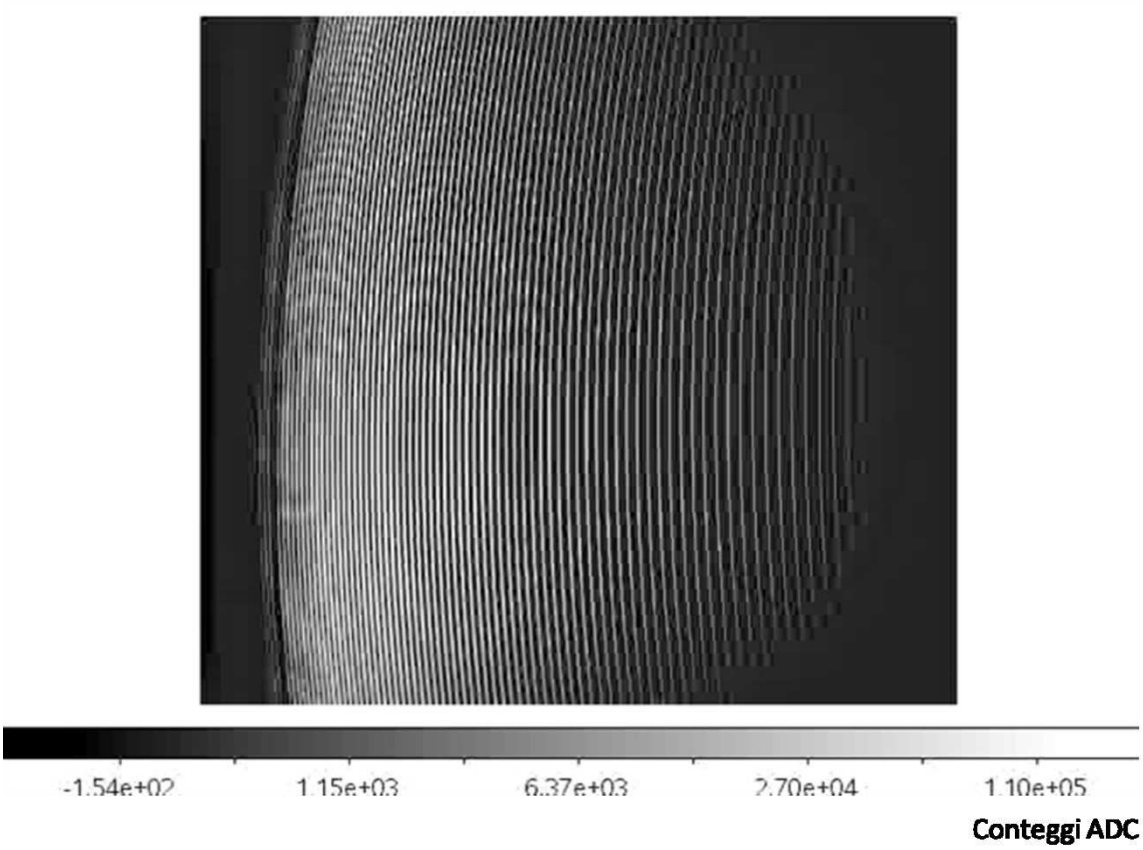

Figura 1.16: Echellogramma di CAOS

1.2.9 L'alloggiamento

Al giorno d'oggi, è possibile ottenere un ampio intervallo spettrale e un'elevata risoluzione mediante un campionamento di Nyquist su pixel che nel caso di CAOS sono di dimensioni 13.5x13.5 μm^2. CAOS, con il suo collimatore da 1 m, è montato su un banco di dimensioni 2.4x1.2 m^2. Con una tipica espansione termica lineare per metalli dell'ordine di 10 μ/K per 1 m di materiale a 20 gradi Celsius, è chiaro che anche un cambiamento di temperatura ambiente di pochi gradi può comportare uno spostamento significativo dell'echellogramma sul CCD. È stato stimato che un cambiamento di 0.001 Kelvin nello spettrografo HARPS (High Accuracy Radial velocity Planet Searcher) [7] comporta uno spostamento di λ nel pixel sul CCD che corrisponde ad una variazione apparente della velocità radiale pari a 1 m/s [7].

Per evitare le variazioni di temperatura dovute all'alternanza giorno-notte e delle stagioni, CAOS è posizionato in una stanza a temperatura controllata. La figura 1.17 mo-

stra che la temperatura viene stabilizzata con un r.m.s.(valor quadratico medio) di 0.006 gradi. Ciò è in accordo con la risoluzione spettrale di CAOS che corrisponde a 12 m/s [3].

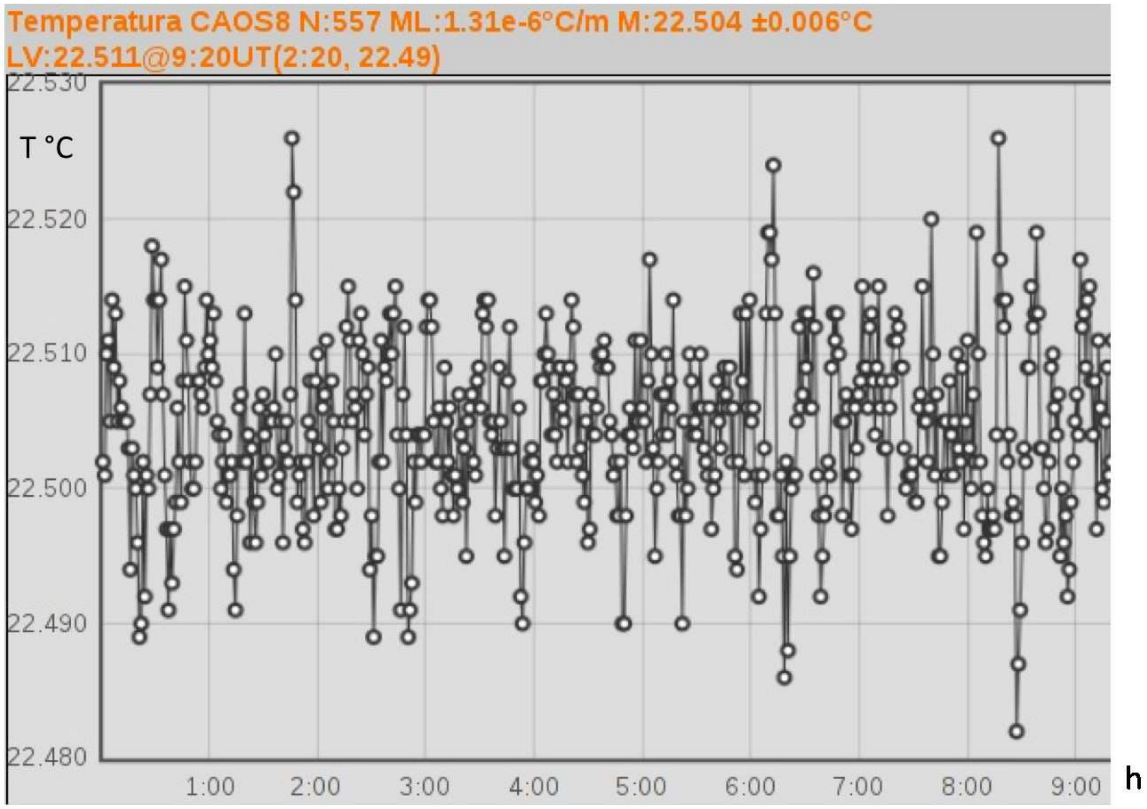

Figura 1.17: Stabilità della temperatura di CAOS in gradi Celsius al variare delle ore

1.2.10 L'efficienza (Throughput)

Si tenga presente che il parametro V rappresenta la magnitudine apparente o luminosità nel visibile in scala logaritmica della stella. Per una stella con V=0, avente la massima luminosità, si ottiene, in buone condizioni di osservabilità, 20000 fotoni al secondo; questo dato è in buon accordo con il calcolo del tempo di esposizione dello spettrografo FEROS (Fiber-fed Extended Range Optical Spectrograph) [6] che dà 19000 fotoni al secondo, con una visibilità di 2'' (arco secondi).

1.3 L'interfaccia attuale di CAOS

L'interfaccia al telescopio, si veda la figura 1.18, ha la funzione principale di convogliare la luce del telescopio verso lo spettrografo. Essa è una scatola montata sulla parte posteriore del telescopio che contiene [1]:

1. Il sistema di calibrazione.
2. Il modulo polarimetrico.
3. Il sistema di autoguida (si veda il paragrafo 1.4)
4. Il modulo porta fibre.

Infine è presente un elettronica di controllo.

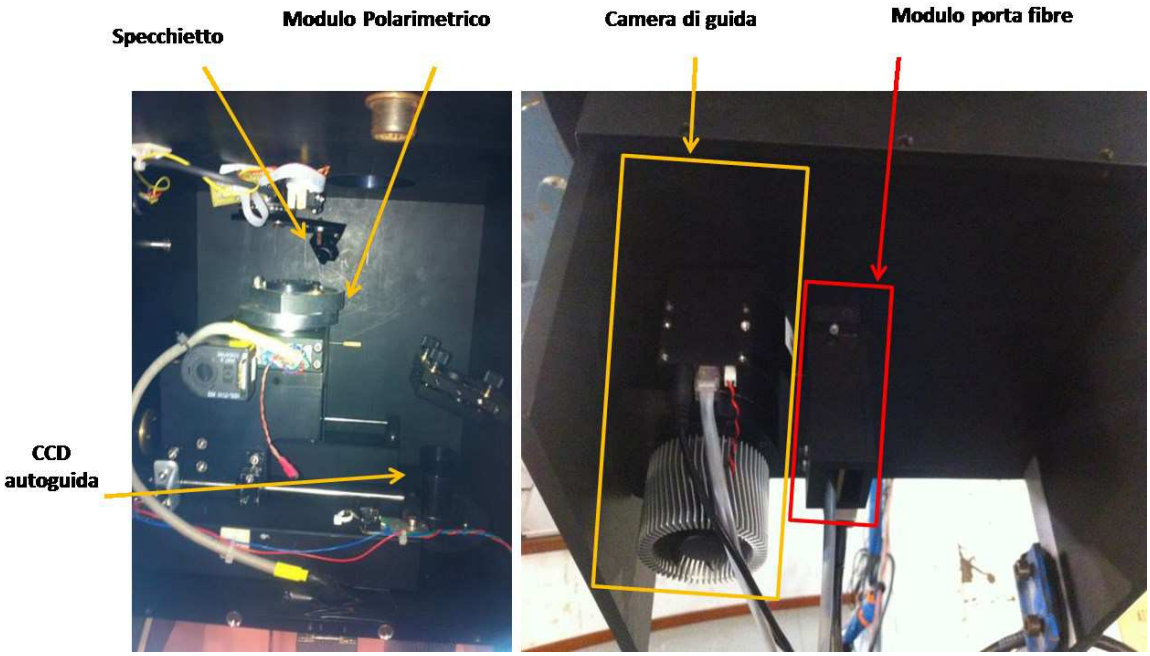

Figura 1.18: L'interfaccia tra il telescopio e CAOS.

Considerando la Figura 1.18, nella parte sinistra si ha l'interno dell'interfaccia che presenta, in alto lo specchio per le calibrazioni, al centro il modulo polarimetrico e a destra una parte del sistema di autoguida. Nella parte destra della figura, è raffigurato l'esterno dell'interfaccia con la camera di guida e il modulo porta fibre.

1.3.1 Il sistema di calibrazione

Tale sistema è costituito dal gruppo lampade e da uno specchietto per le calibrazioni. Il gruppo lampade è attualmente montato sul pilastro del telescopio. Contiene una lampada alogena per i flat ed una al torio-argon per la calibrazione in lunghezza d'onda. E' collegato all'interfaccia attraverso una fibra ottica.

1.3.2 Le fibre ottiche

CAOS è alimentato da una coppia di fibre ottiche di 100 μm di diametro l'una e distanziate da una lunghezza di 2 mm che, dall'interfaccia al telescopio, vanno verso lo spettrografo situato al piano inferiore, si veda la figura 1.19.

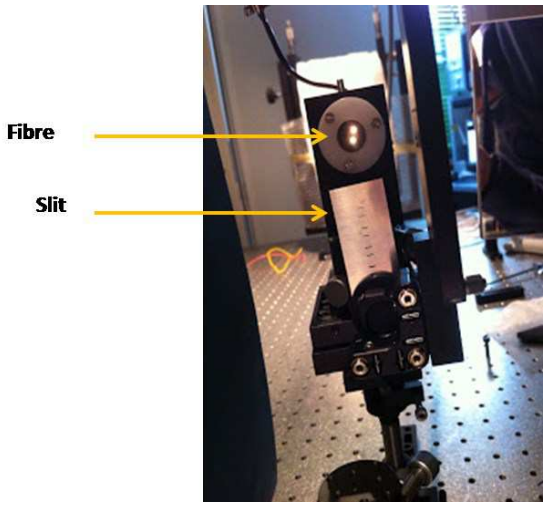

Figura 1.19: Le fibre illuminate così come si vedono all'uscita dell'ottica di preslit.

1.3.3 Il modulo polarimetrico

CAOS è dotato di un modulo polarimetrico che permette di misurare la polarizzazione lineare e circolare nell'intervallo 376-1121 nm.

Tra i vari metodi utilizzati per misurare lo stato di polarizzazione, Clarke & Grainger (1971) [4] hanno dimostrato il vantaggio di tecniche a doppio fascio rispetto a quelle a singolo fascio, così sono stati scartati i polarizzatori come il prisma di Nicol, dove uno dei fasci polarizzati linearmente non emerge.

Il metodo utilizzato per eseguire misure di spettropolarimetria si basa su due lamine di ritardo $\lambda/2$ e $\lambda/4$, che possono essere alternativamente inserite lungo il percorso ottico e

ruotate in riferimento ad un separatore di fascio per misurare, rispettivamente, la polarizzazione lineare e circolare.

Il polarizzatore è costituito da calcite ($CaCO_3$), ha dimensioni di 15x15 mm^2 ed una lunghezza totale relativa alle due piastre cementate di 32 mm, dando luogo ad una separazione del fascio di 2.5 mm nell'intervallo del visibile; entrambi i lati sono rivestiti di materiale antiriflesso per l'intervallo 350-900 nm. Come polarizzatore è stata scelta una plate di Savart perché può essere utilizzata in un fascio convergente, come nel caso di CAOS, permettendo di avere il fuoco di entrambi i fasci (ordinario e straordinario) sullo stesso piano e in asse con le fibre [3].

La plate di Savart è accoppiata in serie con due lamine.

A causa dell'ampio intervallo spettrale coperto da CAOS in modalità polarimetrica (376-1121 nm), le lamine di ritardo di basso ordine o di ordine zero non sono adatte, pertanto sono preferite le lamine di ritardo acromatiche.

Le lamine super acromatiche sono formate da più strati di materiale, ognuno dei quali presenta uno spessore molto piccolo, dell'ordine delle centinaia di μm. Ciò comporta che tali lamine possano dar luogo a fenomeni di interferenza, pertanto non sono state utilizzate.

1.3.3.1 L'allineamento del polarimetro

Per trovare gli angoli α e β che caratterizzano, rispettivamente, l'orientamento della lamina di ritardo e del polarizzatore rispetto ad uno degli assi del sistema di riferimento scelto, per lo spettrografo CAOS, è stata seguita questa procedura.

Il sistema di allineamento è costituito da un polaroid, una lamina di ritardo $\lambda/2$ e una plate di Savart. Un polaroid è un polarizzatore che, se orientato opportunamente, fa emergere un solo stato di polarizzazione. Si consideri una sorgente di luce non polarizzata. Inizialmente si inserisce il polaroid e la plate di Savart, pertanto se il polaroid è orientato correttamente, all'uscita della plate di Savart si ha un solo fascio, ad esempio quello ordinario. A questo punto tra il polaroid e la plate di Savart si inserisce una lamina $\lambda/2$ la quale ruota di 90° il piano di polarizzazione del fascio proveniente dal polaroid. Se la lamina $\lambda/2$ è inserita con un orientamento casuale, all'uscita della plate di Savart emergeranno il raggio ordinario e quello straordinario. Pertanto quando si inserisce la lamina $\lambda/2$, tale lamina

deve essere ruotata opportunamente in modo tale che il fascio ordinario scompaia ed emerga solo il fascio straordinario all'uscita della plate di Savart. In questo modo, osservando solo il fascio straordinario all'uscita del polarizzatore, si è certi che il piano di polarizzazione del fascio emergente dal polaroid è stato ruotato di 90°. Poiché dalla plate di Savart emerge un solo fascio si è certi anche che l'asse ottico del polarizzatore, il quale è dato dalla congiungente i due fasci ordinario e straordinario, e l'asse ottico della lamina $\lambda/2$ coincidono cioè gli angoli α e β sono coincidenti. Si veda la figura 1.20.

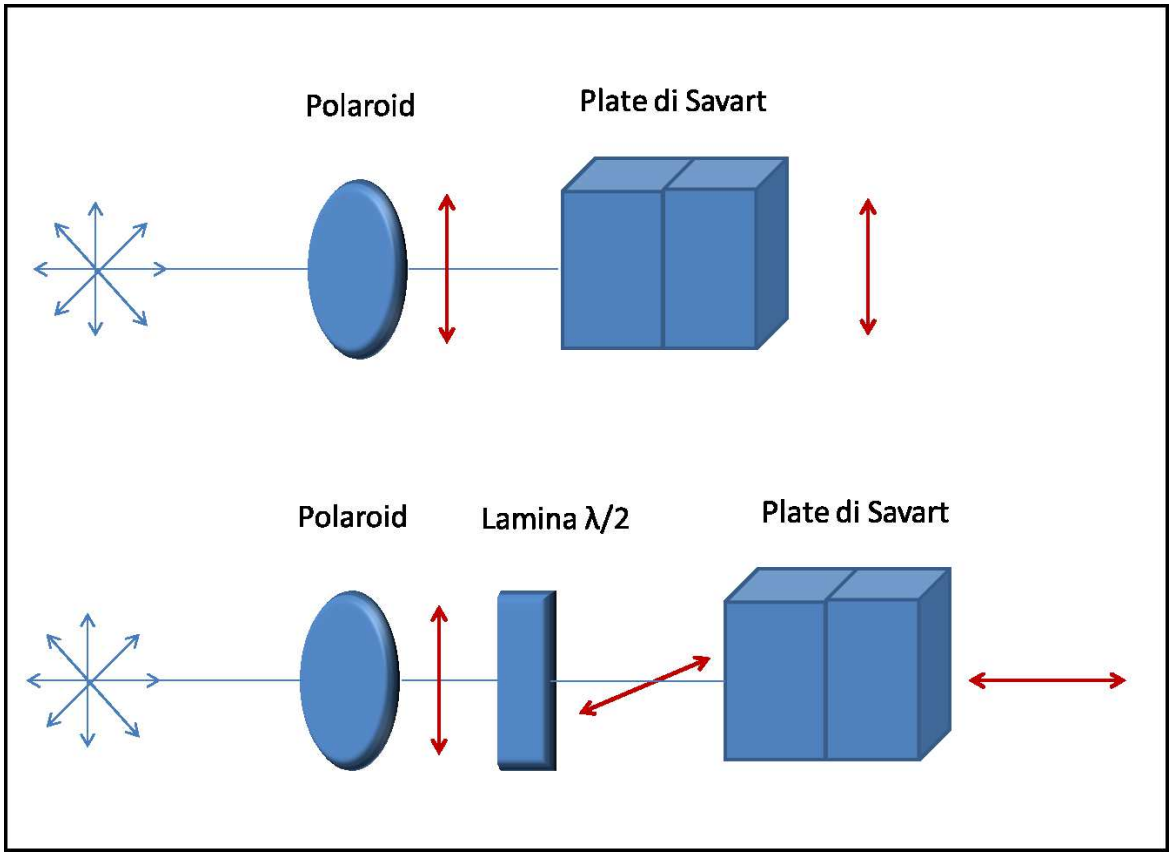

Figura 1.20: Allineamento per la polarizzazione lineare

Per analizzare la luce in polarizzazione circolare, occorre inserire la lamina $\lambda/4$ al posto della lamina $\lambda/2$. La polarizzazione circolare si misura convertendo tale stato di polarizzazione in polarizzazione lineare. Si consideri una sorgente di luce non polarizzata che attraversa il polaroid. Da tale dispositivo emerge un'onda polarizzata linearmente che attraversa la lamina $\lambda/4$. Se l'asse ottico della lamina forma un angolo di 45° rispetto alla direzione di polarizzazione dell'onda entrante dalla lamina emerge un'onda polarizzata

circolarmente in un senso, ad esempio in senso orario. Se l'asse ottico della lamina $\lambda/4$ forma un angolo di 45°+ 90°=135° rispetto alla direzione di polarizzazione dell'onda entrante, la lamina fa emergere un'onda polarizzata circolarmente nel senso opposto al caso precedente, ad esempio in senso antiorario. Un'onda polarizzata circolarmente è formata da due onde o due stati di polarizzazione lineare ortogonali e uguali in intensità, pertanto introducendo una plate di Savart dopo la lamina $\lambda/4$, all'uscita della plate di Savart emergeranno due fasci corrispondenti ai due stati di polarizzazione. Tali fasci sono l'onda ordinaria e l'onda straordinaria, separati spazialmente, ortogonalmente polarizzati e di uguale intensità. In questo caso l'allineamento si realizza inserendo la lamina $\lambda/4$ e facendola ruotare fino a quando dall'uscita della plate di Savart emergono due fasci ortogonali e di uguale intensità. In tal modo si è certi che la lamina $\lambda/4$ ha l'asse ottico formante un angolo di 45° rispetto all'asse ottico della plate di Savart che coincide con quello del polaroid. Si veda la figura 1.21.

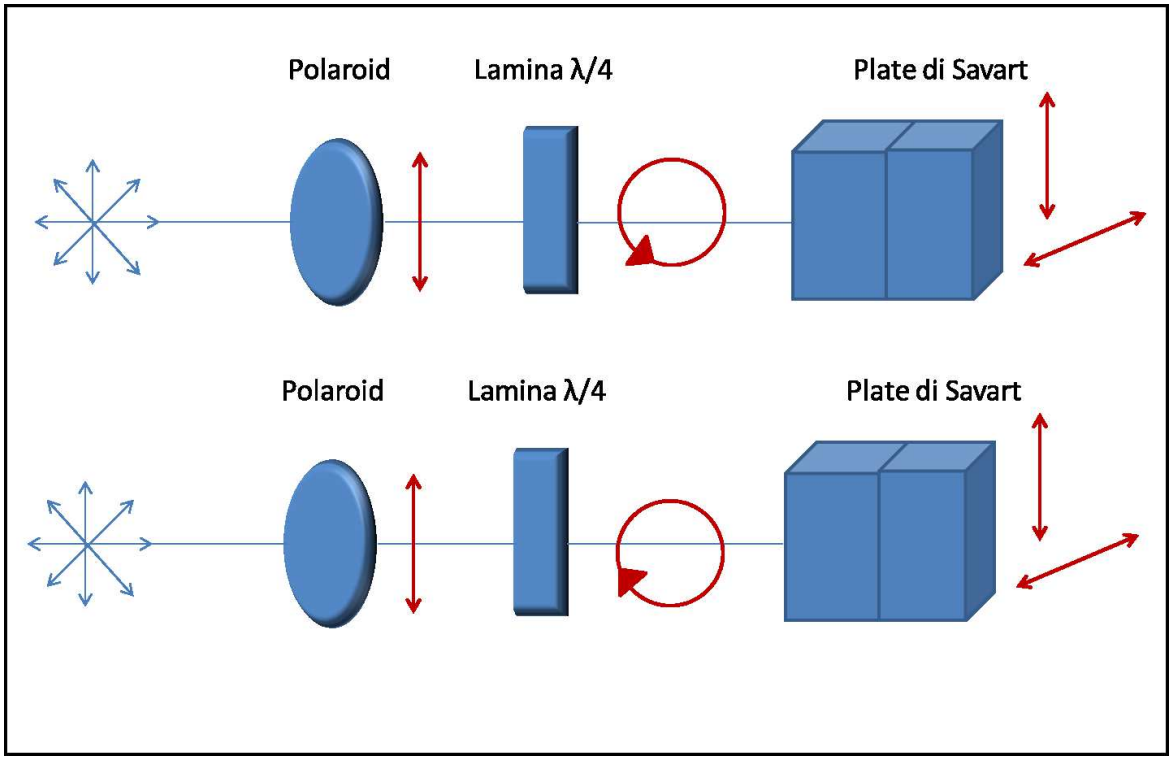

Figura 1.21: Allineamento per la polarizzazione circolare

La procedura precedentemente descritta ci ha permesso di realizzare l'allineamento del polarimetro, cioè di trovare l'angolo di riferimento o angolo di zero rispetto al quale

ruotare la lamina di ritardo per poter ricavare tutti i parametri di Stokes che ci forniscono una misura del grado di polarizzazione del fascio di luce [10].

Da un punto di vista strettamente matematico per allineare l'asse veloce della lamina di ritardo rispetto all'asse di accettazione della plate di Savart, è stato applicato il metodo di Goodric et al. [11], come è stato applicato da Leone et al. [3]

Il flusso del segnale in uscita dal polarizzatore è funzione dell'angolo α tra l'asse veloce della lamina e l'asse di accettazione del polarizzatore e dell'angolo β che l'asse di accettazione del polarizzatore forma rispetto al meridiano principale o celeste (dato dalla circonferenza che passa per la stella e per il polo Nord) e l'angolo δ che indica lo sfasamento introdotto dalla lamina (calcite) tra il raggio ordinario e straordinario:

$$\begin{aligned} S(\alpha,\beta,\delta) = 0.5\,\{\, & I + (Q\cos 2\alpha + U \sin 2\alpha)\cos\,[2(\beta-\alpha)] \\ & - (Q\sin 2\alpha - U\cos 2\alpha)\,\sin\,[2(\beta-\alpha)]\cos\delta \\ & + V \sin 2(\beta-\alpha)\sin\delta\} \end{aligned} \qquad (1.8)$$

CAOS misura i parametri di Stokes sulla base di un numero finito di valori dell'angolo α e ponendo $\beta = 0$. Dall'equazione (1.8) misuro i parametri di Stokes per vari valori dell'angolo α. Il parametro di Stokes V è misurato con una lamina di ritardo $\lambda/4$ ($\delta = \pi/2$) per $\alpha = +45°$ e $\alpha = -45°$. Mediante una lamina di ritardo $\lambda/2$ ($\delta = \pi$), il parametro di Stokes Q è misurato per $\alpha = 0°$ e per $\alpha = +45°$, mentre il parametro di Stokes U per $\alpha = +22°.5$ e $\alpha = +67°.5$. Per combinare i fasci ordinario e straordinario emergenti dal polarizzatore e misurare i parametri di Stokes si segue il metodo del rapporto introdotto da Tinbergen & Rutten [8]. Si assume che sia presente una sensibilità strumentale dipendente dal tempo $G(\lambda)$, per esempio l'efficienza che varia da pixel a pixel, insieme ad una sensibilità $F(\lambda)$ dello spettro, per esempio la variazione della trasparenza del cielo. In questo modo se si considera un regime ideale in cui l'errore dominante è quello fotonico (che segue la statistica di Poisson ed è pari a $\sqrt{n}$, dove n = numero di fotoni) il generico parametro di Stokes $P(\lambda)$, dato dalla 1.9, può essere ottenuto dallo spettro memorizzato per $\alpha = \alpha_1$ e per $\alpha = \alpha_2$, trovati in precedenza:

$$P(\lambda) = \sqrt{Q^2 + U^2 + V^2} \tag{1.9}$$

$$S_{\alpha_{1,o}}(\lambda) = 0.5\,[I(\lambda) + P(\lambda)]G_o(\lambda)F_{\alpha_1}(\lambda) \tag{1.10}$$

$$S_{\alpha_{1,e}}(\lambda) = 0.5\,[I(\lambda) - P(\lambda)]G_e(\lambda)F_{\alpha_1}(\lambda) \tag{1.11}$$

$$S_{\alpha_{2,o}}(\lambda) = 0.5\,[I(\lambda) - P(\lambda)]G_o(\lambda)F_{\alpha_2}(\lambda) \tag{1.12}$$

$$S_{\alpha_{2,e}}(\lambda) = 0.5\,[I(\lambda) + P(\lambda)]G_e(\lambda)F_{\alpha_2}(\lambda) \tag{1.13}$$

Quindi,

con
$$\frac{P(\lambda)}{I(\lambda)} = \frac{R_P(\lambda) - 1}{R_P(\lambda) + 1}$$
$$R_P^2(\lambda) = \frac{S_{\alpha_{1,o}}(\lambda)/S_{\alpha_{1,e}}(\lambda)}{S_{\alpha_{2,o}}(\lambda)/S_{\alpha_{1,e}}(\lambda)} \tag{1.14}$$

Il rapporto $P(\lambda)/I(\lambda)$, per ogni parametro di Stokes rappresenta la percentuale di polarizzazione o di radiazione polarizzata, normalizzata all'intensità della radiazione stessa.

La polarizzazione è indipendentemente dalle funzioni di sensibilità degli strumenti utilizzati, $F(\lambda)$ e $G(\lambda)$.

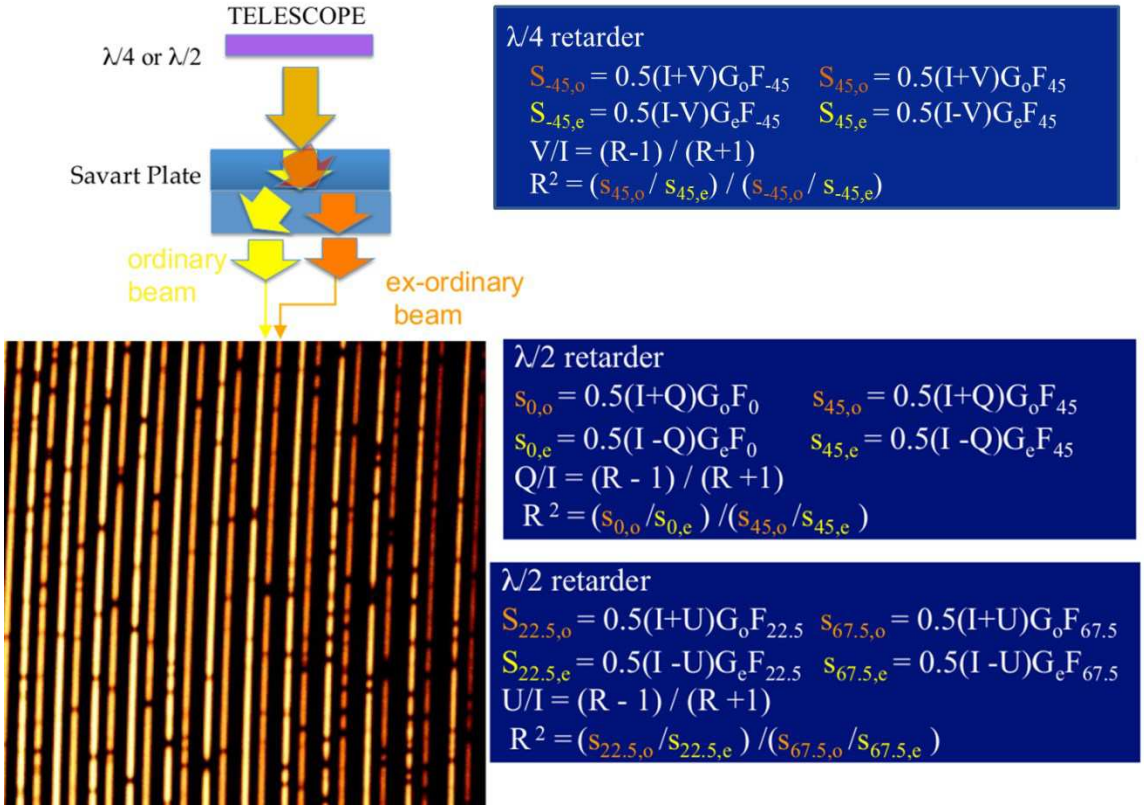

Figura 1.22: Echellogramma di CAOS alimentato da due fibre.

In figura 1.22, a sinistra, è riportato il metodo di allineamento sopra descritto e lo spettro di una stella. Sono presenti due serie di righe ognuna delle quali è dovuta alla presenza delle due fibre, per cui si ha le righe rappresentate in colore giallo corrispondenti al fascio ordinario e quelle arancioni al fascio straordinario. A destra sono raffigurati i parametri di Stokes Q,U,V, l'intensità I e il rapporto R_P (indicato con R) . Tali parametri sono stati ricavati come illustrato in precedenza

1.3.4 L'elettronica di controllo

L'elettronica di controllo è sostanzialmente concentrata sul rack nella parte posteriore del box termostabilizzato, si veda la figura 1.23. Permette il controllo completo di tutte le funzioni automatizzate dello spettrografo.

Figura 1.23: Il rack dell’elettronica di controllo di CAOS

1.4 Descrizione della nuova interfaccia al telescopio

Lo scopo della mia attività è quello di realizzare una nuova interfaccia tra il telescopio da 91 cm e lo spettropolarimetro CAOS, situati all'osservatorio M.G. Fracastoro di SLN.

La nuova interfaccia al telescopio dello spettropolarimetro CAOS nasce da due diverse esigenze:

1. Rendere più efficiente il sistema di autoguida.
2. Aumentare il grado di automazione dell'interfaccia al telescopio.

Da un punto di vista funzionale la nuova interfaccia al telescopio può essere suddivisa in tre differenti sottosistemi, come si vede in figura 1.24:

1. Sottosistema di autoguida.
2. Sottosistema di calibrazione.
3. Sottosistema polarimetrico.

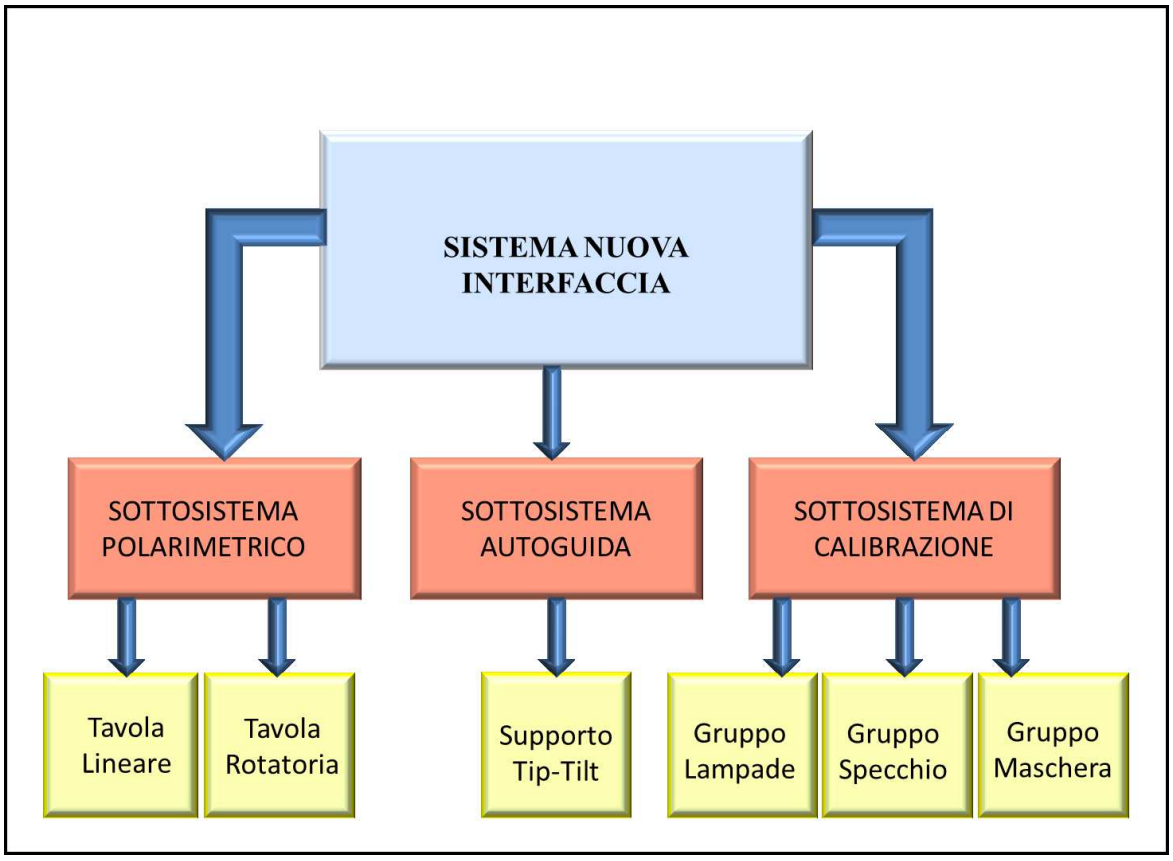

Figura 1.24: Schema a blocchi delle movimentazioni della nuova interfaccia

1.4.1 Il sottosistema di autoguida

Il vecchio sistema di autoguida è costituito da una camera CCD della Finger Lake che monta un CCD Kodak KAF0260 da 512x512 pixel quadrati con dimensioni 20 μm^2 ed un PC Raspberry per la gestione della camera stessa. Un beam sampler montato davanti il porta fibre nella interfaccia al telescopio sottrae il 4% della luce proveniente dal telescopio ed attraverso un sistema ottico che mantiene la scala del telescopio, crea un immagine dell'oggetto da osservare sulla camera CCD. La procedura di guida calcola la differenza tra una posizione di riferimento (quella che sul piano focale del telescopio corrisponde alla posizione della fibra selezionata) e quella corrente dell'oggetto da osservare. L'offset calcolato viene poi inviato al telescopio. A causa delle caratteristiche meccaniche del telescopio ed in funzione delle condizioni osservative (brillanza dell'oggetto, posizione in cielo, seeing) la percentuale di luce persa a causa del sistema di autoguida corrente varia tra il 25% ed il 40% della luce incidente [1].

Al fine di migliorare le prestazioni dell'autoguida si è deciso di cambiare filosofia e di non utilizzare più i movimenti del telescopio per effettuare la guida ma uno specchietto di tip-tilt opportunamente inserito nel cammino ottico. I vantaggi dello specchietto sono la velocità e la precisione della risposta superiori sicuramente in prestazioni a quelli possibili con il telescopio. L'uso del telescopio viene limitato ai casi in cui un eventuale movimento di deriva, dovuto ad un cattivo bilanciamento del telescopio in alcune posizioni, superi la corsa disponibile dello specchietto per le correzioni. Il feedback allo specchietto viene garantito da una nuova camera CCD sempre della Finger Lake con un CCD KAF-0402-1 da 768x512 pixel da 9 μm^2 mostrata in figura 1.25 [1].

Figura 1.25: La nuova camera CCD Finger Lake del sottosistema di autoguida

1.4.2 Il sottosistema di calibrazione

Il vecchio sistema di calibrazione di CAOS consiste dello specchietto per le calibrazioni e del gruppo lampade dello spettrografo FRESCO (Fiber-optic Reosc Echelle Catania Observatory).

Il nuovo sistema di calibrazione è invece così costituito [1]:

1. Tavola lineare per le calibrazioni al posto di quella corrente.
2. Tavola lineare per l'inserimento/disinserimento di una maschera per illuminare la singola fibra.
3. Nuovo gruppo lampade.

Per quanto riguarda lo specchietto, il relativo motore viene sostituito da una tavola lineare perché quello in uso ha il controller che presenta dei malfunzionamenti di comunicazione.

Viene aggiunta una maschera, pilotata mediante una tavola lineare, per consentire alla luce proveniente dal gruppo lampade per la calibrazione, di andare su una singola fibra senza che l'altra venga illuminata, oppure per poter effettuare osservazioni solo con una fibra.

Il gruppo lampade, non oggetto di questa tesi, viene rinnovato per usura e invecchiamento dei suoi componenti, infatti esso è composto da parti mobili che con il passare del tempo danno errori sulle movimentazioni ed ha un'interfaccia basata su una porta paralle-

la, la quale diventa sempre più difficile da trovare nei moderni PC; inoltre non si può effettuare un feedback per sapere se le lampade sono accese. Per tali motivi si è scelto un nuovo gruppo lampade che utilizza una comunicazione Ethernet (protocollo di rete dello standard TCP/IP) ed è basato sul microcontrollore Arduino[2] mediante il quale si può realizzare il feedback, quindi è possibile conoscere se le lampade sono accese.

1.4.3 Il sottosistema polarimetrico

Il vecchio modulo polarimetrico di CAOS è costituito da una tavola rotatoria per i movimenti delle due lamine a mezz'onda e a quarto d'onda. L'inserimento del modulo polarimetrico è attualmente manuale.

Il nuovo sistema di movimentazione dell'interfaccia è invece così costituito [1]:

1. Tavola lineare per l'inserimento/disinserimento del modulo polarimetrico.
2. Tavola rotatoria per i movimenti delle lamine (stesso meccanismo dell'interfaccia corrente).

Nella nuova interfaccia il sottosistema polarimetrico viene gestito in modo automatico. Tale modulo viene inserito se si desidera analizzare sia la componente parallela che quella perpendicolare del fascio di luce proveniente dal telescopio.

[2] https://www.arduino.cc

1.5 Passaggio dalla vecchia alla nuova interfaccia

Nell'attuale interfaccia, si veda la figura 1.26, la luce proveniente dal telescopio, dopo aver eventualmente attraversato il polarimetro, a seconda del tipo di analisi che si vuole effettuare, focalizza sulle microlenti all'entrata delle fibre che alimentano lo spettrografo CAOS. Per permettere ad una porzione del fascio di luce di entrare nell'autoguida, poco prima del fuoco (circa 1 cm) è posizionato un beam sampler[3] [12], una piccola lastra di vetro che presenta una superficie non trattata con rivestimento (coating*)* antiriflesso e inclinata a circa 20 gradi rispetto alla fascio proveniente dal telescopio. Sfruttando la riflessione di Fresnel, circa il 4% della radiazione viene quindi riflessa dal sampler (e non trasmesso verso le fibre) e tramite una lente, nuovamente focalizzata sul sensore della camera del sistema di autoguida. Quindi, al variare della posizione dell'immagine della sorgente sul piano del fuoco del telescopio, dovuta ad esempio alle imprecisioni meccaniche del sistema di tracciamento (tracking) del telescopio, l'immagine sulla camera di guida presenterà uno spostamento (offset). E' possibile allora ricavare in funzione del tempo lo spostamento sulla camera di guida dell'immagine della sorgente e conseguentemente lo spostamento dell'immagine sul piano delle fibre di CAOS; tale spostamento, riducendo la radiazione che va ad alimentare lo spettrografo, è ovviamente negativo per le prestazioni dello strumento. Per eliminarlo (o comunque ridurlo), dalla posizione della sorgente sulla camera di guida vengono ricavate delle correzioni cioè gli spostamenti angolari da inviare al sistema di controllo del telescopio, in modo da riportare la sorgente "in fibra" così da correggere la posizione dell'immagine della sorgente sul piano delle fibre. Questa operazione viene ripetuta, durante l'utilizzo dello strumento, alcune volte al minuto.

La nuova interfaccia si differenzia sostanzialmente dalla vecchia per la presenza di uno specchietto tip-tilt che intercetta il fascio prima di dirigerlo verso il beam sampler e verso le fibre. Tale specchietto, controllato da un sistema PLC Beckhoff, ha il compito di correggere la posizione dell'immagine sulle fibre, svolgendo il ruolo che precedentemente svolgeva il telescopio; in questa maniera non è necessario movimentare di continuo il telescopio che per sua natura (a causa delle dimensioni, dell'usura, e dell'età) tende ad essere meno preciso rispetto ad un sistema di dimensioni molto ridotte (alcuni centimetri di dia-

[3] https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=913

metro) come lo specchietto tip-tilt. I nuovi collegamenti sono evidenziati in rosso in figura 1.27.

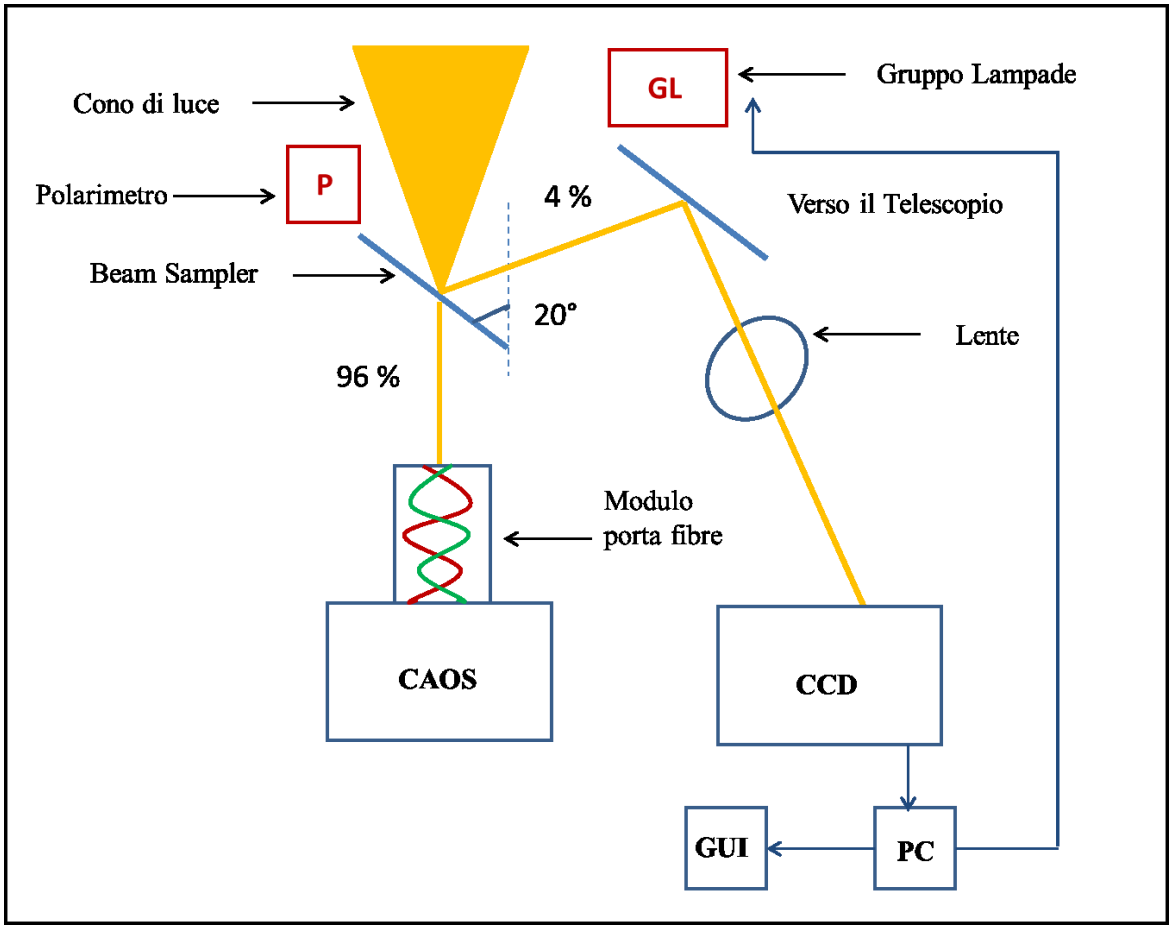

Figura 1.26: La vecchia interfaccia tra il telescopio e CAOS

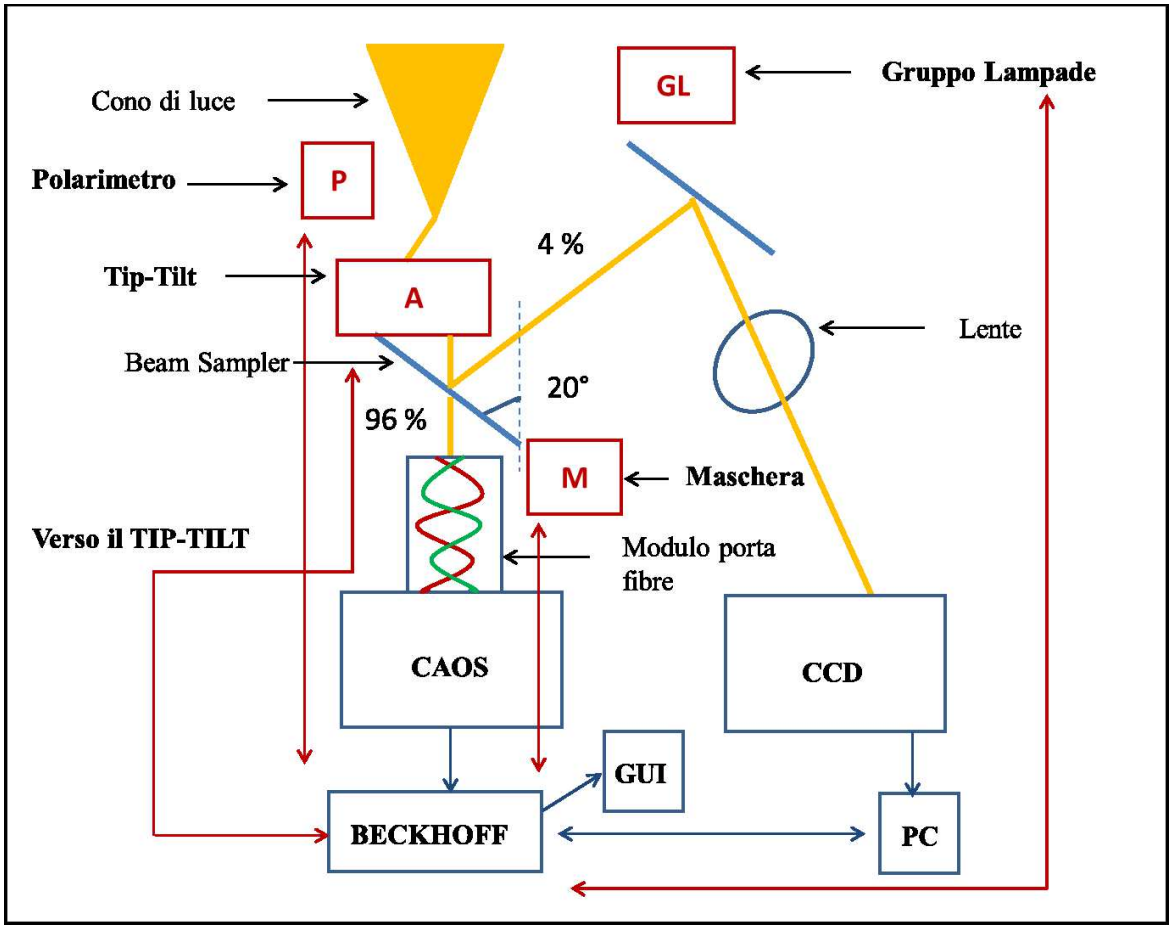

Figura 1.27: La nuova interfaccia tra il Telescopio e CAOS

CAPITOLO 2

Funzioni delle movimentazioni della nuova interfaccia

2 Introduzione

In questo capitolo saranno descritte inizialmente le funzionalità delle movimentazioni dei vari sottosistemi che compongono la nuova interfaccia CAOS con il telescopio, poi si passerà alla descrizione dettagliata delle caratteristiche dell'hardware utilizzato per ogni sottosistema ed infine saranno dati alcuni cenni sull'ambiente di sviluppo Beckhoff TwinCAT 3, utilizzato in questo lavoro per lo sviluppo del software necessario per la configurazione delle movimentazioni e per la creazione di un' interfaccia grafica.

2.1 Descrizione delle funzionalità dell'interfaccia

2.1.1 Il sottosistema di autoguida

In questo sottosistema esiste una sola movimentazione: lo specchietto di Tip-Tilt. Nella soluzione utilizzata lo specchietto viene montato in una montatura ottica motorizzata tipo gimbal della Zaber, modello T-OMG (figura 2.5). Il meccanismo ha due assi tip e tilt, inoltre è dotato di controller e comunica con l'esterno attraverso una porta seriale.

Le funzionalità previste per questo meccanismo sono [1]:

1. Inizializzazione dei due assi. Ricerca dei fine corsa e posizionamento in una posizione predefinita per ciascuno degli assi.
2. Movimenti relativi continui dei due assi rispetto alla posizione di inizializzazione.
3. Movimento di Homing (ricerca dei fine corsa).

Nello specifico, la posizione iniziale predefinita dei due assi, ovvero quella in cui i due assi si posizionano dopo l'inizializzazione rispetto al fine corsa, coincide con la posi-

zione di fine corsa. A parte la posizione iniziale non sono previste per questo meccanismo altre posizioni predefinite.

2.1.2 Il sottosistema di calibrazione

Il sottosistema di calibrazione è costituito da due movimentazioni e dal gruppo lampade.

Le due movimentazioni sono [1]:

1. Tavola lineare per inserimento/disinserimento dello specchio di calibrazione.
2. Tavola lineare per inserimento/disinserimento della maschera per l'illuminazione o schermatura della singola fibra.

L'integrazione del gruppo lampade nel sottosistema di calibrazione del sistema di controllo, non è oggetto di questa tesi.

2.1.2.1 Tavola specchio di calibrazione

La tavola per l'inserimento/disinserimento dello specchio di calibrazione è un tavola lineare della Zaber modello LSA25A (figura 2.8) su cui sarà montato uno specchietto da 1/2 pollice. Il meccanismo ha una corsa totale di 25 mm e un sensore di fine corsa, non è dotato di controller e comunica con l'esterno attraverso un connettore 15 pin.

Le funzionalità previste per questo meccanismo sono [1]:

1. Inizializzazione dell'asse. Ricerca del fine corsa e posizionamento in una posizione predefinita, detta di default.
2. Movimento di inserimento dello specchietto. Dalla posizione di default lo specchietto deve muoversi in maniera da inserirsi nel cammino ottico della luce proveniente dal telescopio, intercettare il fascio in uscita dal gruppo lampade ed inviarlo alle fibre.
3. Movimento di disinserimento dello specchietto. Con questo movimento lo specchietto montato sulla tavola verrà rimosso dal cammino ottico della luce proveniente dal telescopio e ritornerà nella posizione di default.
4. Movimento di Homing (ricerca del fine corsa).

2.1.2.2 Tavola maschera

La tavola per l'inserimento/disinserimento della maschera per l'illuminazione o la schermatura della singola fibra, è un tavola lineare della Zaber modello LSA25A (figura 2.8). Sul meccanismo sarà montata una maschera di dimensione e forma tali da permettere, quando inserita, di illuminare una fibra alla volta. Il meccanismo ha una corsa totale di 25 mm e un fine corsa, non è dotato di controller e comunica con l'esterno attraverso un connettore 15 pin.

Le funzionalità previste per questo meccanismo sono [1]:

1. Inizializzazione dell'asse. Ricerca dei fine corsa e posizionamento in una posizione predefinita, detta di default.
2. Movimento di inserimento nella posizione denominata "fibra A". Dalla posizione di default o dalla posizione "fibra B", la maschera deve muoversi in maniera da inserirsi nel cammino ottico della luce proveniente dal telescopio o dal gruppo lampade ed oscurare il fascio ottico così da permettere l'illuminazione della sola fibra A.
3. Movimento di inserimento nella posizione denominata "fibra B". Dalla posizione di default o dalla posizione "fibra A", la maschera deve muoversi in maniera da inserirsi nel cammino ottico della luce proveniente dal telescopio o dal gruppo lampade ed oscurare il fascio ottico così da permettere l'illuminazione della sola fibra B.
4. Movimento di disinserimento della maschera. Con questo movimento la maschera verrà rimossa dal cammino ottico (posizione "fibra A" o "fibra B") e ritornerà alla posizione di default.
5. Movimento di Homing (ricerca del fine corsa)

2.1.3 Il sottosistema polarimetrico

Il sottosistema polarimetrico è costituito da due movimentazioni [1]:

1. Tavola lineare per l'inserimento del modulo polarimetrico.
2. Tavola rotatoria per il movimento della lamina a mezz'onda e della lamina a quarto d'onda.

Per quanto concerne la nuova interfaccia, sono state implementate le procedure relative solo alla prima movimentazione. Le procedure per la seconda movimentazione non sono state implementate in quanto il polarimetro deve essere smontato in modo da poter caratterizzare il motore della tavola rotatoria cioè pilotarlo mediante il sistema Beckhoff. Tuttavia ciò non è possibile poiché il polarimetro è attualmente in uso, pertanto la tavola rotatoria non è disponibile.

2.1.3.1 Tavola per l'inserimento del modulo polarimetrico

La tavola per l'inserimento/disinserimento del modulo polarimetrico è una tavola lineare della Standa modello 8MT150 (figura 2.2). Sul meccanismo sarà montato il modulo polarimetrico. Il meccanismo ha una corsa totale di 150 mm e due fine corsa, non è dotato di controller e comunica con l'esterno attraverso un connettore 9 pin.

Le funzioni previste per questo meccanismo sono [1]:

1. Inizializzazione dell'asse. Ricerca dei fine corsa e posizionamento in una posizione predefinita, detta di default.
2. Movimento di inserimento del modulo. Dalla posizione di default, il modulo polarimetrico deve muoversi in maniera da inserirsi nel cammino ottico della luce proveniente dal telescopio o dal gruppo lampade.
3. Movimento di disinserimento del modulo. Con questo movimento il modulo polarimetrico verrà rimosso dal cammino ottico della luce e ritornerà nella posizione di default.
4. Movimento di Homing (ricerca del fine corsa).

2.2 Il sistema di controllo

Il sistema di controllo utilizzato in questo lavoro è stato selezionato in maniera da soddisfare i requisiti funzionali e di sicurezza richiesti dal progetto. Sono state valutate varie caratteristiche come le prestazioni in tempo reale e la compattezza del sistema, l'elevata affidabilità e la ridotta necessità di manutenzione. In particolare il sistema di controllo è basato su un PC embedded e su una serie di componenti modulari periferici prodotti dalla Beckhoff. Tale sistema di controllo si avvale di standard industriali, software e hardware, largamente diffusi, si veda [35] [36].

Il software di controllo a basso livello è stato realizzato utilizzando l'applicazione TwinCAT 3 di Beckhoff che è un'ambiente per la programmazione di PLC (Programmable Logical Controller) molto diffuso.

2.2.1 Componenti hardware – PLC Beckhoff

Il PC Beckhoff utilizzato è il modello CX-5020-0120, si veda la figura 2.1. Monta un processore Intel® Atom™ Z530 con frequenza di clock di 1.6 GHz, presenta una RAM di 512 MB e, come memoria di massa è presente una scheda Compact Flash di 8 GB, accessibile dall'esterno, che opera come hard-disk. La Compact Flash viene utilizzata all'avvio del dispositivo e come supporto di memorizzazione.

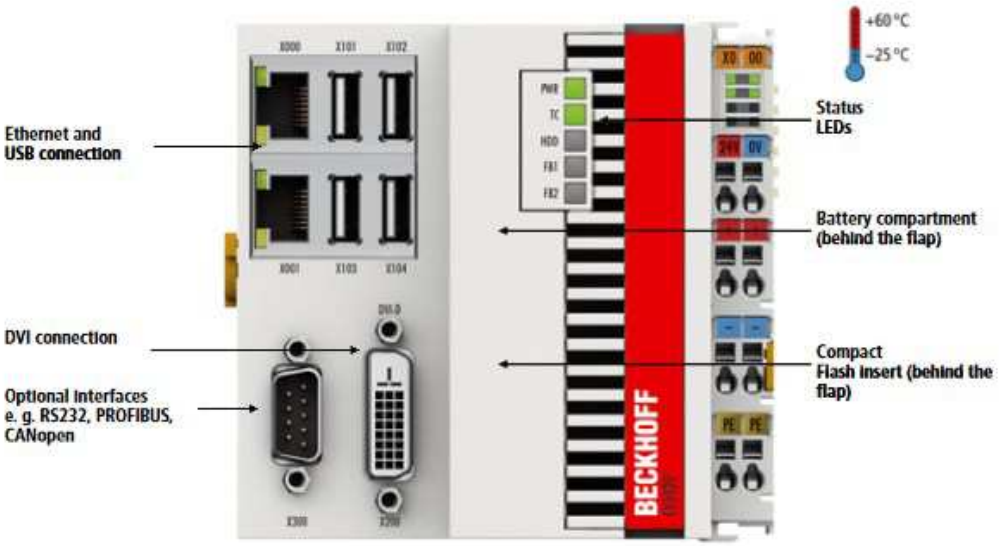

Figura 2.1: Dispositivo Beckhoff e connessioni

Inoltre sono presenti 2 porte Gigabit Ethernet RJ-45, 4 porte USB 2.0 e una porta DVI-D la quale consente il collegamento ad un monitor DVI. Inoltre è presente un gruppo UPS integrato. Il range di temperatura di lavoro varia tra -25 °C e +60 °C. Il sistema operativo presente è Windows CE.

2.2.2 Terminali EtherCAT

Il PC embedded CX5020 è stato sviluppato per supportare e rendere ottimale l'integrazione con EtherCAT (Ethernet Control Automation Technology).

I terminali o moduli che sono stati scelti, sono i seguenti:

- EL2008: Terminale di uscite digitali a 8 canali con tensione d'ingresso a 24 Volt DC e corrente di 0.5 A DC e con un consumo di corrente E-bus pari a 110 mA [21].
- EL1018: Terminale di ingressi digitali a 8 canali con tensione d'ingresso a 24 Volt DC e corrente di 3 mA DC e con un consumo di corrente nominale sui power contacts pari a 2 mA [20].
- EL7041: Tale terminale è ideato per il controllo motori stepper (passo-passo) a due fasi, di medie prestazioni e che richiedono un'alimentazione nominale tra +8 Volt e +50 Volt [25].
- EL3002: Terminale di ingressi analogici a 2 canali (input), alimentato via E-bus. Processa segnali nell'intervallo compreso tra -10 Volt e +10 Volt [22].
- EL7031: Tale terminale è ideato per il controllo motori stepper (passo-passo) a due fasi, di medie prestazioni e che richiedono un'alimentazione nominale inferiore a +8 Volt e fino a +24 Volt [24].
- EL6002: Tale terminale è ideato per gestire interfacce seriali [23].

2.2.2.1 Il protocollo EtherCAT

EtherCAT[1] è un sistema di bus di campo sviluppato dalla ditta Beckhoff Automation. Il protocollo è standardizzato nella norma IEC 61158 ed è adatto per soddisfare i requisiti hardware e software di applicazioni in tempo reale (real-time) nell'ambito dell'automazione.

[1] https://www.ethercat.org/en/technology.html

Si considerino le sue principali caratteristiche.

EtherCAT è un bus di campo industriale basato su Ethernet: i telegrammi EtherCAT hanno la stessa struttura di un telegramma Ethernet TCP/IP (46÷1500 byte). La sua baud-rate è pari a 100 Mbit/s, più precisamente il protocollo può gestire:

- 256 canali I/O digitali in 11 μs
- 200 canali I/O analogici in 30 μs
- 100 servo-assi in 100 μs

EtherCAT è una rete basata sul modello master-slave: l'unico nodo in grado di inviare autonomamente telegrammi EtherCAT è il Master, e tutti i nodi Slave usano i frame inviati dal Master per scambiare i loro dati.

Un pacchetto EtherCAT si può propagare su 3 mezzi fisici differenti :

- 100BASE-TX : cavo Ethernet standard con connettore RJ45 (massima distanza tra due nodi 100 m)
- 100BASE-FX : fibre ottiche (massima distanza tra due nodi 2 km con fibra multimodale e 20 km con fibra monomodale)
- LVDS[2] : contatti metallici sui terminali di I/O (massima lunghezza 10 m)

Con EtherCAT i pacchetti Ethernet o trame non sono più ricevuti, interpretati e copiati come dati di processo in ogni nodo. Il dispositivo EtherCAT Slave legge i dati indirizzati ad esso mentre il pacchetto attraversa il nodo stesso, processando i dati "al volo". In modo analogo i dati in ingresso sono inseriti mentre il pacchetto attraversa il nodo, pertanto a prescindere dalla topologia di rete, il frame EtherCAT viene processato dagli slave seguendo uno schema ad **anello logico**:

- il frame viene inviato dal Master EtherCAT con dati in uscita aggiornati per tutti gli Slave
- ogni Slave riceve il frame dal nodo precedente
- lo Slave processa il frame: legge i suoi dati in uscita dal frame e scrive i suoi dati in ingresso sul frame
- ogni Slave inoltra il frame al nodo successivo
- dopo aver attraversato tutti i nodi slave presenti in rete, il frame ritorna al Master EtherCAT con dati in ingresso aggiornati

[2] LVDS: Low Voltage Differential Signalling secondo ANSI/TIA/EIA-644 utilizzato anche in IEEE 802.3ae (10 Gigabit Ethernet)

- Il frame EtherCAT viene processato da ogni slave tramite un ASIC specifico detto EtherCAT Slave Controller (ESC) disponibili anche con tecnologia FPGA.

Si è scelto di utilizzare tale protocollo per il suo largo uso nell' ambito della ricerca astrofisica, in particolare per il progetto ASTRI (si veda [35] [36]), per l'elevata affidabilità nelle applicazioni real-time, per l'elevata capacità di sincronizzazione, per le altissime prestazioni in termini di velocità e larghezza di banda, per l'elevata flessibilità di cablaggio (supporta cavi differenti) e non ultimo, per la possibilità di riutilizzare delle competenze relative ad un protocollo che, essendo affermato anche in ambito industriale, viene continuamente aggiornato e migliorato.

2.3 Descrizione delle caratteristiche hardware dei sottosistemi

2.3.1 Il sottosistema polarimetrico

Il sistema in uso per lo sviluppo del software rappresenta una versione da laboratorio rispetto al sistema che sarà implementato come interfaccia al telescopio, infatti si utilizza una Workstation (PC) da laboratorio mentre il sistema Beckhoff (costituito dal PC embedded e dai moduli) e il motore sono quelli che verranno utilizzati nell'implementazione della nuova interfaccia.

Per sviluppare il nostro sistema, sono stati utilizzati i seguenti dispositivi:

- Una Workstation: computer remoto collegato tramite Ethernet (TCP/IP) al PC embedded CX5020-0120. Sulla Workstation è stato installato il software di configurazione TwinCAT 3 ed è presente l'ambiente di sviluppo Eclipse per Java.
- Un PC Beckhoff CX5020-0120: controllore PC-based (si veda il paragrafo 2.2) sul quale viene eseguita l'applicazione di controllo.
- Un modulo Beckhoff EL7041.
- Un motore "Translation Stage 8MT175-150 – numero FL42STH33-0404B" prodotto dalla Standa.
- Un cavo schermato a 9 poli, con connettore DB-9 F.

2.3.1.1 Caratteristiche del motore Translation Stage 8MT175-150

Figura 2.2: Il motore Translation Stage 8MT175-150 – Standa

Il motore “Translation Stage 8MT175-150 – numero FL42STH33-0404B è un motore stepper a due fasi il quale presenta le caratteristiche deducibili dallo schema illustrato in figura 2.3 [32].

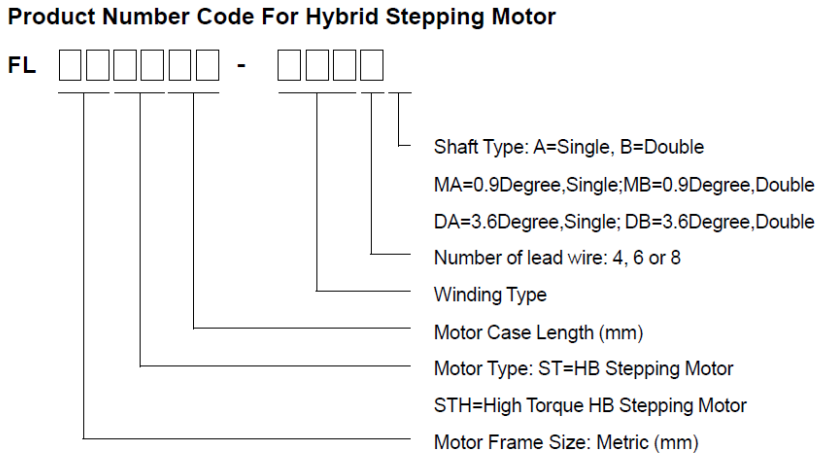

Figura 2.3: Codice del motore Translation Stage 8MT175-150 – Standa

In particolare per il motore utilizzato sono presenti i seguenti dati di targa:

- Motor frame size = 42 mm
- Motor Type = STH
- Motor Case Length = 33 mm
- Winding type = 040
- Number of lead wire = 4
- Shaft type = B

Le specifiche meccaniche ed elettriche sono riassunte nella tabelle 2.1 e 2.2

Grandezze Meccaniche	**Valore**
Travel Range – intervallo di lavoro	150 mm
Lead screw pitch – passo della vite di piombo	0.5 mm
Resolution	2.5 µm
Load capacity:	
Horizontal	8 Kg
Vertical	3 Kg
Weight	1.38 Kg

Tabella 2.1: Specifiche meccaniche del motore Translation Stage 8MT175-150 – Standa

Grandezze Elettriche	Valore
Motor Type	"BERGER" stepper motor RDM 253/50(B)
Step per revolution	200
Step angle	1.8°
Positional accuracy	0.09°
Rated Voltage	12 V
AMPS/Phase (Ampere/Fase)	240 mA
Resistance per winding	36 Ohm
Holding torque	12 Ncm
Limit switch	SW1/SW2 - Limit/Home switches
Corrent/phase	0.4 A
Resistence/phase	30 ohm
Inductance/ phase	37 mH

Tabella 2.2: Specifiche elettriche del motore Translation Stage 8MT175-150 – Standa

2.3.1.2 Il modulo Beckhoff EL7041

Il modulo che viene utilizzato per il controllo del motore sopra descritto è il EL7041 che è collegato all'unità centrale CX5020 mediante bus EtherCAT.

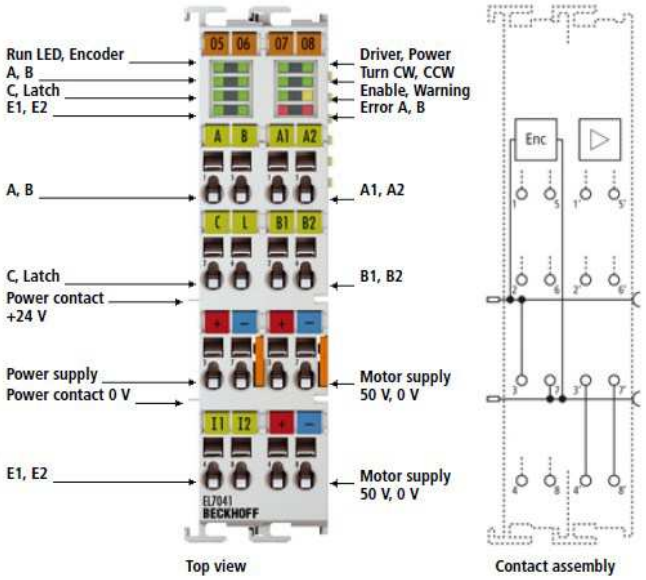

Figura 2.4: Il terminale EL7041-1000

Tale terminale, si veda la figura 2.4, è ideato per motori stepper a due fasi di medie prestazioni. Si alimenta con tensione d'ingresso fino a 50 Volt DC via E-bus ed eroga una corrente pari 5 A DC. Gli stadi di uscita PWM coprono una vasta gamma di tensioni e correnti. Una risoluzione pari a 64 microstep assicura un funzionamento del motore particolarmente silenzioso e preciso. In tale modulo sono presenti due ingressi digitali per collegare gli switches di fine corsa o di homing e la possibilità di collegare un encoder incrementale.

Il modulo si compone di due sotto-moduli: nel primo (a sinistra nella figura 2.4) sono presenti i contatti che vengono utilizzati per realizzare i collegamenti con un encoder esterno e con gli switches di fine corsa (homing) di un motore stepper, mentre nel secondo (a destra) i contatti vengono utilizzati per realizzare i collegamenti con il motore stepper. In questo lavoro di tesi, in base alle specifiche relative alla precisione, non è stato utilizzato nessun encoder.
La scelta di tale terminale è dovuta principalmente al basso assorbimento di corrente pari a 50 mA nei power contacts e 140 mA nell'E-bus e per il pilotaggio di motori stepper che richiedono correnti di uscita di 5 A e tensioni di alimentazioni superiori a 8 V [25].

2.3.2 Il sottosistema di autoguida

I dispositivi utilizzati per riprodurre il sistema in laboratorio sono i seguenti:

- Workstation: computer remoto collegato tramite Ethernet (comunicazione TCP/IP) al PC embedded CX5020-0120. Sulla Workstation viene installato il software di configurazione TwinCAT 3 ed è presente l'ambiente di sviluppo Eclipse per Java.
- PC Beckhoff CX5020-0120: controllore PC-based sul quale viene eseguita l'applicazione di controllo.
- Il modulo Beckhoff EL6002.
- Il motore T-OMG: Motorized Two-Axis Optic Mount, prodotto dalla Zaber.

2.3.2.1 Caratteristiche del motore T-OMG – Zaber

Figura 2.5: Il motore T-OMG – Zaber

La sigla T-OMG sta per "Motorized Two-Axis Optic Mount" ovvero Motore a due assi con supporto (montaggio) per un elemento ottico: l'asse azimutale e l'asse di elevazione. Infatti, il T-OMG è un sistema che presenta due attuatori stepper corrispondenti a due assi ad alta risoluzione di movimentazione, di due controller o driver e può essere con-

trollato da computer. È un'unità autonoma che richiede solo un alimentatore da 15 V. Due controller integrati, uno per ogni attuatore (motore), consentono una manipolazione facile e indipendente dall'asse di rotazione oltre ad una connessione diretta sulla porta RS-232 o USB di qualsiasi PC. Più unità possono essere collegate tra di loro su una singola porta EIA RS-232 o USB di un qualsiasi PC, inoltre è possibile collegare tra di loro più dispositivi della serie T della ZABER formando una catena di dispositivi che ha il vantaggio di poter condividere la potenza fornita da un singolo alimentatore. L'utilizzo di cavi con connettori Mini-Din a 6 pin consentono il collegamento diretto tra le unità nelle immediate vicinanze. Per distanze più lunghe, possono essere utilizzate prolunghe di cavi standard.

Si considerino in dettaglio le varie parti che compongono il motore in esame (si veda la figura 2.6) [29]:

1) una manopola fornisce un controllo manuale molto fine a velocità variabile in entrambe le direzioni;
2) un'apertura di 24 mm compatibile con le dimensioni del supporto ottico di diametro pari a 25 mm su una staffa di montaggio a due assi (two-axis gimbal mount) con un range di lavoro pari a +/- 7 gradi;
3) i cuscinetti Sapphire garantiscono la ripetizione del movimento e il posizionamento per il montaggio/supporto ottico;
4) porta seriale (con un adattatore T-DSUB 9) o USB (con un cavo T-USB DC);
5) possibilità di collegare in daisy chain più dispositivi tra loro senza cavi aggiuntivi;
6) un attacco filettato di tipo M6 con l'asse di rotazione, presente sulla superficie inferiore, per permettere il montaggio nel nostro sistema ottico;
7) due motori stepper con controller per ciascun motore.

Figura 2.6: Il motore T-OMG – Zaber in dettaglio

Le caratteristiche generali del dispositivo sono elencate nella tabella 2.3.

Specification	**Value**	**Alternate Unit**
Built-in Controller	Yes	
Range	+/- 7 degrees	
Communication Interface	RS-232	
Communication Protocol	Zaber Binary	
Aperture Diameter	24 mm	0.945 "
Maximum Current Draw	700 mA	
Power Supply	12-16 VDC VDC	
Power Plug	2.1 mm center positive	
Motor Steps Per Rev	200	
Motor Type	Stepper (2 phase)	
Inductance	1.5 mH/phase	
Default Resolution	1/64 of a step	
Data Cable Connection	Minidin 6	
Motor Frame Size	NEMA 08	
Mechanical Drive System	Precision lead screw	
Limit or Home Sensing	Magnetic hall sensor	
Manual Control	Yes, Potentiometer	
Axes of Motion	2	
LED Indicators	Yes, Bi-colour	
Mounting Interface	M6 threaded hole for optic post	
Optic Mounting Interface	1" or 25 mm optics	
Operating Temperature Range	0 to 50 degrees C	
RoHS Compliant	Yes	
CE Compliant	Yes	
Weight	0.33 kg	

Tabella 2.3: Specifiche del dispositivo T-OMG – Zaber

Asse azimutale

L'attuatore azimutale muove il supporto ottico da un lato all'alto o a destra e a sinistra attorno all'asse verticale – movimento "tip". Le sue caratteristiche sono riportate nella tabella 2.4.

Specification	Value	Alternate Unit
Microstep Size (Default Resolution)	0.000115378 degrees	2.014 urad
Accuracy (unidirectional)	0.055 degrees	0.959750 mrad
Repeatability	< 0.007 degrees	< 0.122 mrad
Backlash	< 0.005 degrees	< 0.087 mrad
Maximum Speed	11 deg/s	1.8 rpm
Minimum Speed	0.00054 deg/s	9.425 urad/s
Speed Resolution	0.00054 deg/s	
Encoder Type	None	
Communication Interface	RS-232	
Communication Protocol	Zaber Binary	

Tabella 2.4: Specifiche dell'attuatore azimutale

Asse di elevazione

L'attuatore di elevazione inclina il supporto ottico su e giù attorno ad un asse orizzontale – movimento "tilt". Le specifiche sono riportate nella tabella 2.5:

Specification	Value	Alternate Unit
Microstep Size (Default Resolution)	0.000057689 degrees	1.007 urad
Accuracy (unidirectional)	0.0275 degrees	0.479875 mrad
Repeatability	< 0.004 degrees	< 0.070 mrad
Backlash	< 0.0025 degrees	< 0.044 mrad
Maximum Speed	7 deg/s	1.2 rpm
Minimum Speed	0.00027 deg/s	4.712 urad/s
Speed Resolution	0.00027 deg/s	
Encoder Type	None	
Communication Interface	RS-232	
Communication Protocol	Zaber Binary	

Tabella 2.5: Specifiche dell'attuatore di elevazione

2.3.2.2 Il modulo Beckhoff EL6002

Il modulo che viene utilizzato per il controllo del motore sopra descritto è il EL6002, collegato all'unità centrale CX5020 mediante bus EtherCAT.

Tale terminale, si veda la figura 2.7, è ideato per fornire un'interfaccia seriale al motore. Esso è alimentato via E-bus, consuma una corrente di 170 mA e consente il collega-

mento di 2 dispositivi con interfacce RS-232, mediante connettori D-sub a 9 pin, pertanto i dispositivi collegati ai terminali EtherCAT EL6002 comunicano con il dispositivo di automazione (PLC) tramite il connettore seriale. Il canale di comunicazione funziona in modalità full duplex con rate di trasmissione che va da 300 baud fino a 115.2 kbaud. I data buffer sono di 864 bytes in ricezione e di 128 bytes in trasmissione. Le interfacce RS-232 garantiscono elevata immunità alle interferenze con i segnali isolati elettricamente [23].

La sua scelta è dovuta al fatto che è l'unico terminale per la realizzazione di una comunicazione seriale, inoltre si è scelto tale modulo per motivi di ridondanza e in prospettiva di una futura espansione del progetto in quanto presenta due porte seriali.

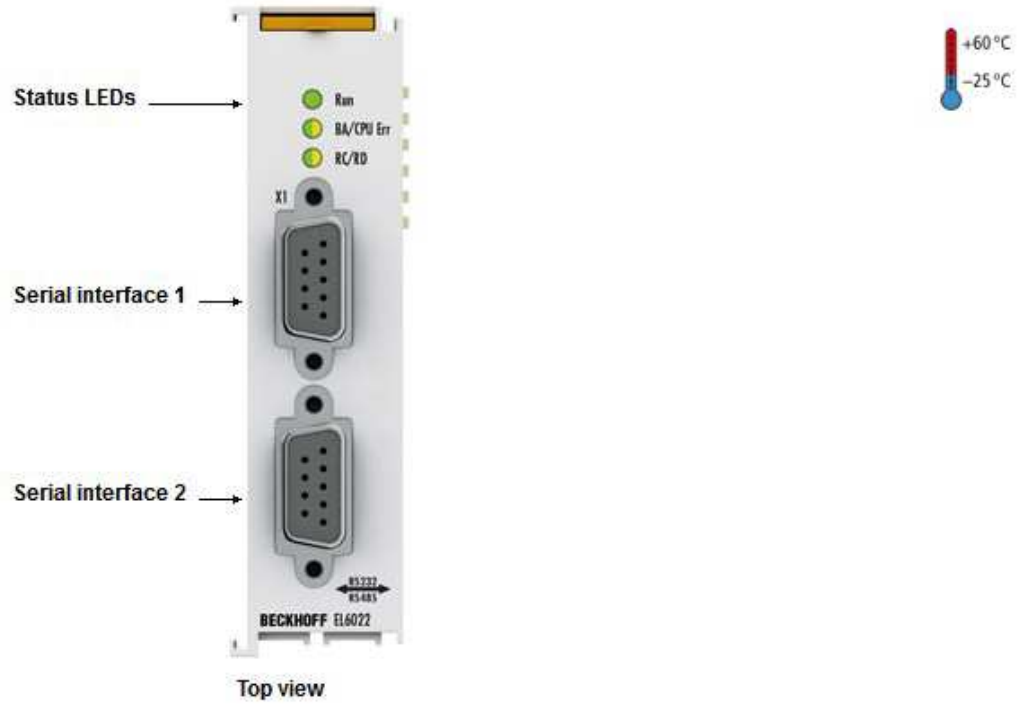

Figura 2.7: Il terminale EL6002

2.3.3 Il sottosistema di calibrazione

I dispositivi utilizzati per riprodurre il sistema in laboratorio sono i seguenti:

- Una Workstation: computer remoto collegato tramite Ethernet (comunicazione basata sui protocolli TCP/IP) al PC embedded CX5020-0120. Sulla Workstation vie-

ne installato il software di configurazione TwinCAT 3 ed è presente l'ambiente di sviluppo Eclipse per Java.

- Un PC Beckhoff CX5020-0120 : controllore PC-based sul quale viene eseguita l'applicazione di controllo.
- Due moduli Beckhoff EL7031
- Due motori "LSA25A-T4-MC04 " prodotti dalla Zaber.
- Due cavi schermati a 15 poli, con connettore Dsub-15 F.

2.3.3.1 Caratteristiche del motore LSA25A – Zaber

Figura 2.8: Il motore LSA25A-T4-MC04 – Zaber

Il motore "LSA25A-T4-MC04 – Zaber" è un motore stepper a due fasi e presenta le caratteristiche deducibili dallo schema illustrato in figura 2.9 [31].

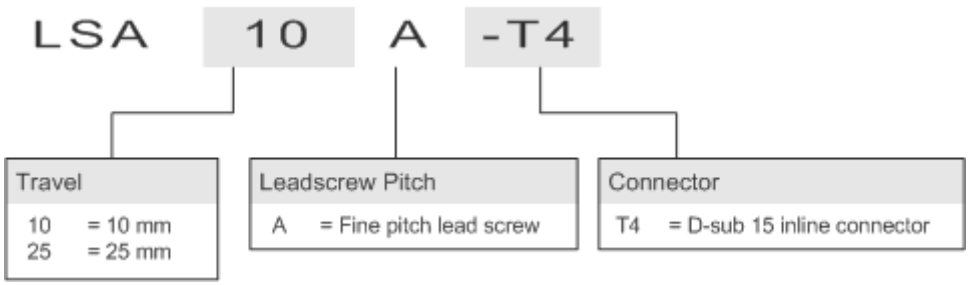

Figura 2.9: Codice del motore LSA25A-T4-MC04 – Zaber

Nel caso del motore utilizzato la corsa totale è pari a 25 mm.

Le specifiche meccaniche ed elettriche sono riportate nella tabella 2.6:

Specification	**Value**	**Alternate Unit**
Microstep Size (Default Res.)	0.0238125 µm	
Built-in Controller	No	
Recommended Controller	X-MCB1 (24 V)	
Travel Range	25 mm	0.984 "
Accuracy (unidirectional)	24 µm	0.000945 "
Repeatability	< 1 µm	< 0.000039 "
Backlash	< 5 µm	< 0.000197 "
Maximum Speed	10 mm/s	0.394 "/s
Minimum Speed	0.0000145 mm/s	0.000001 "/s
Speed Resolution	0.0000145 mm/s	0.000001 "/s
Encoder Type	None	
Peak Trust	35 N	7.8 lb
Maximum Continuous Thrust	25 N	5.6 lb
Maximum Centered Load	30 N	6.7 lb
Maximum Cantilever Load	125 N-cm	177.0 oz-in
Guide Type	Ball bearing	
Vertical Runout	< 4 µm	< 0.000157 "
Horizontal Runout	< 13 µm	< 0.000512 "
Pitch	0.14 degrees	2.443 mrad
Roll	0.05 degrees	0.873 mrad
Yaw	0.12 degrees	2.094 mrad
Linear Motion Per Motor Rev	0.3048 mm	0.012 "
Motor Steps Per Rev	200	
Motor Type	Stepper (2 phase)	
Motor Rated Current	240 mA/phase	
Motor Winding Resistance	20.4 ohms/phase	
Inductance	5 mH/phase	
Motor Rated Power	2.35 Watts	
Motor Connection	D-sub 15	
Motor Frame Size	NEMA 08	
Mechanical Drive System	Precision lead screw	
Limit or Home Sensing	Magnetic hall sensor	
Axes of Motion	1	
Mounting Interface	M2, M3 threaded holes M4 threaded centre hole	
Compatible Products	AB106, T-LSM	
Operating Temperature Range	0 to 50 degrees C	
RoHS Compliant	Yes	
CE Compliant	Yes	
Weight	0.132 kg	

Tabella 2.6: Specifiche del motore LSA25A-T4-MC04 – Zaber

2.3.3.2 Il modulo Beckhoff EL7031

Il modulo che viene utilizzato per il controllo del motore sopra descritto è il EL7031, collegato all'unità centrale CX5020 mediante bus EtherCAT.

Tale terminale, si veda la figura 2.9, è ideato per il controllo di motori stepper a due fasi di medie prestazioni. Si alimenta con tensione d'ingresso fino a 24 Volt DC via power contacts e via E-bus ed eroga una corrente di 1.5 A DC. Gli stadi di uscita PWM per le due bobine del motore sono situati nel terminale EtherCAT insieme ai due ingressi per gli interruttori di fine corsa o di homing. Una risoluzione pari a 64 microstep assicura un funzionamento del motore particolarmente silenzioso e preciso.

La scelta di tale terminale è dovuta principalmente al basso assorbimento di corrente pari a 30 mA nei power contacts e 120 mA nell'E-bus (EtherCAT-bus) e per il pilotaggio di motori stepper che richiedono basse correnti di uscita pari a 1.5 A, e tensioni di alimentazione inferiori a +8 V [24].

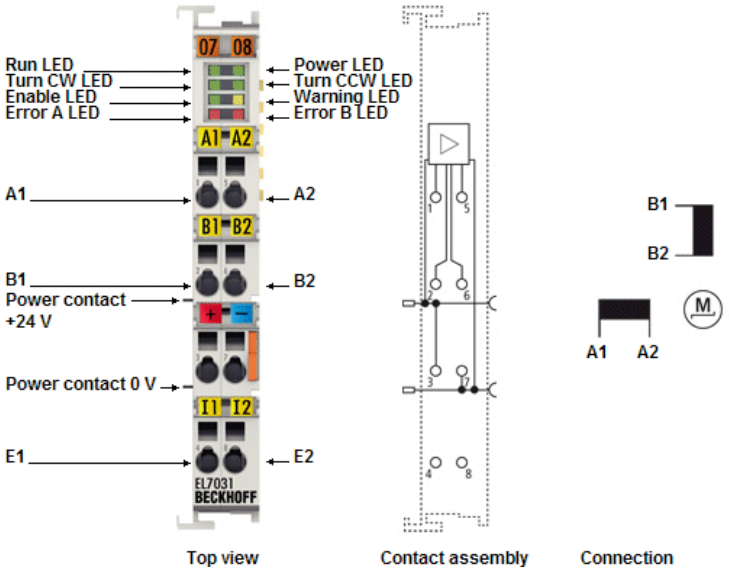

Figura 2.10: Il terminale EL7031

2.4 Componenti software – L'ambiente TwinCAT 3

Il sistema TwinCAT (The Windows Control and Automation Technology) è un pacchetto software per l'automazione real-time su piattaforma Windows con funzioni PLC e Motion, integrato nella suite di Visual Studio della Microsoft.

TwinCAT 3 è un software di controllo basato su PC di nuova generazione, sviluppato dalla società Beckhoff. Con TwinCAT 3 sono possibili vari tipi di applicazioni di controllo e l'utente può accedere a diversi linguaggi di programmazione per realizzare l'applicazione desiderata. Oltre al linguaggio di programmazione IEC 61131-3 Structured Text (ST) per una programmazione tradizionale sul PLC, l'utente può anche programmare con i linguaggi di alto livello C e C ++ [16] [17].

La filosofia TwinCAT 3 è di rendere disponibile un'innovativa architettura software, basata su un software di controllo modulare, in modo da ridurre lo sforzo ingegneristico che è necessario per controllare le macchine più complesse. Per questo motivo le singole funzioni, sistemi o gruppi di macchine sono considerate come moduli, i quali potrebbero essere utilizzati per incapsulare la funzionalità di questi oggetti, aumentando il riutilizzo, l'estensione e la manutenzione del codice di controllo. Come mostrato in figura 2.11, questa filosofia è stata implementata sia nella sezione dell'ingegneria che nella fase di esecuzione, fornendo due differenti ambienti chiamati eXtended Automation Engineering (XAE) e eXtended Automation Runtime (XAR) [15].

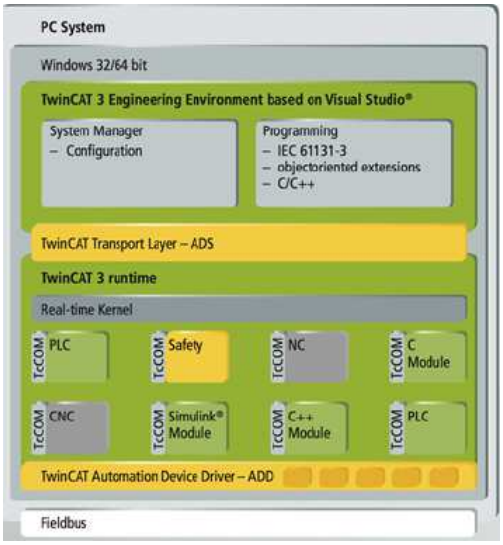

Figura 2.11: Architettura TwinCAT 3 per gli ambienti eXtended Automation Engineering (XAE) e eXtended Automation Runtime (XAR)

2.4.1 eXtended Automation Engineering (XAE)

Al fine di semplificare il software ingegneristico, gli strumenti di ingegneria TwinCAT 3 XAE sono integrati come estensione dell'ambiente di sviluppo software Microsoft Visual Studio. L'integrazione in Microsoft Visual Studio consente di programmare oggetti di automazione (moduli) con l'ausilio della 3ª edizione del linguaggio IEC 61131-3 e dei linguaggi C o C++. Il TwinCAT 3 mette a disposizione diversi moduli, schematizzati in figura 2.10, ognuno dei quali fornisce strumenti specifici.

Per la nuova interfaccia al telescopio 91 cm, i moduli utilizzati sono stati il PLC (controllore logico programmabile) e il Controllo Numerico (NC).

Il PLC TwinCAT è un modulo che offre una serie di funzioni PLC-open per il controllo del movimento e si basa sui linguaggi standard di IEC 61131-3 [19]:

- Instruction List (IL)
- Ladder Diagram (LD)
- Function Block Diagram (FBD)
- Sequential Function Chart (SFC)
- Structured Text (ST)

Esso fornisce un ambiente di sviluppo che consente di avere delle dimensioni del codice del programma molto ridotte, un editor facile ed un potente compilatore. Il PLC è il modulo principale attraverso il quale sono state implementate tutte le funzionalità del software di controllo.

Se nel TwinCAT è installato il modulo PLC, si vedrà *Configurazione PLC* nella visualizzazione struttura del System Manager (sezione Solution Explorer). In tale voce possono essere aggiunti i progetti PLC, presenti nella directory POUs (Program Organization Units), le interfacce grafiche presenti nella directory VISUs e le impostazioni del progetto (numero di sistemi di run-time, numero porta, tempi di ciclo, ecc.). In pratica tale sezione ci consente di inviare determinati comandi o impostare i parametri dal PLC verso il modulo (il terminale Beckhoff connesso al motore) quindi una via di comunicazione dal PLC al modulo (nel Twincat si usa il gergo "From PLC to NC").

Il Controllo Numerico (NC) è un modulo programmabile nel quale un processo è controllato da numeri, lettere e simboli. Qui è possibile creare gli assi richiesti e impostare i parametri per ciascun asse, infatti il TwinCAT NC PTP (Point to Point) include il soft-

ware per il posizionamento degli assi (impostazione dei parametri, controllo della posizione), un software integrato PLC con interfaccia NC che ci consente di impostare i movimenti on-line e una connessione di I/O per gli assi attraverso vari bus di campo. TwinCAT NC PTP sostituisce i moduli convenzionali per il posizionamento e il controller NC. I controller che vengono simulati dal PC, scambiano ciclicamente i dati con gli azionamenti e i sistemi di misura tramite i bus di campo. Questo è il modulo TwinCAT utilizzato per il controllo del movimento degli assi azimut e di elevazione per il sistema Zaber T-OMG, per i motori Standa e Zaber LSA trattati nel capitolo 3. In pratica la sezione *Configurazione NC* ci consente di inviare determinati comandi o impostare i parametri dal modulo verso il PLC (l'unità centrale CX5020) quindi una via di comunicazione dal modulo al PLC (nel Twincat si usa il gergo "From NC to PLC").

Per le potenzialità del TwinCAT 3, gli oggetti (moduli) generati possono scambiare dati l'uno con l'altro indipendentemente dal tipo di linguaggio con cui sono stati scritti. In questo modo è necessario un solo software per configurare, parametrizzare, programmare i dispositivi di automazione e rivelarne eventuali malfunzionamenti.

2.4.2 eXtended Automation Runtime (XAR)

Il TwinCAT 3 Runtime offre un ambiente in tempo reale, in cui possono essere caricati, eseguiti e amministrati i moduli TwinCAT. I singoli moduli non devono necessariamente essere creati con lo stesso compilatore e quindi possono essere programmati indipendentemente e da diversi produttori o sviluppatori. Inoltre, non è importante se questi moduli vengono generati utilizzando il PLC, NC o utilizzando i linguaggi C/C++. I moduli generati possono essere richiamati ciclicamente da processi o da altri moduli (ad esempio utilizzando il linguaggio C/C++).

2.5 Il protocollo ADS - Automation Device Specification

L'ADS [18] è un protocollo del livello di trasporto all'interno del sistema TwinCAT 3 del sistema Beckhoff. Esso è stato sviluppato per lo scambio di dati tra differenti moduli software, ad esempio per la comunicazione tra NC e PLC. Inoltre, tale protocollo viene utilizzato per la comunicazione tra moduli software presenti in PC differenti, in particolare l'ADS è collocato in cima allo stack TCP/IP, come si vede in figura 2.12, consentendo in un sistema di rete l'accesso ai dati da qualsiasi punto desiderato.

Il sistema TwinCAT 3 consente ai singoli moduli software (ad esempio il TwinCAT PLC, il TwinCAT NC, ecc.) di essere trattati come dei dispositivi indipendenti: per ogni processo (task) è presente un modulo software (Server o Client).

I messaggi tra questi "oggetti" sono scambiati attraverso un'interfaccia ADS mediante un router, presente su ogni TwinCAT PC e su ogni bus controller, che gestisce e distribuisce tutti i messaggi all'interno del sistema e tra le connessioni TCP/IP.

Tali "message router" consentono a tutti i programmi Client e Server di scambiare comandi e dati.

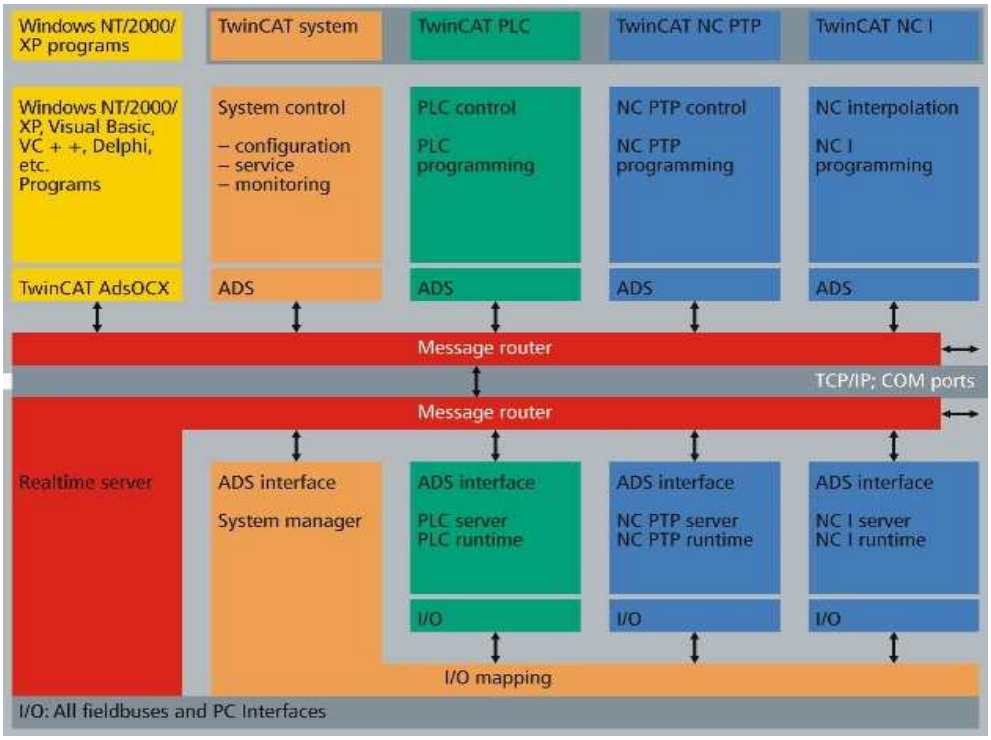

Figura 2.12: Panoramica dei protocolli Beckhoff

2.5.1 Le librerie ADS

L'interfaccia ADS consente ai software Server e Client di scambiare messaggi. Pertanto, mediante tale interfaccia, e cioè attraverso librerie DLL (Dynamic Link Library), è possibile accedere ai dati del progetto, leggerli e parametrizzarli. Al di fuori del sistema Beckhoff, sono disponibili un'elevata varietà di metodi che consentono di scambiare dati con altri strumenti software.

Al fine di consentire la partecipazione nella comunicazione ADS (come client ADS o, eventualmente, come server ADS) sono disponibili i seguenti oggetti software: le librerie *TcAdsDll* che forniscono funzioni per la comunicazione ai Client ADS e le librerie *ADS-Java-DLL* utilizzate per la programmazione in Java. Per il progetto del Client sono state utilizzate le librerie *AdsToJava.dll*.

CAPITOLO 3

Sviluppo del software per il controllo dei sottosistemi

3 Introduzione

In questo capitolo verranno descritte le tecniche adoperate per la configurazione dei parametri elettrici e meccanici relativi ad ogni singola movimentazione dei relativi sottosistemi.

Per ciascuna configurazione, è stata realizzata un'interfaccia grafica ingegneristica (ENI) in ambiente TwinCAT 3 per ogni singolo motore, che permette una comunicazione "user-friendly" e, allo stesso tempo, consente di verificare le funzioni implementate con il linguaggio IEC 61131-3 [19]. In un scenario reale di comunicazione a distanza, in rete, tale software rappresenta la parte Server.

Infine è stata realizzato un Client grafico in Java, per ogni singolo motore, con lo scopo di testare il corrispondente Server e fornire un interfacciamento con il Server stesso.

3.1 Il sottosistema polarimetrico

Questo sottosistema è composto da uno stepper motor che aziona uno stage lineare. Il controllo del motore, viene effettuato dal PC embedded CX5020 di Beckhoff in combinazione con il modulo Beckhoff EL7041.

In seguito si illustrerà come configurare il PC CX5020 in modo da poter controllare un dispositivo connesso al bus EtherCAT.

3.1.1 Requisiti

Per il sistema polarimetrico il requisito richiesto sul posizionamento della movimentazione è che il centro del sistema polarimetrico debba avere un offset rispetto all'asse ottico del telescopio non superiore a 1 mm.

La risoluzione del motore, come riportata nelle caratteristiche (si veda il paragrafo 2.3.1.1, tabella 2.1), è di 2.5 μm.

Le caratteristiche di risoluzione del motore unite alle funzionalità implementate hanno consentito di soddisfare con ampio margine il requisito richiesto.

3.1.2 Sviluppo del software TwinCAT per il sottosistema polarimetrico

Le potenzialità dell'ambiente di sviluppo TwinCAT 3 permettono di poter pilotare un attuatore, sia localmente, sia mediante un opportuno Client attraverso la suite TCP/IP. A tal proposito ricordiamo che il TwinCAT 3 è di per sé un Server con il quale è possibile comunicare attraverso l'uso di vari protocolli tra cui l'ADS. Il codice presente nel progetto è stato scritto utilizzando lo standard IEC 61131-3 utilizzato per la programmazione di sistemi di controllo basati su Controllori Logici Programmabili (PLC). Tale norma prevede due tipi di linguaggi di programmazione:

- testuali
- grafici

I linguaggi testuali sono:

- Structured Text (ST)
- Instruction List (IL)

I linguaggi grafici sono:

- Function Block Diagram (FBD)
- Ladder Diagram (LD)
- Sequential Flow Chart (SFC)

Il linguaggio ST è un linguaggio testuale ad alto livello che presenta una sintassi simile a quella del linguaggio Pascal, ma è stato sviluppato espressamente per applicazioni di controllo. Esso è molto semplice da comprendere e da leggere, è adatto per calcoli aritmetici anche complessi, inoltre i programmi possono essere costruiti con molta libertà ed è possibile inserire caratteri di tabulazione, spazi o commenti in maniera analoga a quanto si fa per altri linguaggi di programmazione. Inoltre esso è il linguaggio più potente e portabile rispetto agli altri linguaggi della norma IEC 61131-3.
La IEC 61131-3 contempla 3 differenti unità di software, dette POU (Program Organization Unit):

- Programmi (PRG)
- Blocchi Funzione (FB)
- Funzioni (FUN)

Un programma (PRG) può essere chiamato da un task o da un altro PRG. I program possono chiamare blocchi funzione, funzioni e programmi, inoltre essi mantengono memoria dello stato delle variabili in cicli PLC successivi.

Un blocco funzione deve essere istanziato prima del suo utilizzo, può avere più variabili sia in ingresso che in uscita e mantiene memoria dello stato delle variabili interne in cicli PLC successivi.

Una funzione non deve essere istanziata per essere utilizzata, inoltre può avere più variabili in ingresso e una sola uscita, la quale è dello stesso tipo con cui è stata definita la funzione stessa. Una funzione non mantiene memoria dello stato delle variabili interne.

Sia i blocchi funzione che le funzioni possono essere definite dall'utente o essere prelevate da librerie importate nel progetto.

Il codice è stato scritto utilizzando in linguaggio ST e prevede l'utilizzo di elementi fondamentali dette POU come function, function block, program sopra citate. Maggiori dettagli verranno presentati nel paragrafo 3.2.5.4.

3.1.2.1 Configurazione delle variabili

Attraverso il menù di configurazione, in particolare nel modulo *Configurazione NC,* è possibile accedere alle singole movimentazioni (Assi). All'interno di tale sezione sono presenti tutte le sottosezioni, ognuna delle quali si riferisce al controllo di un singolo asse:

- Asse 1
- Asse 2
- Asse 3

Nello specifico per ogni asse si hanno le seguenti sottosezioni:

- Enc
- Drive
- Ctrl
- Ingressi
- Uscite

Il polarimetro ha come riferimento *Asse 1*. Selezionando tale sezione appare una finestra di dialogo avente il seguente menù (figura 3.1):

- Generale
- Impostazioni
- Parametro
- Dinamica
- In linea
- Funzioni
- Accoppiamento
- Compensazione

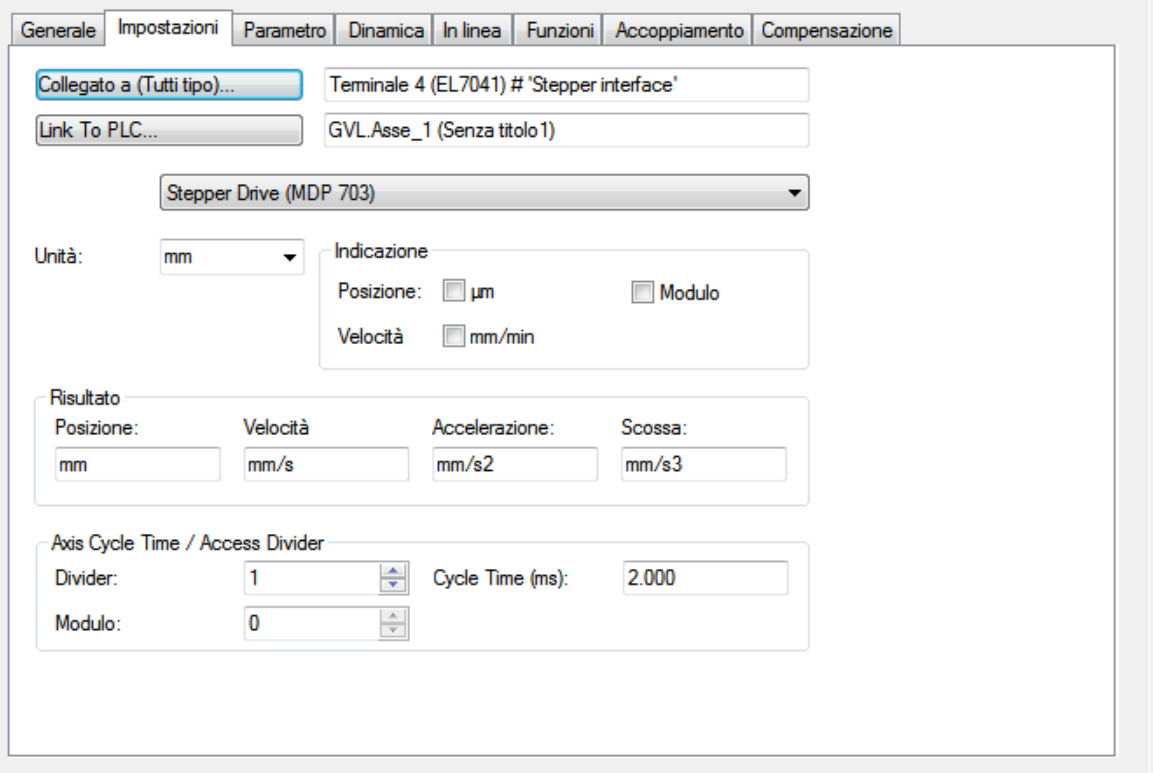

Figura 3.1: Impostazioni Asse 1

Dal menù *In linea* si ha la possibilità di muovere il motore in modo indipendente, senza implementazioni software. Questa funzionalità permette solo di fare movimenti con controllo di velocità o di posizione, ma non è possibile effettuare la procedura di homing. Questa è possibile solo mediante la seguente procedura di linking.

Linking significa associare ad una variabile di immagine di processo PLC, che si può definire variabile "software", un'altra variabile d'immagine di processo dei terminali fisici I/O (di ingresso/uscita), che si può definire variabile "hardware", per cui si crea un link o riferimento tra una variabile software e una variabile fisica o dei dispositivi I/O.

Inizialmente si definiscono le variabili globali per il sottosistema polarimetrico in modo che esse siano visibili da tutto il progetto. Tali variabili sono:

- Asse_1 → è una variabili di tipo AXIS_REF [25] e rappresenta la movimentazione dello stage lineare del modulo polarimetrico
- hs1 → è una variabile booleana che rappresenta lo switch di fine corsa presente sul lato opposto della manopola (si veda la figura 3.7)

- hs2 → è una variabile booleana che rappresenta lo switch di fine corsa presente sullo stesso lato della manopola (si veda la figura 3.7)

Nel nostro caso, alle variabili globali che vengono definite inizialmente, Asse_1, hs1, hs2 , si associano i vari Input/Output dei vari moduli e lo status degli switches di fine corsa. Nel caso della variabile Asse_1, i vari links vengono creati automaticamente mentre per quanto concerne gli switches di fine corsa, questi vengono realizzati manualmente.

La figura 3.2 illustra il codice di definizione di tali variabili. Occorre notare che esse devono essere visibili da tutto il sistema software per cui vengono definite come variabili globali.

```
VAR_GLOBAL

    Asse_1          :  AXIS_REF;  // asse_1 deve essere di tipo AXIS_REF
    hs1     AT%I* :  BOOL; // lo switch 2 è un input=ingresso)
    hs2     AT%I* :  BOOL; // lo switch 1 è un input=ingresso)

END_VAR
```

Figura 3.2: Codice GVL per il motore Translation Stage 8MT175-150

In particolare, alla scansione dei vari moduli del PC embedded, tutti i links principali tra il modulo *Configurazione NC* e il Terminale 4 (quello corrispondente al modulo EL7041) sono automaticamente creati. Le variabili "software" associate al Terminale 4 vengono linkate automaticamente con le variabili "hardware" presenti nella sezione Encoder del modulo *Configurazione NC*. Analogamente tutte le variabili "software" presenti in nella sezione Status di Terminale 4 vengono linkate con le variabili "hardware" presenti nella sezione Drive del modulo *Configurazione NC* (Numeric Control). Ciò si può vedere in figura 3.3 (Link L1) dove sono riportate varie sezioni che compongono l'albero di configurazione del progetto (lo schema dei moduli hardware collegati e dei moduli software di TwinCAT 3), presente nell'ambiente TwinCAT 3.

Inoltre vengono create due cartelle nel modulo PLC:

- *PLC Task_Standard Inputs*
- *PLC Task_Standard Outputs*
- Facendo il download della configurazione fin qui implementata (figura 3.2), vengono creati automaticamente tutti i links (L2), si veda la figura 3.4, tra le variabili "software" di Output per il PLC, presenti nell'immagine di processo *PLC Task_Standard Outputs*, e le variabili "hardware" di Input per il modulo NC, presenti nella sottocartella *From PLC* di Asse 1. Analogamente vengono creati automaticamente tutti i links (L2) tra le variabili "software" di Input per il PLC, presenti nell'immagine di processo *PLC Task_Standard Inputs*, e le variabili "hardware" di Output per il modulo NC, presenti nella sottocartella *To PLC* di Asse 1.

In questo modo, nel caso del modulo EL7041, tali link vengono realizzati automaticamente dal sistema. Ricordiamo però che le variabili GVL.hs1 e GVL.hs2, presenti in PLC Task_Standard Inputs, non sono linkate, per cui tale operazione deve essere fatta manualmente.

Si consideri in dettaglio le figure seguenti. In figura 3.3 si hanno a sinistra le immagini di processo di input e quelle di output delle sezioni Enc e Drive di Asse 1 (modulo NC), mentre a destra sono raffigurate le corrispondenti immagini di processo di input e di output della sezione Input/Output per il Terminale 4. In figura 3.4 si hanno a sinistra le immagini di processo di input della Ctrl di Asse 1 (modulo NC), *FromPLC* e le immagini di processo di output della sezione Ctrl di Asse 1 (modulo NC), *ToPLC*. Tali immagini sono linkate rispettivamente (parte destra) con le immagini di processo di output e con quelle di input della sezione Input/Output per il Terminale 8.

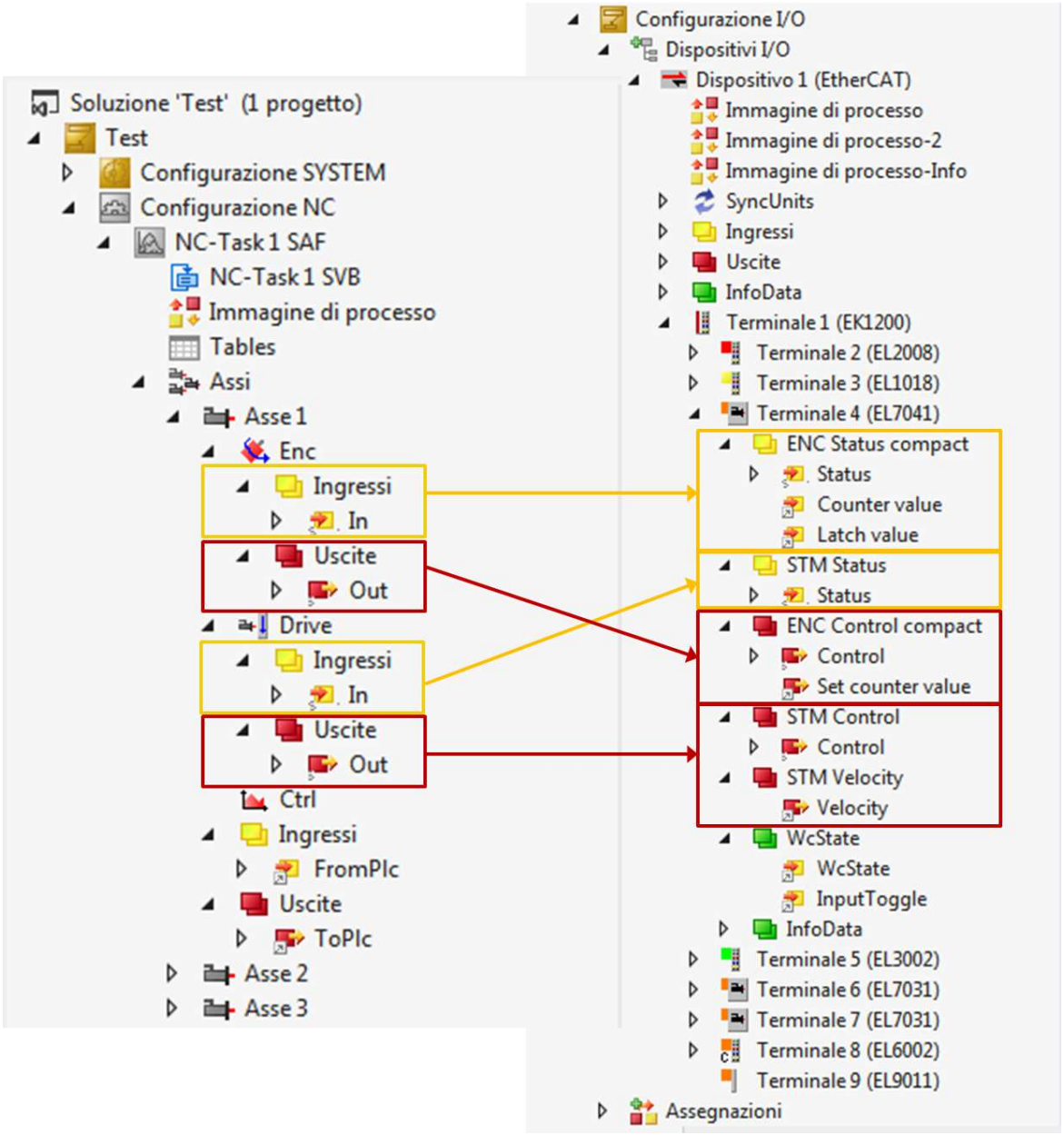

Figura 3.3: Links L1

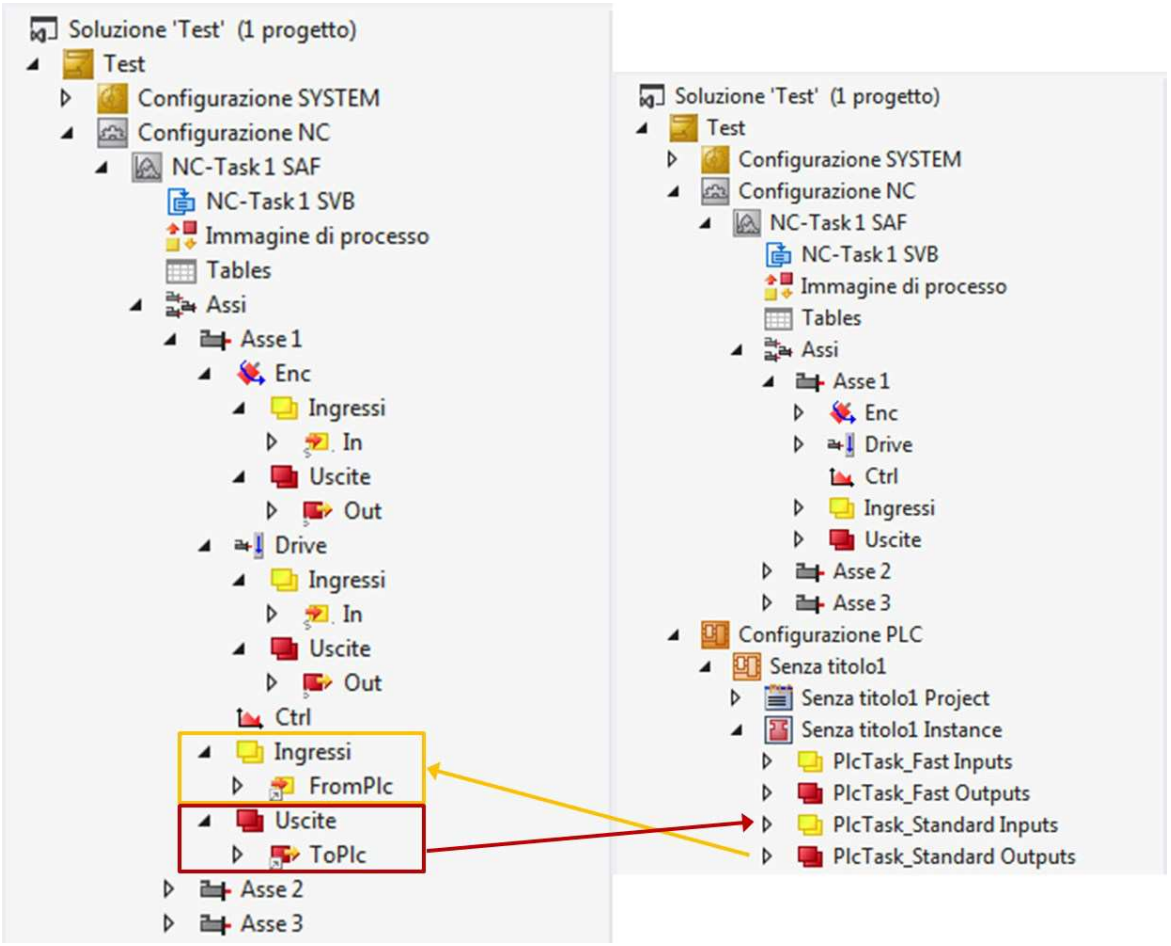

Figura 3.4: Links L2

3.1.1.2 Configurazione dei parametri meccanici

Prima di poter realizzare una qualsiasi movimentazione del motore, occorre impostare correttamente i suoi parametri meccanici ed elettrici.

Motori Stepper

Un motore di tipo stepper è un motore che si muove per unità discrete di movimento detti passi o steps.

La prima operazione da fare è impostare correttamente i parametri meccanici della movimentazione. Per realizzare ciò, occorre posizionarsi sulla scheda *Parametro* della sezione Asse 1. Dai dati di targa del motore [32], è noto che:

- 1 step = 1.8°, l'angolo percorso dalla vite per 1 step
- 1 step = 2.5 µm, movimento lineare dello stage per 1 step
- 1 tacca della manopola = 2 step, un incremento di una tacca equivale ad un incremento di 2 step.

Si calcola il numero di step per un giro completo del rotore; esso è dato da:

$$\frac{360°}{1.8°/step} = 200\ step \tag{3.1}$$

Essendo 1 tacca = 2 step, per un giro completo la manopola collegata al motore ruota di :

$$\frac{200\ step}{2\ step/tacca} = 100\ tacche \tag{3.2}$$

A questo punto si calcola la velocità di riferimento e la velocità massima.

Per quanto riguarda la v_{max}, si ottiene:

$$v_{max} = base\ frequency/motor\ frequency \tag{3.3}$$

Il motor frequency è un dato di targa fisico del motore e rappresenta il massimo numero di step o fullstep per 1 giro o revolution del rotore. Per una risoluzione R di 1 step = 64 microstep, tale parametro ha il valore di 200 step/rev.

Il parametro base frequency o speed range, indica di quanti fullstep al secondo il motore è in grado di muoversi. Per le impostazioni di default si ha una base frequency o speed range pari a 2000 step/s, ciò vuol dire che è possibile inviare un comando di movimento assoluto o relativo di 2000 step/s al massimo. Esso varia nell'intervallo (0,..,2000) step/s e l'estremo superiore è, di default, il valore impostato. Ne segue che v_{max}, essendo dipendente dal motor frequency, rappresenta la massima velocità che può essere calcolata in base ai valori del dato di targa motor frequency e del valore da noi impostato del parametro base frequency.

$$v_{max} = \frac{2000\ step/s}{200\ step/rev} = 10\ rev/s \tag{3.4}$$

Questa, come si vedrà più avanti, è la stessa v_{ref}, qui espressa in rev/s (rivoluzioni al secondo) che sarà tradotta in mm/s. Prima di esaminare la velocità di riferimento, v_{ref}, occorre porre l'attenzione sul parametro Scaling Factor (SF). Tale parametro rappresenta il numero di millimetri di movimento dello stage che corrispondono ad 1 fullstep del motore. Con la presenza di un encoder collegato al modulo EL7041, il PLC legge tale valore dall'encoder stesso, ma non essendo presente alcun encoder, tale parametro deve esser impostato. Lo scaling factor, senza encoder, è dato da [25]:

$$SF = \frac{distance\ per\ rev}{fullstep * microstep} \tag{3.5}$$

$$SF = \frac{360°/rev}{200\ step/rev * 64\ micro\ step/step} = 0.028125°/step \tag{3.6}$$

Se si vuole esprimere il parametro SF in mm/step, dalla (3.6) si ottiene:

$$SF = \frac{0.5\ mm/rev}{200\ step/rev * 64\ microstep/step} = 0.0000390625\ mm/microstep \quad (3.7)$$

Se si imposta correttamente lo scaling factor presente nella sezione *Enc* di Asse 1, si potrà leggere il valore del numero di step fatti dal motore. La variabile che ci fornisce il numero di step è *nDataOut1*.

La velocità di riferimento è la velocità massima teorica per la quale il motore si muove con estrema precisione, per cui il motore si muove anche se io imposto una v_{ref} più grande ma lo spostamento non è preciso. Ad esempio se si fa muovere il motore di 10 mm, si misurerà con il calibro uno spostamento differente. La velocità di riferimento, espressa in gradi al secondo, è data da:

$$v_{max} = v_{ref} = \frac{rev}{s} * 360° \quad (3.8)$$

In mm/s si ottiene:

$$v_{ref} = \frac{step}{rev} * \frac{Mov\ lineare}{step} * \frac{rev}{s} \quad (3.9)$$

$$v_{max} = v_{ref} = \frac{200}{1} * \frac{2.5\ \mu m}{1} * \frac{10}{1} = 5\ mm/s \quad (3.10)$$

Allo stesso tempo v_{ref} è un parametro dal quale il software si ricava le seguenti informazioni:

1) Il valore di fullstep in mm

$$1\ fullstep = \frac{5\ mm/s}{2000\ step/s} = 0.0025\ mm/step \tag{3.11}$$

2) Il valore del microstep in mm

$$1\ microstep = \frac{0.0025\ mm/step}{64\ microstep/step} = 0.0000390625\ mm/microstep \tag{3.12}$$

3) Il numero di fullstep di cui muoversi corrispondenti a x mm, per un movimento di 1 mm, in base alla seguente proporzione, è dato da:

$$2000\ fullstep : 5\ mm = x\ fullstep : 1\ mm \tag{3.13}$$

$$x = \frac{1\ mm * 2000\ fstep}{5\ mm} = 400\ step \tag{3.14}$$

Pertanto, per un movimento di 1 mm, il rotore si muove di 400 step.
Invertendo, si ha che il motore si muove di 1 step per ogni 0.0025 mm

$$\frac{1}{400\ fstep} = 0.0025\ mm \tag{3.15}$$

4) Il numero di full step al secondo di cui muoversi corrispondenti a x mm/s, per un movimento di 1 mm/s, in base alla seguente proporzione:

$$5\ mm/s : 2000\ fstep/s = 1mm/s : x\ fstep/s \tag{3.16}$$

è pari a:

$$x = \frac{2000\ fstep/s * 1\ mm/s}{5\ mm/s} = 400\ fstep/s \tag{3.17}$$

Pertanto, per un movimento di 1 mm/s, il rotore si muove di 400 step/s.

Invertendo, si ha che il sistema TwinCAT 3 manda un impulso al motore ogni 0.0025 s, cioè il motore si muove di 1 step per ogni 0.0025 s.

$$\frac{1}{400\ fstep/s} = 0.0025\ s \tag{3.18}$$

La v_{max} presente in Asse 1 invece è quella oltre la quale il motore vibra e fa rumore e non può superare la v_{ref}.

In conclusione occorre impostare:

- v_{ref} = 5 mm/s
- v_{max} = 5 mm/s
- SF = 0.0000390625 mm/microstep

Facciamo un'ultima ma non meno importante considerazione: se si modifica il parametro speed range da 2000 a 8000 full step/s e allo stesso tempo si cambia v_{ref} da 5 mm/s al valore di 20 mm/s, si osserva che il motore si muove più velocemente, ma la distanza che deve percorrere rimane sempre la stessa cioè il motore continua a muoversi con precisione, infatti se si considera il rapporto base frequency/v_{ref}, si ottiene:

$$R_M \triangleq \frac{base\ frequency}{v_{ref}} = \frac{8000\ fstep/s}{20\ mm/s} = 400\ step/mm \tag{3.19}$$

Quindi il numero di step di cui il motore deve muoversi, per un movimento di 1 mm, rimane inalterato ed è pari a 400. Si definisce tale rapporto R_M come Rapporto del Motore in quanto esso rappresenta le proprietà intrinseche del motore stesso. Per ottenere una movimentazione più veloce ed egualmente precisa, è possibile modificare i suoi parametri, base frequency o speed range e v_{ref}, ma tale rapporto deve rimanere inalterato.

Si considerino quali sono i vantaggi e gli svantaggi di impostare solo una v_{ref} più grande e di lasciare inalterato il parametro base frequency *o* speed range:

Vantaggio: avere una velocità massima teorica più grande comporta che il motore si muove più velocemente;

Svantaggio: il motore si muove più velocemente ma la movimentazione è meno fine perché modificando il v_{ref} è come se modificassimo il rapporto di risoluzione R da 64 microstep/step a 64/4=16 microstep/step, risultando una diminuzione del rapporto di risoluzione. Più alto è il rapporto di risoluzione, più fine sarà la movimentazione.

3.1.2.2.1 Parametri Asse 1

I parametri seguenti, presenti nella sezione Velocities vengono impostati ad un valore che è stato scelto sulla base della relazione (3.10) e in base a prove in laboratorio.

- velocità manuale (rapida) = 1.0 mm/s
- velocità manuale (lenta) = 0.5 mm/s
- velocità calibrazione (in avanti) = 1.0 mm/s
- velocità calibrazione (all'indietro) = 0.5 mm/s
- incremento spinta (in avanti) = 0.5 mm/s
- incremento spinta (all'indietro) = 0.5 mm/s

3.1.2.2.1.2 Parametri Encoder

Come tipo di Encoder si sceglie un Encoder simulato non essendo presente alcun encoder collegato al modulo EL7041. Il fattore di scala si sceglie in base alla relazione mentre per l'homing , avendo scelto hs2 come sensore di fine corsa o di homing e poiché lo stage si muove di default verso hs1, occorre selezionare al valore TRUE la voce "Inverti

direzione ricerca eccentrico calibrazione". Una spiegazione di quanto affermato sarà illustrata nel paragrafo 3.1.2.4.

- Type: Codificatore simulazione (tipo di encoder)
- Scaling Factor (fattore di scala) = 0.0000390625 mm/step
- Sezione Homing: Inverti direzione ricerca eccentrico calibrazione: TRUE

3.1.2.3 Configurazione dei parametri elettrici

Successivamente occorre modificare i parametri elettrici presenti nella scheda *CoE-Online*.

Nel manuale on-line relativo al modulo EL7041 [25], è riportato sia il valore predefinito (default) sia l'unità di misura per ogni parametro. Tra tutti, occorre impostare i seguenti:

- Maximal Current = 250 mA
- Nominal Voltage = 12000 mV
- Motor Coil Resistance = 3000 (unità = 0.01 Ohm) → 30 Ohm*100
- Motor Fullsteps = 200
- Speed range = 2000 Fullsteps/s
- Feedback Type: Internal Counter(1)

3.1.2.4 Configurazione calibrazione (homing)

Si definisca un'istanza della FB (Function Block) MC_Home nel file (PRG) POLARIMETRO:

```
Home        : MC_Home;
```

La procedura di homing non viene fatta automaticamente dall'istanza della function block (FB) MC_HOME, ma si deve programmare. Dopo aver installato la libreria Tc_MC2, occorre valutare la polarità delle due direzioni del movimento. Andando sulla scheda *In linea* di Asse 1 e selezionando la casella Movimento in avanti, che per convenzione rappresenta la direzione positiva, si osserva che lo stage si muove verso la direzione

opposta alla manopola quindi avendo impostato lo switch hs2 dal lato della manopola, quando esso viene raggiunto dallo stage, occorre disattivare la direzione positiva mentre per lo switch hs1, impostato dal lato opposto alla manopola, occorre disattivare la direzione negativa, si veda la figura 3.7. Si consideri di voler programmare la procedura di homing sullo switch di fine corsa hs2.

Si tenga presente che la procedura di homing corretta è la seguente: quando lo stage incontra lo switch, si ferma temporaneamente e torna indietro di poco, lasciando lo switch nello stato di riposo e non sotto pressione, pertanto per ottenere la corretta procedura di homing si deve individuare la variabile booleana che diventi FALSE quando la procedura di homing è terminata. Per fare ciò, occorre osservare come variano le variabili durante l'esecuzione del programma. Si consideri la parte alta della finestra principale (main window), dove sono riportate le FB sotto la colonna *Espressione*. Espandendo la voce Home, si notano la presenza di altre voci come Done, Busy, Active, ecc. Rilasciando il commutatore GO_HOME (che vedremo in seguito nel Pannello di visualizzazione), si osserva che la variabile booleana *Active* cambia stato passando da TRUE a FALSE, pertanto è questo il parametro che restituisce l'informazione sulla terminazione della procedura di homing. A questo punto, poiché si desidera che la procedura di homing sia avviata solo durante il fronte di salita del segnale di trigger, viene cambiato il pulsante da commutatore a tasto in modo che attiva il comando fin quando è premuto e disattiva il comando quando è rilasciato. In questo modo, terminata la procedura di homing, la nuova variabile *Home.Active* assume valore FALSE e la procedura di homing viene eseguita correttamente senza il "cut-off" del movimento, consentendo allo stage di tornare indietro. Inoltre si ha che la funzione di homing viene eseguita correttamente rilasciando lo switch di fine corsa hs2 e la direzione negativa non viene disabilitata (il LED FW ENABLED è ON). In questo istante se si preme ulteriormente il tasto jog- o jog--, andando in contatto con lo switch hs2, entra in gioco la condizione sullo switch hs2 che disabilita la direzione negativa. Il codice per la procedura di homing è riportato in figura 3.5.

```
IF NOT Home.Active THEN
   IF hs2 THEN   // se lo switch 2 (quello dal lato della manopola) è in contatto
  Enable_asse1.Enable_Negative:=FALSE;
   ELSE
  Enable_asse1.Enable_Negative:=TRUE;
   END_IF
END_IF
```

Figura 3.5: Codice homing

Occorre richiamare l'istanza della FB MC_Home:

```
Home(
   Axis:=Asse_1,
   HomingMode:=MC_DefaultHoming,
   bcalibrationcam:=hs2,
   Execute:=bhome,
   Busy => bBusy);
```

Figura 3.6: Codice di chiamata dell'istanza Home

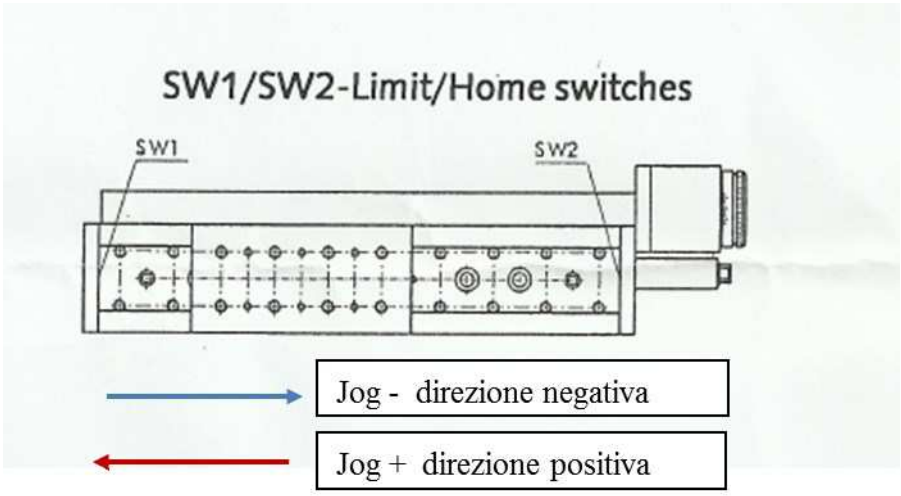

Figura 3.7: Direzioni per il movimento dello *stage* Standa

3.1.2.5 Configurazione del movimento con controllo della velocità (jog)

Impostando questa modalità di movimento, l'operatore deve fornire la velocità, l'accelerazione, la decelerazione e la direzione di moto. Il motore si muove lungo la direzione selezionata con la velocità desiderata fino a quando non viene rilasciato il tasto relativo alla funzione stessa.

Ci sono 4 possibili movimentazioni:

- Jog-- = movimentazione veloce nella direzione negativa
- Jog- = movimentazione lenta nella direzione negativa
- Jog+ = movimentazione lenta nella direzione positiva
- Jog++ = movimentazione veloce nella direzione positiva

La FB che si utilizza per realizzare tali comandi dal PLC cioè dall' interfaccia grafica, è la FB MC_Jog. Come per l'homing, non è sufficiente istanziare la FB MC_Jog per ottenere tali comandi, pertanto si deve implementare un codice opportuno; in particolare per realizzare i 4 tasti corrispondenti alle 4 funzioni di jog occorre:

1. Definire nella POU Polarimetro, le 4 variabili booleane o istanze per i 4 tipi di movimenti.
2. Definire le condizioni in base alle quali, se si invia il comando relativo ad un determinato movimento, venga attivata tale FB.

Per sapere quali sono i parametri di ingresso e di uscita di tale FB, considerando le librerie presenti nel progetto, facilmente accessibili all'interno dell'ambiente TwinCAT 3, è possibile visualizzare una finestra dove sono presenti due schede: una si riferisce al grafico della FB, dove viene schematizzato il blocco della FB con i suoi ingressi e le uscite, e la scheda "documentazione" che riporta i dettagli sui parametri I/O.

- Grafico

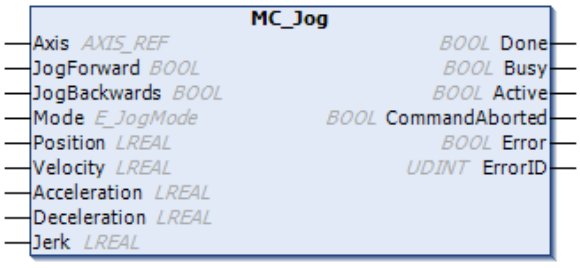

Figura 3.8: Schema del FB MC_Jog

- Documentazione

FUNCTION_BLOCK **MC_Jog**

Nome	Tipo dati	Ereditato da	Indirizzo	Valore iniziale	Commento
Axis	AXIS_REF				
JogForward	BOOL				
JogBackwards	BOOL				
Mode	E_JogMode				
Position	LREAL				
Velocity	LREAL				
Acceleration	LREAL				
Deceleration	LREAL				
Jerk	LREAL				
Done	BOOL				
Busy	BOOL				
Active	BOOL				
CommandAborted	BOOL				
Error	BOOL				
ErrorID	UDINT				

Figura 3.9: Documentazione del FB MC_Jog

Si noti che il parametro d'ingresso **Mode** è di tipo E_JogMode. Tale tipo è un Enumerate e dalla sua documentazione è possibile vedere i valori che esso può assumere:

ENUM **E_JogMode**

Nome	Tipo dati	Ereditato da	Indirizzo	Valore iniziale	Commento
MC_JOGMODE_STANDARD_SLOW	INT				motion with standard jog parameters for slow motion
MC_JOGMODE_STANDARD_FAST	INT				motion with standard jog parameters for fast motion
MC_JOGMODE_CONTINOUS	INT				axis moves as long as the jog button is pressed using parameterized dynamics
MC_JOGMODE_INCHING	INT				axis moves for a certain relative distance
MC_JOGMODE_INCHING_MODULO	INT				axis moves for a certain relative distance - stop position is rounded to the distance value

Figura 3.10: Parametri di E_JogMode

I parametri ai quali siamo interessati sono:

- MC_JOGMODE_STANDARD_SLOW, per le movimentazioni lente
- MC_JOGMODE_STANDARD_FAST, per le movimentazioni veloci

A questo punto si fa uso di 2 istruzioni IF, una per le movimentazioni lente e l'altra per quelle veloci, in cui si utilizzano i suddetti parametri. Ad esempio, se si invia il comando di movimentazione lenta positiva o negativa, il PLC inoltrerà al modulo EL7041 il comando corrispondente ad una movimentazione lenta. Analoga procedura viene seguita per i comandi per le movimentazioni veloci (il primo e il secondo IF in figura 3.11). Ciò non è sufficiente per garantire un corretto funzionamento per tali movimentazioni, infatti occorre inserire altre 2 condizioni IF-ELSE (il terzo e il quarto IF in figura 3.11) in modo

che se si invia il comando di movimentazione positiva, lenta o veloce, il PLC invierà al modulo il comando corrispondente ad una movimentazione positiva. Analoga procedura viene seguita per i comandi di movimentazioni negative. Dopo aver istanziato la FB:

```
Move_JOG        : MC_Jog;
```

si implementa il codice opportuno per la corretta esecuzione di tale movimento, che riportiamo in figura 3.11.

```
//codice-condizioni per le funzioni jog (movimento a impulsi)

IF bjogFW OR bjogBW THEN
  Move_JOG.Mode:= MC_JOGMODE_STANDARD_SLOW;
END_IF

IF bjogFFW OR bjogFBW THEN
  Move_JOG.Mode:= MC_JOGMODE_STANDARD_FAST;
END_IF

IF bjogFW OR bjogFFW THEN
  Move_JOG.JogForward:= TRUE;
ELSE
  Move_JOG.JogForward:= FALSE;
END_IF

IF bjogBW OR bjogFBW THEN
  Move_JOG.JogBackwards:= TRUE;
ELSE
  Move_JOG.JogBackwards:= FALSE;
END_IF
```

Figura 3.11: Codice per il movimento jog

Infine si deve richiamare l'istanza della FB MC_Jog, mediante la seguente isrtruzione:

Move_JOG(Axis:=Asse_1);

3.1.2.6 Configurazione del movimento con controllo della posizione (assoluto e relativo)

3.1.2.6.1 Movimento assoluto

Impostando tale modalità, l'operatore deve fornire la posizione finale i parametri di velocità, accelerazione, decelerazione necessari per generare la traiettoria che deve essere seguita per il raggiungimento della posizione finale da parte dello stage. Quando il motore giunge nella posizione desiderata, il movimento si ferma automaticamente. Pertanto mediante il posizionamento assoluto, viene specificata una posizione finale B che viene raggiunta a partire da una posizione iniziale A.

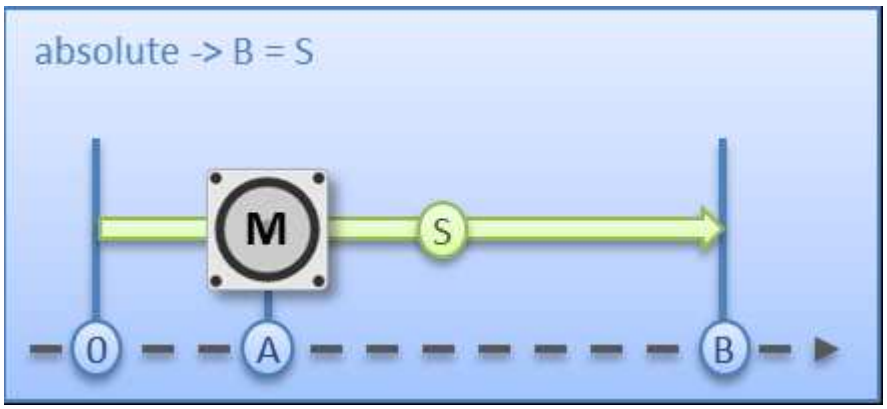

Figura 3.12: Movimento assoluto

Procedendo in modo analogo al caso precedente, dalle librerie è possibile accedere ad una finestra con due schede: una si riferisce al grafico della FB, dove viene schematizzato il blocco della FB con i suoi ingressi e le uscite, e la scheda documentazione che riporta i dettagli sui parametri I/O.

Inizialmente si istanzia la FB:

Move_ABS : MC_Moveabsolute;

e successivamente si richiama la sua istanza:

```
Move_ABS(
    Axis:=Asse_1,
    execute:= bABS,
    position:=lpositionABS ,
    velocity:=lvelocityABS );
```

Figura 3.13: Codice di chiamata dell'istanza Move_ABS

3.1.2.6.2 Movimento relativo

Impostando tale modalità, l'operatore deve fornire lo spostamento desiderato che si somma algebricamente alla posizione iniziale per l'ottenimento della posizione finale dello stage. Inoltre è necessario fornire i parametri di velocità, accelerazione, decelerazione necessari per generare la traiettoria che deve essere seguita per il raggiungimento della posizione finale. Quando il motore giunge nella posizione desiderata, il movimento si ferma automaticamente. Pertanto nel posizionamento relativo, l'operatore specifica una lunghezza di spostamento S, che viene aggiunta alla posizione corrente A, producendo la posizione di destinazione B.

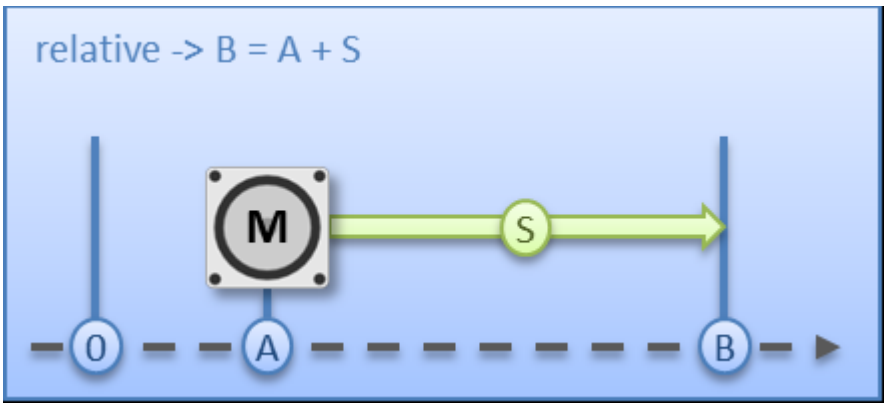

Figura 3.14: Movimento relativo

Analogamente al movimento assoluto, si considera la FB MC_Moverelative. Inizialmente si istanzia la FB:

```
Move_REL        : MC_Moverelative;
```

e successivamente si richiama la sua istanza

```
Move_REL(
        Axis:=Asse_1,
        execute:=bREL,
        distance:=ldistanceREL,
        velocity:=lvelocityREL );
```

Figura 3.15: Codice di chiamata dell'istanza Move_REL

Nella tabella 3.1 sono riportate le Function Blocks definite e le relative istanze.

ISTANZA	**FUNCTION BLOCK**
Enable_asse_1	MC_Power
Move_ABS	MC_Moveabsolute
Move_REL	MC_Moverelative
Move_JOG	MC_Jog
Reset	MC_Reset
Home	MC_Home
Stop	MC_Stop
Write_parameter	MC_Writeparameter
Setpos	MC_Setposition
Status_parameter	MC_ReadParameter

Tabella 3.1: Function Blocks definite per il sottosistema polarimetrico

In figura 3.16 sono rappresentate le Function Blocks definite per il polarimetro.

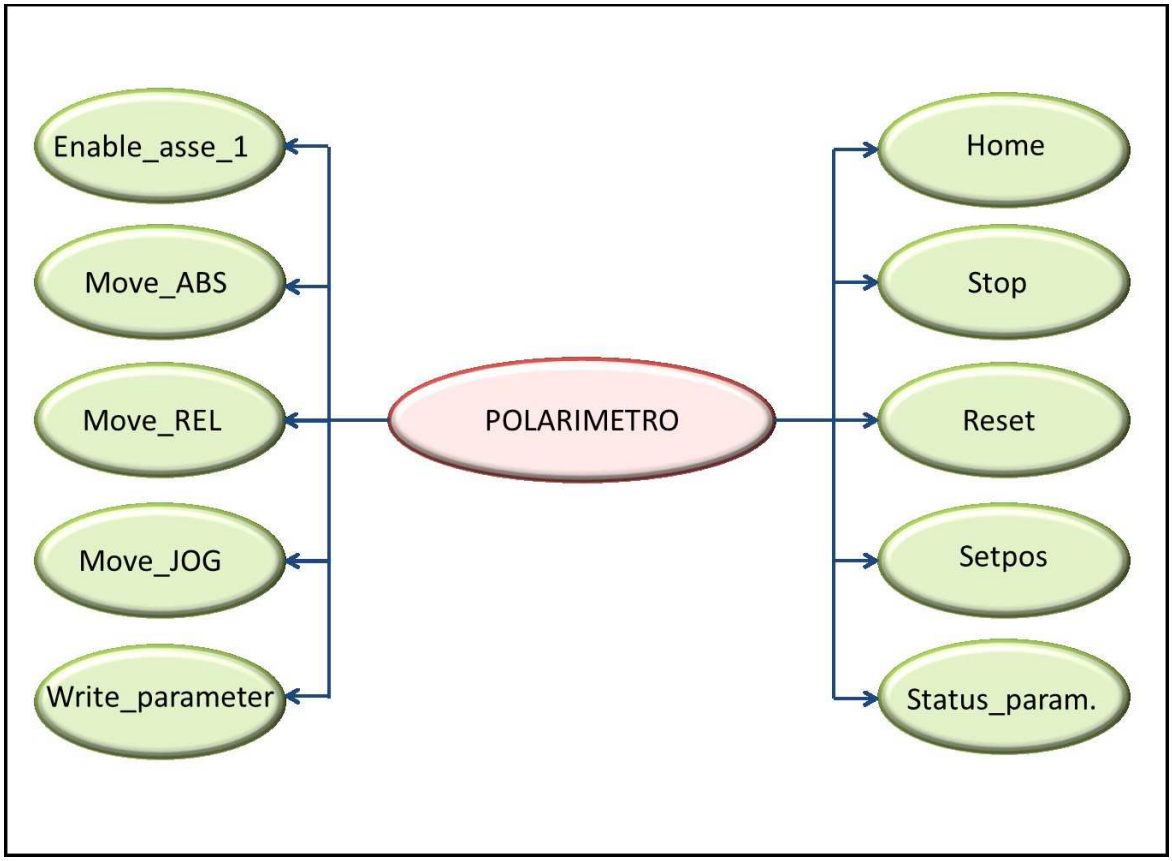

Figura 3.16 : Function Blocks definite per il sottosistema polarimetrico

3.1.2.7 Implementazione delle variabili di stato del motore

Si considerino le variabili, definite nel Server TwinCAT, che restituiscono le informazioni da noi desiderate.

1) Set Init

In tale sezione è definita la variabile *iSetInit* che restituisce il valore 0 oppure 1 a seconda che sia stato ricevuto correttamente il comando di homing.

2) Get Init

In tale sezione è definita la variabile *iGetInit* che restituisce il valore 0 oppure 1 a seconda che il comando di homing sia stato eseguito correttamente.

3) Get Status

In tale sezione è definita la variabile *iGetStatus* che restituisce il valore compreso tra (0÷5) a seconda dello stato del motore.

4) Get Error

In tale sezione è definita la variabile *iGetError* che restituisce il valore compreso tra (0÷5) a secondo del tipo di errore.

3.1.2.8 Pannello di visualizzazione

Un'interfaccia grafica utente (GUI) permette agli utenti di interagire con i dispositivi elettronici attraverso icone grafiche e indicatori visivi, in contrasto con interfacce basate su testo nelle quali l'utente impartisce comandi testuali in input mediante tastiera alfanumerica e riceve risposte testuali in output dall'elaboratore su riga di comando.

Le interfacce grafiche ingegneristiche sono implementate per il collaudo e la manutenzione di ciascun sistema che compone la nuova interfaccia. Tali interfacce sono complete in quanto consentono di visualizzare e manipolare tutti i parametri relativi ad un dato sottosistema da parte dell'amministratore.

Per creare un'interfaccia grafica dalla quale poter visualizzare sia i comandi da inviare che gli stati delle variabili più significative, occorre creare un pannello di visualizzazione grafica, detto VISU. Il sistema di programmazione contiene un editor di visualizzazione integrata, permettendo all'utente di creare oggetti di visualizzazione nello stesso ambiente di sviluppo dell'applicazione. Ciò è possibile grazie alla libreria Visudialogs e alla directory VISUs preposta per questo scopo. Ci sono due modalità:

1. In modo diretto nel sistema di programmazione

In modalità on-line si ottiene immediatamente una visualizzazione delle VISU create all'interno del sistema di programmazione.

2. Target-Visualization

Mediante tale modalità, utilizzata sui controllers con display integrato, è possibile fare un'operazione di upload dei dati di visualizzazione e dell'applicazione dal sistema di programmazione al sistema di destinazione. Tali dati vengono automaticamente visualizzati. Questa soluzione può essere realizzata da tutti i dispositivi che sono programmabili con TwinCAT PLC Control.

Si utilizza la prima modalità in quanto il PLC non dispone di un display integrato. La prima operazione consiste nel creare il file Visualization Manager che gestisce le impostazioni generali per tutti gli oggetti creati nella VISU di un progetto PLC. Successiva-

mente si aggiunge un file VISU dove si collocano gli oggetti che rappresentano le variabili e le funzioni che si vogliono visualizzare. Il passo successivo è quello dell'operazione di linking dell'oggetto con la variabile, sia essa software definita nel file POLARIMETRO, sia hardware. In figura 3.17 è presente un'istantanea del pannello di visualizzazione per il polarimetro.

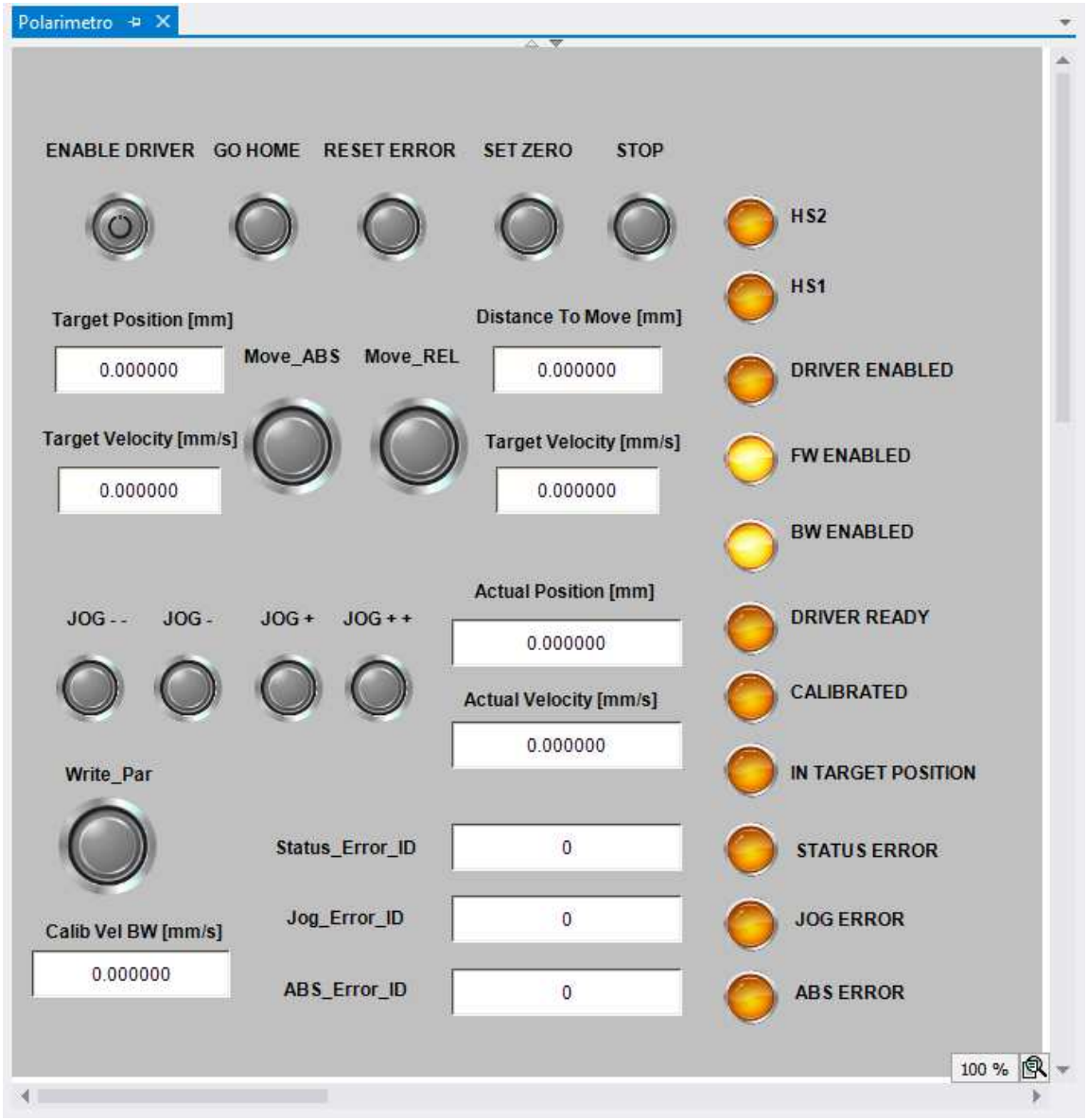

Figura 3.17: Pannello di visualizzazione TwinCAT 3 per il sottosistema polarimetrico

Premendo l'interruttore indicato sotto la dicitura ENABLE DRIVER, si abilita il motore e si accendono i LED seguenti:

- DRIVER ENABLE: esso si accende quando il motore è abilitato per la ricezione dei comandi di stato e di movimentazione
- DRIVER READY: esso si accende quando il motore è pronto (in stato operativo) per la ricezione dei comandi di stato e di movimentazione
- FW ENABLE: il motore è abilitato per la movimentazione in avanti (positiva)
- BW ENABLED: il motore è abilitato per la movimentazione all'indietro (negativa)

Successivamente si invia il comando di Homing premendo l'interruttore GO HOME. Tale procedura porterà lo stage in prossimità dello switch di fine corsa hs2 e inizializzerà la posizione che verrà visualizzata nella casella di testo ACTUAL POSITION, inoltre si accenderà il LED CALIBRATED per indicare che la procedura di homing è stata effettuata con successo e il LED IN TARGET POSITION per indicare che il movimento è andato a buon fine. Le caselle ACTUAL POSITION e ACTUAL VELOCITY ci forniranno, in tempo reale, la posizione e la velocità attuale.

Per quanto riguarda la parte relativa alle movimentazioni, sono presenti tre tipi di movimenti:

- Movimento assoluto
- Movimento relativo
- Movimento Jog

Per impostare un movimento assoluto, si digita sulla casella Target Position la posizione finale, sulla casella Target Velocity la velocità e infine si attiva il comando premendo l'interruttore Move_ABS.

In modo analogo si procede per un movimento relativo.

Inoltre sono presenti 4 interruttori per l'esecuzione dei movimenti jog.

Tra i vantaggi della creazione di una interfaccia grafica vi è la possibilità di poter personalizzare tale interfaccia e, di conseguenza, anche "il dialogo" tra l'amministratore e il singolo sottosistema, aggiungendo le funzioni che sono di interesse o che potrebbero rivelarsi utili per il monitoraggio e il collaudo (testing) del sistema in esame. A tal proposito è stato aggiunto l'interruttore WRITE_PARAMETER che consente di poter settare, la velocità negativa di calibrazione verso hs1.

Infine sono presenti tre LED: STATUS_ERROR, JOG_ERROR, ABS_ERROR e tre caselle di testo corrispondenti. I LED indicano la presenza di un errore sullo stato o sui movimenti assoluto e jog mentre le tre caselle di testo indicano, rispettivamente, il codice di errore sullo stato del motore e sui movimenti assoluto e jog.

3.1.2.9 Sviluppo del software in linguaggio Java

Nell'ambiente di sviluppo Eclipse è possibile implementare vari algoritmi per la realizzazione di un Client in linguaggio Java, con lo scopo di effettuare dei test sul Server TwinCAT. Inoltre, facendo uso delle librerie *javax.swing*, è possibile realizzare un'interfaccia grafica "user-friendly" in modo da poter inviare più facilmente i comandi al Server. Poiché l'interfaccia ingegneristica realizzata in ambiente TwinCAT 3 si può utilizzare solo in tale ambiente, per avere la possibilità di poter controllare il sistema anche da un PC remoto e al di fuori dell'ambiente TwinCAT 3 nasce la necessità di implementare un'interfaccia mediante un linguaggio ad alto livello come il Java. Tale interfaccia non è di tipo ingegneristica ma è più sintetica ed è stata ideata per consentire all'utente di interagire con il sistema mediante funzioni ad alto livello le quali possano soddisfare gli interessi e gli studi scientifici e applicativi dell'utente come utilizzatore.

Per poter realizzare gli algoritmi che consentono la comunicazione con il Server, si fa uso delle librerie ADS (DLL) messe a disposizione della ditta Beckhoff, per il linguaggio Java.

L'interfaccia è la seguente:

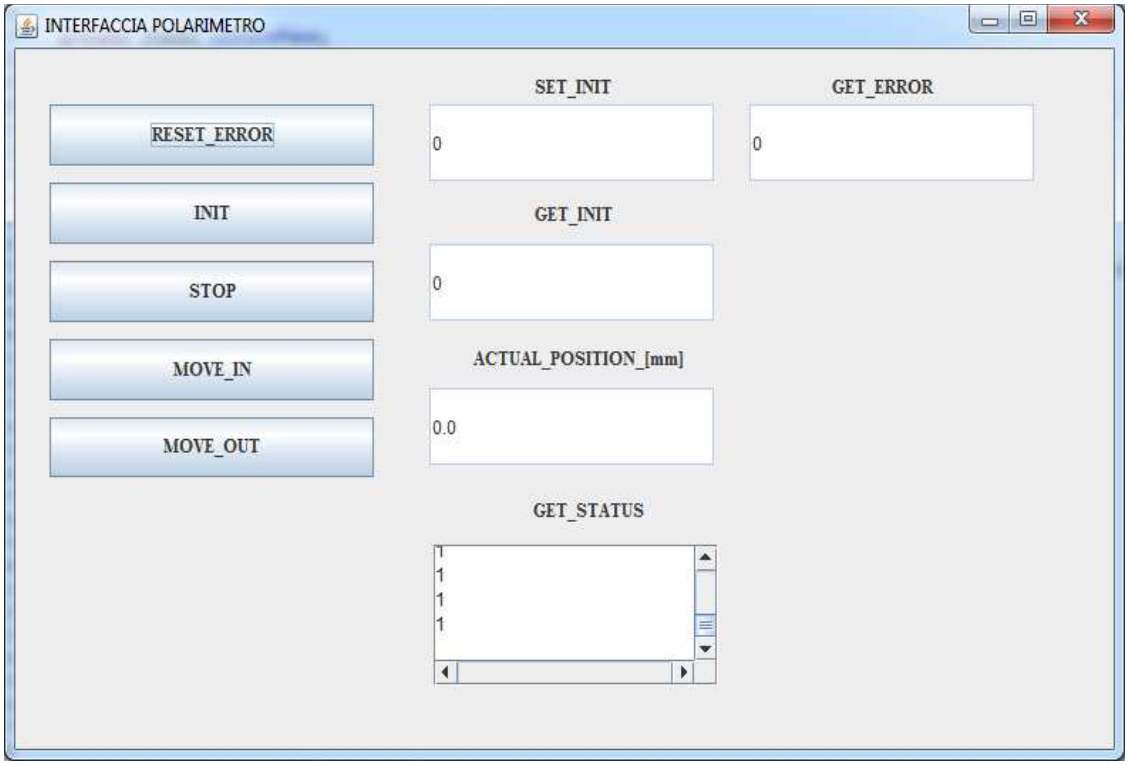

Figura 3.18: Interfaccia Client per il sottosistema polarimetrico

Come si evince da tale interfaccia, sono state implementate le seguenti funzioni:

- RESET_ERROR

Tale funzione è necessaria quando si verificano degli errori durante l'homing o durante il funzionamento del motore, a causa dei quali il motore stesso va nello stato di "stallo". Per poter sbloccare il motore occorre inviare tale comando.

- INIT

Questa funzione ingloba vari comandi. Il primo comando che viene inviato al Server TwinCAT è il comando Enable che consente di abilitare il motore. Successivamente viene inviato il comando di Homing.

- STOP

Questa funzione invia il comando di arresto al motore.

- MOVE_IN

Questa funzione invia al motore un comando per effettuare un movimento assoluto, a partire dalla posizione di zero impostata dall'Homing, verso la posizione di 40 mm che in prima approssimazione dovrebbe essere la posizione di intercettazione del fascio luminoso.

- MOVE_OUT

Questa funzione invia al motore un comando per effettuare un movimento assoluto verso la posizione di 0 mm, in modo tale che il polarimetro non intercetti il fascio luminoso.

Inoltre sono presenti delle caselle di testo che ci informano sulle variabili impostate nel Server e viste in precedenza; tra queste è presente l'ACTUAL POSITION che restituisce la posizione corrente del motore.

3.2 Il sottosistema di autoguida

Questo sottosistema è composto da due stepper motor che azionano, rispettivamente, due assi di movimentazione. Il controllo del motore, il cui supporto presenta connettore RS-232, viene effettuato da PC embedded CX5020 di Beckhoff in combinazione con il modulo Beckhoff EL6002. In seguito sarà illustrato come configurare il PC CX5020 in modo da poter controllare un dispositivo connesso al bus EtherCAT.

3.2.1 Requisiti

Lo specchietto di tip-tilt è stato pensato per correggere gli errori di inseguimento dovuti alle caratteristiche meccaniche del telescopio. La precisione richiesta per il riposizionamento dell'oggetto da osservare nella posizione di riferimento è, considerati i valori medi del seeing (2-3 secondi d'arco) e le dimensioni proiettate in cielo dalla fibra ottica (< 5 secondi d'arco), di 0.5 secondi d'arco (equivalenti a 1.39×10^{-4} gradi o 2.4 μrad)[1].

La risoluzione del motore dell'asse azimuth (tabella 2.4 paragrafo 2.3.2.1) vale circa 2 μrad mentre quella dell'asse di elevazione è circa 1 μrad (tabella 2.5 paragrafo 2.3.2.1) e combinate assieme (somma quadratica) ci danno una risoluzione di 2.25 μrad paragonabile quindi al requisito di riposizionamento richiesto.

Da un punto di vista della funzionalità delle movimentazioni, per quanto descritto nei capitoli precedenti, tale dispositivo dovrà effettuare dei piccoli movimenti, in modo continuo, pertanto sono necessari sia la funzione di movimento assoluto che quella di movimento relativo.

[1] 1 radiante = 57° = 206265" (secondi d'arco) → 1"≅ 4.848E-6 radianti ≅ 2.76E-4 gradi

3.2.2 Calcolo della posizione angolare del dispositivo T-OMG

Il T-OMG è costituito da due attuatori e due controller che sono collegati in cascata all'interno dell'alloggiamento di controllo [29]. Per riferirci a ogni asse in modo indipendente, occorre attenerci al protocollo di comunicazione Zaber Binary. I due assi sono l'Azimuth e l'Elevazione. L'attuatore azimutale muove il supporto ottico da un lato all'alto cioè a destra e a sinistra attorno all'asse verticale (movimento tip*)*, mentre l'attuatore di elevazione inclina il supporto ottico su e giù attorno ad un asse orizzontale (movimento tilt*)*. Gli assi di elevazione e azimutale si intersecano a 90 gradi al centro del montaggio ottico.

Riportiamo alcune utili espressioni per consentire il calcolo dei dati in microstep e l'angolo degli assi del mount, ricavate da [28]. Si considerino le due seguenti formule. La prima (3.20) indica la posizione espressa in microstep sulla base della posizione angolare nella quale si trova l'attuatore. Tale formula indica una corrispondenza tra posizione angolare e posizione in microstep. Essa viene utilizzata per ricavare la posizione in microstep che deve raggiungere il motore. Si fornisce in ingresso l'angolo θ, theta, che è la posizione angolare desiderata, in genere espressa in gradi, e la formula restituisce i microsteps corrispondenti:

$$data = \tan(\theta) * A * \frac{R}{L} \tag{3.20}$$

La seconda formula (3.21) indica la dimensione angolare di un microstep sulla base della posizione nella quale si trova il motore rispetto a zero gradi. Da tale relazione si deduce che la dimensione angolare di un microstep non è costante ma varia al variare della posizione corrente nella quale si trova il singolo attuatore per cui l'angolo di cui il motore deve muoversi non è costante ma varia a seconda della posizione angolare in cui attualmente il motore si trova, rispetto alla posizione di 0°

Pertanto mediante tale formula, sulla base della posizione in microstep in cui si trova l'attuatore rispetto alla posizione assoluta di 0, è possibile sapere quanto è grande la dimensione angolare di un microstep, in radianti. In ingresso si fornisce il valore in microstep corrispondente alla posizione angolare nella quale si trova il motore e la formula ci

restituisce la dimensione angolare di un microstep in radianti, che si può indicare con ψ. La formula è la seguente:

$$\psi = \frac{L/R/A}{(data * (L/R)/A)^2 + 1} \ [radianti] \tag{3.21}$$

La nomenclatura è la seguente:

- $data$ è il numero di microstep rispetto alla posizione assoluta di 0; $data$ può essere positivo o negativo.
- θ è l'angolo di rotazione (movimento) rispetto allo 0. La maggior parte dei software calcola l'angolo in radianti.
  - $x\ [radianti] = x * 180/\pi\ [gradi]$ (3.22)
  - $y\ [gradi] = y * \pi/180\ [radianti]$ (3.23)
- *A* è la distanza tra l'attuatore e l'asse di rotazione, in particolare si ottiene [28]:
  - $A_{Azimuth}$ = 11825 μm
  - $A_{Elevation}$ = 23650 μm
- *R* (microstep/step) è la risoluzione del microstep, indica quanti microstep sono presenti dentro uno step e di default è pari a 64 microstep/step
- *L* è il movimento lineare per un fullstep dell'attuatore, per cui indica la dimensione lineare di uno step espressa in micron. Per entrambi gli assi, esso è 1.524 μm/step.
- ψ e la dimensione angolare di 1 microstep in radianti in base alla posizione angolare dell' attuatore

In base alle considerazioni precedenti, date due posizioni angolari A=angolo iniziale e B=angolo finale, è possibile descrivere i metodi mediante i quali sapere di quanti microstep si deve muovere l'attuatore per andare da A verso B.

- **Metodo A**: data la posizione angolare A, si calcola la posizione equivalente in microstep mediante la formula (3.20) e si ottiene la posizione equivalente in microstep A'. Data la data la posizione angolare B, si calcola la posizione equivalente in microstep mediante la formula (3.20) e si ottiene la posizione equivalente in microstep B'. Si cal-

cola la differenza tra le due posizioni in microstep A'- B'= C', così il motore si dovrà muovere di C' microsteps.

Esempio:

A=0°, B=0.02° → calcolo A'=0 microstep, calcolo B'= 173 microsteps, per cui si ha C'=A'-B'= 173 microsteps.

- **Metodo B**: data la posizione angolare iniziale A e quella finale B e indicando con γ la dimensione angolare di un microstep, si consideri la differenza angolare δ = B-A e si calcoli il numero di microstep di cui deve muoversi l'attuatore mediante il seguente rapporto:

$$microstep = \frac{\delta}{\gamma} \tag{3.24}$$

In virtù della formula (3.21), calcolando il numero di microstep dalla relazione (3.24), si commette un errore. La legge con cui varia la dimensione angolare di un microstep in gradi si ottiene facendo il grafico della relazione matematica tra la posizione in microstep e la dimensione angolare di un microstep corrispondente, sia per l'attuatore azimutale che per l'attuatore di elevazione. I grafici ottenuti sono i seguenti (figura 3.19 e figura 3.20):

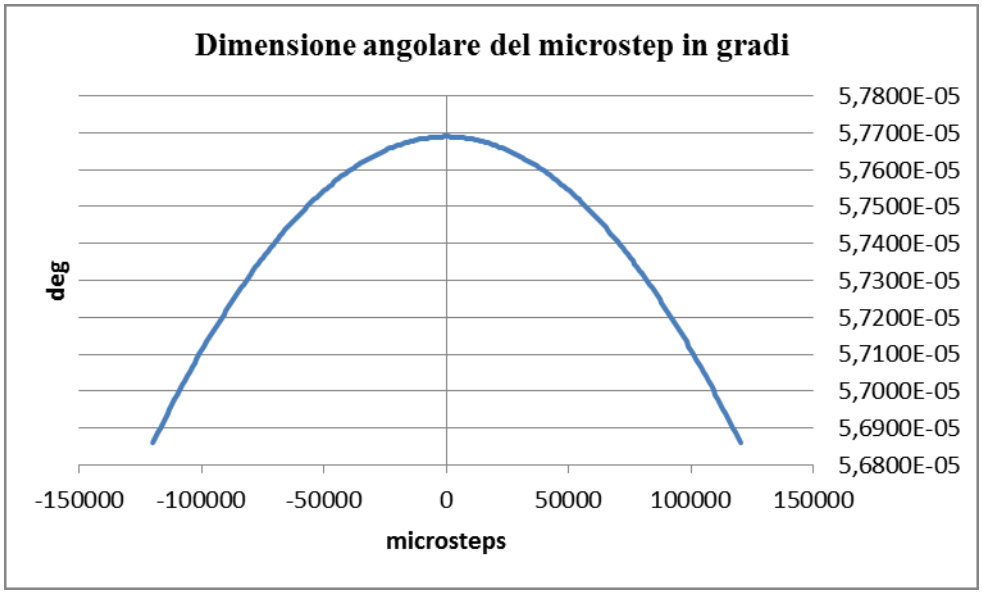

Figura 3.19: Dimensione angolare del microstep in gradi per l'attuatore di elevazione

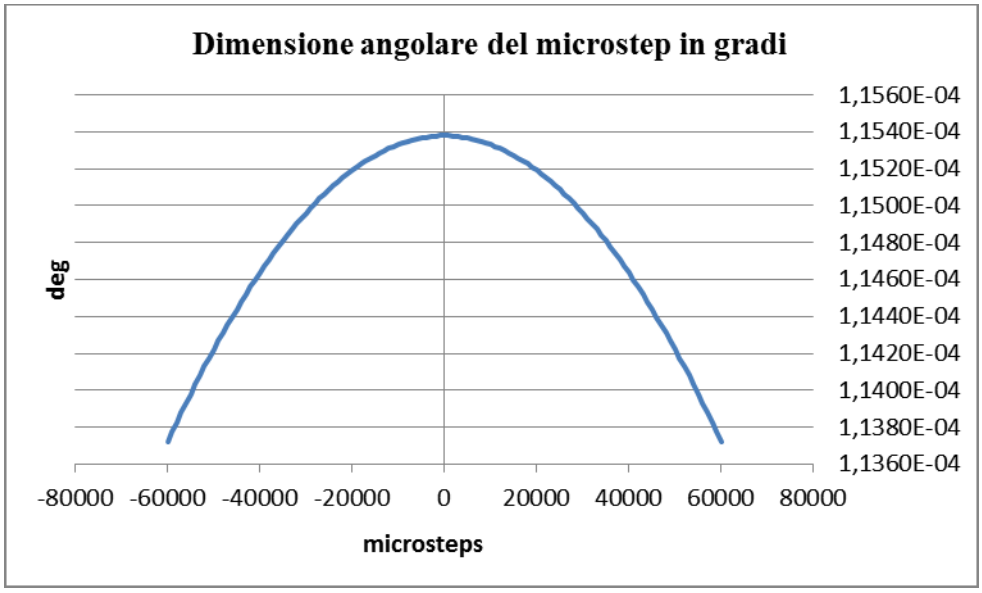

Figura 3.20: Dimensione angolare del microstep in gradi per l'attuatore azimutale

Si evince che l'andamento per entrambi gli attuatori è di tipo parabolico con il massimo valore nella posizione di 0 microstep e un valore minimo agli estremi, pertanto nel seguire il Metodo B si commetterebbe un errore, dovuto alla variazione della dimensione angolare di un microstep che varia da un valore massimo se siamo nella posizione 0 ad un valore minimo agli estremi. Più precisamente, esprimendo i valori in radianti, si ottiene:

AZIMUTH		ELEVATION	
Posizione (microstep)	Valore (radianti)	Posizione (microstep)	Valore (radianti)
0	1.1538E-04	0	5.7689E-05
$\pm$60.000	1.1373E-04	$\pm$120.000	5.7480E-05

Tabella 3.2: Variazione angolare di un microstep agli estremi del dispositivo T-OMG – Zaber

- **Metodo C**: data la posizione angolare iniziale A e quella finale B, si sceglie un numero di divisioni N con cui suddividere l'intervallo angolare δ = B-A in modo da calcolare, per ogni singolo sub-intervallo, il numero di microstep di quanto il motore deve muoversi. Poi si sommano i microstep per ogni sub-intervallo e si ricava il numero totale di microsteps. Con tale metodo più complesso del precedente, si commette sempre un errore che dipende dal numero di sub-intervalli con cui si ha suddiviso la dimensio-

ne angolare da percorrere, ma tale errore è minore per aver suddiviso l'intervallo angolare di movimento in sub-intervalli, pertanto per evitare tale errore utilizzeremo il metodo A. Tale metodo è utilizzabile grazie alla formula (3.20) senza la quale si sarebbe stati costretti ad utilizzare uno degli altri due metodi. Mentre per i movimenti assoluti è possibile tradurre i gradi direttamente in bytes, per i movimenti relativi in gradi, il metodo A e quindi la formula (3.20) è indispensabile infatti occorre tradurre tutti i dati in microstep.

Un'altra formula utile è l'inversa della (3.20) la quale consente di passare da microstep a radianti [28]:

$$\theta = \frac{\arctan(data) * L}{A * R} \quad [radianti] \tag{3.25}$$

3.2.3 Controllo attraverso la porta RS-232

Tutti i dispositivi serie T utilizzano lo stesso protocollo di comunicazione RS232. Le impostazioni di comunicazione devono essere [28]:

- 9600 baud,
- no handshaking,
- 8 bit,
- no parity,
- 1 bit stop.

Tutte le istruzioni sono costituiti da un gruppo di 6 byte. Esse devono essere trasmesse ad un intervallo temporale non superiore a 10 ms, per cui l'intervallo temporale tra ogni byte non deve superare i 10 ms. Se il dispositivo ha ricevuto meno di 6 byte ed è trascorso un periodo più lungo di 10 ms, il dispositivo ignora i byte che ha già ricevuto.

Il seguente elenco mostra il formato delle istruzioni:

- Byte 1 – Dispositivo o attuatore # (# sta per Numero).
- Byte 2 – Comando #

- Byte 3 – Data – Byte meno significativo (LSB)
- Byte 4 – Data
- Byte 5 – Data
- Byte 6 – Data – Byte più significativo (MSB)

Il primo byte è il numero del dispositivo o attuatore. Se si hanno più dispositivi collegati in serie, il dispositivo numero 1 è il dispositivo più vicino al computer, il dispositivo numero 2 è il successivo e così via. Il numero 0 è utilizzato per tutti i dispositivi che eseguiranno il comando contemporaneamente. Nel nostro caso è presente un solo dispositivo, per cui, di default, il numero 1 sarà riferito all'attuatore azimutale, il numero 2 all'attuatore di elevazione, infine il numero 0 ad entrambi.

Il secondo byte è il numero del comando. I byte 3,4,5,6 sono i dati di tipo long integer (intero lungo) nel formato "complemento a 2" con il byte meno significativo trasmesso per primo.

Si considerino alcuni esempi di comandi:

- Homing di tutti i dispositivi: (0,1,0,0,0,0)
- Versione firmware di tutti i dispositivi: (0,51,0,0,0,0)
- Il dispositivo 1 si muove verso una posizione assoluta di 257 microsteps (comando 20): (1,20,1,1,0,0)
- Il dispositivo 2 si muove di una posizione relativa di -1 microstep (comando 21): (2,21,255,255,255,255)

Per la maggior parte delle istruzioni, il dispositivo risponde con un codice di ritorno. Viene restituito un gruppo di 6 byte. Il primo byte è il numero del dispositivo. Il byte 2 è l'istruzione appena completata o 255 (0xFF) se si verifica un errore. I byte 3, 4, 5 e 6 sono byte di dati che si presentano nello stesso formato dei dati del comando inviato.

3.2.3.1 Relazioni per la conversione dei dati e calcolo delle velocità

Per poter controllare il dispositivo T-OMG, è necessario generare i 6 byte, in particolare i byte 3,4,5,6 da un singolo valore e viceversa e successivamente implementare una

serie di algoritmi per poter inviare tali dati al dispositivo. Si descriverà come procedere per effettuare entrambe le conversioni.

3.2.3.1.1 Conversione dei dati di comando in byte di comando da inviare ai dispositivi Zaber

Se i dati di comando sono negativi (Cmd_Data <0), allora si ottiene [28]:

$$Cmd\ Data = 256^4 + Cmd\ Data \tag{3.26}$$

$$Cmd\ Byte\ 6 = \frac{Cmd\ Data}{256^3} \tag{3.27}$$

$$Cmd\ Data = Cmd\ data - 256^3 * Cmd\ Byte\ 6 \tag{3.28}$$

$$Cmd\ Byte\ 5 = \frac{Cmd\ Data}{256^2} \tag{3.29}$$

$$Cmd\ Data = Cmd\ Data - 256^2 * Cmd\ Byte\ 5 \tag{3.30}$$

$$Cmd\ Byte\ 4 = \frac{Cmd\ Data}{256} \tag{3.31}$$

$$Cmd\ Data = Cmd\ Data - 256 * Cmd\ Byte\ 4 \tag{3.32}$$

$$Cmd\ Byte\ 3 = Cmd\ Data \tag{3.33}$$

In questo modo vengono ricavati i byte 3,4,5,6 del comando da inviare, dati dalle formule (3.33), (3.31), (3.29), (3.27), rispettivamente.

3.2.3.1.2 Conversione dei byte di risposta del dispositivo Zaber in un singolo valore

Dati i bytes di risposta ad un comando, la trasformazione di tali bytes in un singolo valore avviene mediante la seguente formula [28]:

$$Reply\ Data = 256^3 * Byte\ 6 + 256^2 * Byte\ 5 + 256 * Byte\ 4 + Byte\ 3 \quad (3.34)$$

Se il Byte 6 di risposta è minore di 127 (Rpl_Byte_6 > 127), allora si ottiene:

$$Reply\ Data = Reply\ Data - 256^4 \quad (3.35)$$

In questo modo viene ricavato il singolo valore della risposta a partire dai byte 3,4,5,6 di risposta.

Nella tabella 3.3 sono riportati alcuni comandi tra i più utilizzati:

Nome Istruzione	Numero Comando	Dati Comando	Tipo Comando	Dati di Risposta
Reset	0	Ignorati	Comando	Nessuno
Home	1	Ignorati	Comando	Posizione finale
Move Absolute	20	Posizione Assoluta	Comando	Posizione finale
Move Relative	21	Posizione Relativa	Comando	Posizione finale
Stop	23	Ignorati	Comando	Posizione finale
Set Home Speed	41	Velocità	Impostazione	Velocità
Set Target Speed	42	Velocità	Impostazione	Velocità
Set Current Position	45	Nuova Posizione	Impostazione	Nuova Posizione
Return Status	54	Ignorati	Impostazione di sola lettura	Stato
Return Current Position	60	Ignorati	Impostazione di sola lettura	Posizione
Error	255	n/a	Risposta	Codice Errore

Tabella 3.3: Comandi principali del dispositivo T-OMG – Zaber

3.2.3.1.3 Calcolo delle velocità

Si considerino i due comandi che ci consentono di impostare la velocità di Homing e la velocità di destinazione utilizzata per i movimenti

La velocità di homing può essere impostata in modo indipendente dalla velocità di destinazione mediante il comando 42.

Per quanto riguarda l'impostazione della velocità di destinazione, quando viene inviato il comando di un movimento assoluto o relativo, il dispositivo accelererà da una velocità iniziale, in genere 0 mm/s, fino alla velocità determinata da questo comando, in base al valore di accelerazione impostato. La velocità di destinazione può essere modificata "al volo" (on-the-fly), cioè anche quando il dispositivo sta eseguendo un movimento. Il dispositivo regolerà automaticamente la velocità, ma la posizione finale verso la quale si muoverà, sarà quella specificata nel primo comando.

Per il calcolo delle due velocità, se si vuole esprimere la velocità in mm/s oppure in °/s, occorre utilizzare la seguente relazione [28]:

$$Data * 9.375 * M = Speed\ required\ \ (Velocità\ desiderata) \tag{3.36}$$

Pertanto se, ad es. di desidera una velocità pari a 0.1°/s, si ottiene:

$$Data * 9.375 * M = 0.1 \tag{3.37}$$

dove:

- Data è il valore del dato del comando
- M (mm oppure °) è la dimensione del microstep a 0° o a 0 mm; esso dipende dall'attuatore, in particolare:
- M (per l'attuatore azimutale) = 1.1538E-4 mm o gradi → si può indicare tale costante con $M1$ (motore 1)
- M (per l'attuatore di elevazione) = 5.769E-4 mm o gradi → si può indicare tale costante con $M2$ (motore 2)

  per cui si ottiene, per il motore 1:

$$Data = \frac{0.1}{0.001082} \cong 92.44 \cong 92 \tag{3.38}$$

Pertanto affinché il motore 1 si muova a velocità pari a 0.1°/s, occorre che Data=92. Convertendo tale valore in byte di comando da inviare al dispositivo Zaber, si ha che i byte 3,4,5,6 sono rispettivamente: 92,0,0,0. Il comando da inviare per l'Home Speed è (1,41,92,0,0,0), mentre per il Target Speed si ha (1,42,92,0,0,0).
Per il motore 2, si ottiene:

$$Data = \frac{0.1}{0.00054084375} \cong 184.89 \cong 185 \tag{3.39}$$

Pertanto affinché il motore 2 si muova a velocità pari a 0.1°/s, occorre che Data=185. Convertendo tale valore in byte di comando da inviare al dispositivo Zaber, si ha che i Byte 3,4,5,6 sono rispettivamente: (185,0,0,0). Il comando da inviare per l'Home Speed è (2,41,185,0,0,0), mentre per il Target Speed si ha (2,42,185,0,0,0).

3.2.4 Comunicazione seriale con il modulo EL6002

3.2.4.1 Hardware supportato

Il terminale EtherCAT seriale viene fatto funzionare in modalità 22 byte, in modo che 22 byte di dati alla volta possono essere trasferiti o ricevuti dal terminale. Sono necessari per ogni scambio 3 cicli PLC. In questo caso vengono trasmessi 22 byte ogni 3 cicli, cioè per ogni scambio o pacchetto (il modulo EL6002 opera in 22-byte mode), pertanto se si desidera che il numero di byte trasferiti in ogni pacchetto sia pari a LB = 8 e si imposta una baudrate pari a bps=9600, si ricava il tempo di ciclo T necessario, dalla seguente formula [27]:

$$Bps = \frac{LB * \frac{22}{3}}{T} \tag{3.40}$$

dove si ottiene:

- LB = numero di byte di dati, pari a 8
- T = tempo di ciclo del PLC

Il tempo di ciclo T sarà pari a:

$$9600 = \frac{8 * \frac{22}{3}}{T} \tag{3.41}$$

$$T \cong 0{,}00604\ s \cong 6\ ms \tag{3.42}$$

3.2.4.2 Principio di comunicazione

Come descritto nella sezione 3.2.4.1, la velocità di trasferimento dei dati massima effettiva dipende in parte dal tempo di ciclo del PLC. Così, per esempio, per la comunicazione con il terminale di bus seriale (EL6002) ad un tasso effettivo di 9600 Bps, è richiesto un tempo di ciclo di 6 ms. In molte applicazioni dove è richiesto uno scambio di dati di grandi dimensioni, un tale tempo di ciclo così breve per tutte le operazioni del PLC caricherebbe fortemente lo stesso PLC.

Per processo (task) si intende l'attività di esecuzione di un programma in modo sequenziale, ovvero un compito che il processore dell'elaboratore deve portare a termine su richiesta dell'utente. Più precisamente è un'attività controllata da un programma che si svolge su un processore in genere sotto la gestione o supervisione del rispettivo sistema operativo. In questo contesto un task è l'elemento che determina lo scheduling temporale delle istanze dei programmi associati al task stesso.

Poiché per molte applicazioni tempi di ciclo più lunghi, per esempio, 10 ms sono più che sufficienti, è possibile, con l'aiuto della libreria *COMlibV2*, disaccoppiare il traffico dati tra il PLC e l'hardware (il modulo EL6002) dal restante traffico dati che riguarda le applicazioni PLC. Infatti poiché il PLC oltre all'invio e ricezione dei dati con il dispositivo T-OMG esegue anche tutta una serie di operazioni come controllare la sintassi dei comandi, verificare se il motore ha raggiunto o meno una determinata posizione, ecc. per trattare tali dati con priorità differente rispetto ai dati di Tx e Rx con il modulo EL6002, e quindi con il dispositivo T-OMG, la Beckhoff ci fornisce la possibilità di definire un software di Background collegato ad un task più veloce in cui i cicli vengono eseguiti o ripetuti a periodi più brevi che è il FB *SerialLineControl*, si veda la figura 3.21, il quale ha l'unico compito (è un FB "dedicato") di gestire il traffico dati Tx verso il dispositivo hardware e Rx dall'hardware (il modulo EL6002) verso il PLC, mentre le applicazioni PLC saranno collegate al task più lento. In questo modo vengono create nel programma PLC due task con differenti priorità. Il task standard viene eseguito con un tempo di ciclo del PLC più lungo, per esempio, 10 ms, mentre un secondo task di comunicazione viene eseguito con un tempo di ciclo più veloce, per esempio, 2 ms [27].

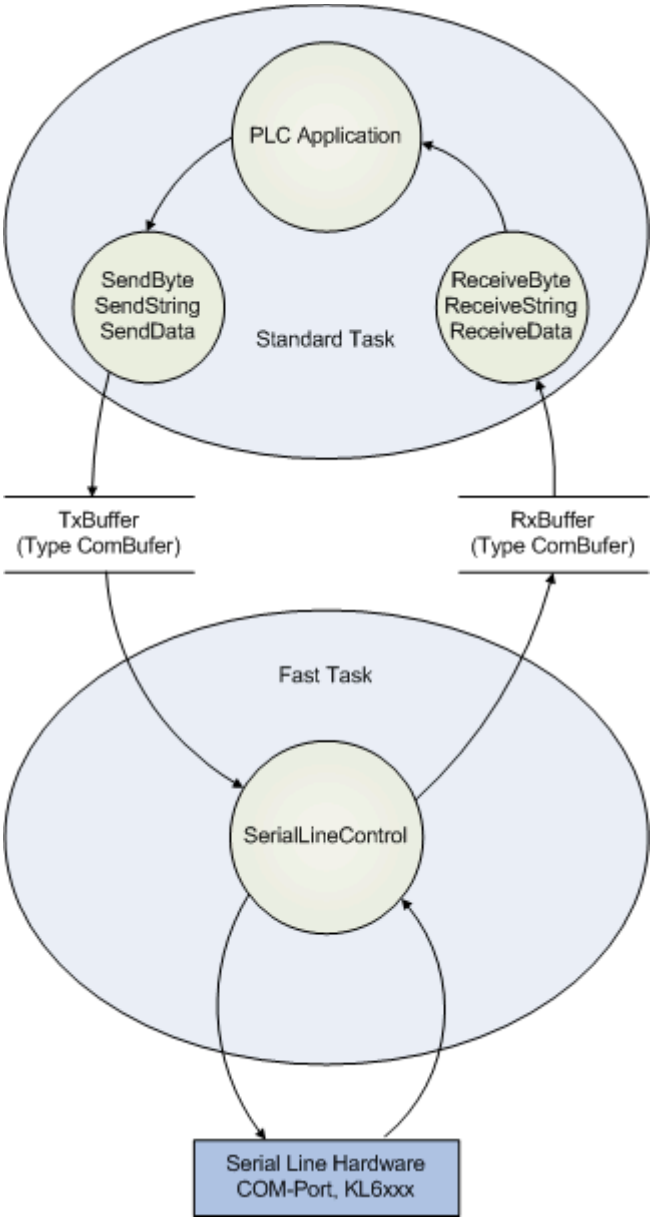

Figura 3.21: Principio di comunicazione su porte seriali

Si impostano i due task associando il task standard al Main principale di tutto il progetto, mentre il task fast sarà associato al PRG Background. In particolare, nel task fast, si modificano i seguenti valori:

- il valore di priorità pari a 19 rispetto al valore della priorità nel task standard pari a 20
- il valore della porta che, per il task fast è pari a 350, mentre per il task standard è pari a 351, si vedano le figure 3.22, 3.23.

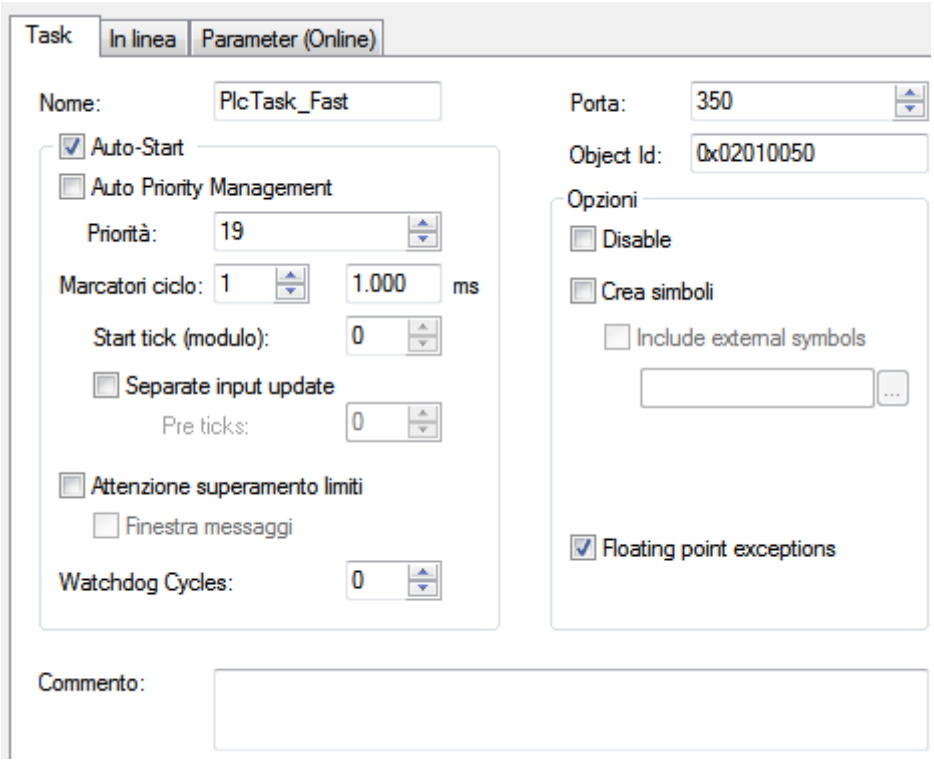

Figura 3.22: Impostazioni del task fast

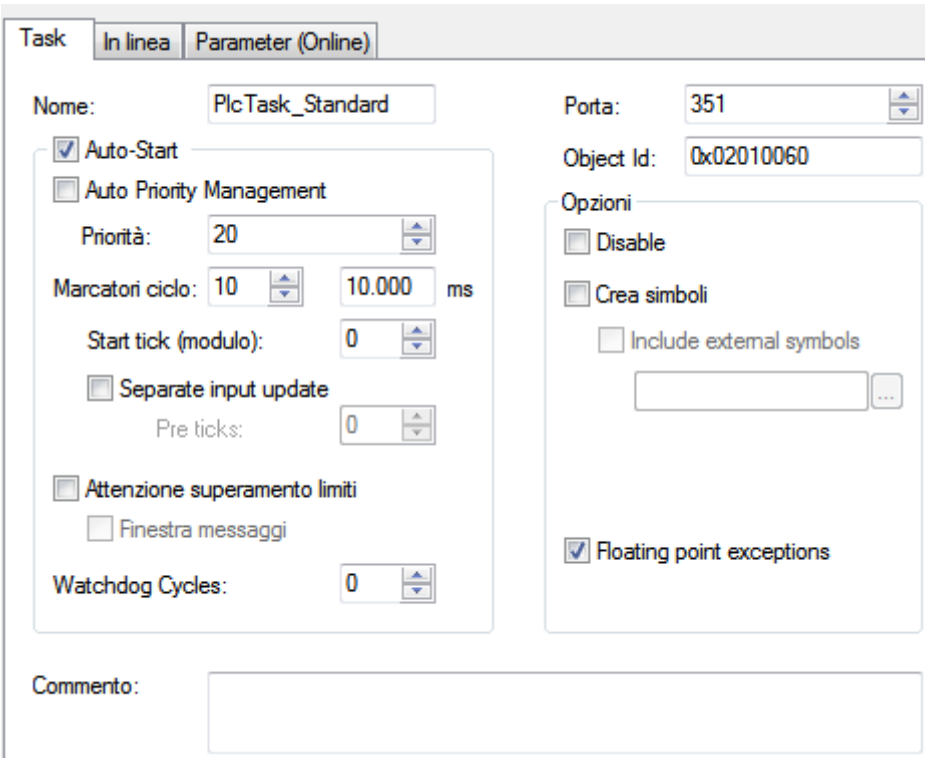

Figura 3.23: Impostazione del task standard

3.2.5 Sviluppo del software TwinCAT per il sottosistema di autoguida

Le potenzialità dell'ambiente di sviluppo TwinCAT 3 permettono di poter pilotare gli attuatori in esame, sia localmente, sia mediante un opportuno Client attraverso la suite TCP/IP. A tal proposito si ricordi che il TwinCAT 3 è di per sé un Server con il quale è possibile comunicare attraverso l'uso di vari protocolli tra cui l'ADS.

3.2.5.1 Configurazione del terminale

La prima operazione da fare è la configurazione del modulo EL6002, i cui parametri della comunicazione seriale devono coincidere con i dati di targa del motore T-OMG. A tal proposito si verificano le impostazioni della comunicazione seriale nella sezione "Com Port Setting"; qui si ottiene:

- 9600 bps
- No parity
- 8 bit
- 1 bit stop
- 864 Fifo Threshold (si lascia così di default)

3.2.5.2 Impostazione coerente dei dati

Il passo successivo è quello di impostare i dati; per un modulo EL6002 i dati vengono trasmessi a gruppi di 22 byte, per cui la funzione che si dovrà definire nel file Background deve essere la seguente: *fbEL6002ctrl*. Per accertarsi che tale FB sia coerente con il numero di byte utilizzati per la trasmissione, occorre valutare il parametro "Mode" di tale FB. Il suo valore è il seguente:

Mode:= SERIALLINEMODE_EL6_22B,

gli ultimi caratteri sono _22B, quindi tale FB utilizza per la trasmissione 22 Bytes.

3.2.5.3 Creazione dei collegamenti

Il passo seguente è la creazione dei collegamenti (links) tra le immagini di processo del PLC (variabili "software") e le immagini di processo dei terminali fisici (variabili "hardware").

Per quanto riguarda gli ingressi, i links devono essere creati tra le variabili di Input del PLC che sono presenti nella sezione *PlcTask Fast Inputs*, con le variabili di Input di immagini di processo del dispositivo seriale, presenti nella sottosezione *COM TxPDO-Map Inputs Channel 1* di Terminale 8 (si veda la figura 3.23):

Per quanto riguarda le uscite, i links devono essere creati tra le variabili di Output del PLC che sono presenti nella sezione *PlcTask Fast Outputs*, con le variabili di Output del dispositivo seriale, presenti nella sottosezione *COM TxPDO-Map Outputs Channel 1* di Terminale 8 (si veda la figura 3.24):

Si consideri in dettaglio tali links: in figura 3.24 e 3.25 sono riportate varie sezioni che compongono l'albero di configurazione del progetto (lo schema dei moduli hardware collegati e dei moduli software di TwinCAT 3), presente nell'ambiente TwinCAT 3. In particolare in fig. 3.24 si hanno a sinistra le immagini di processo di input della sezione PLC, relative al Task Fast mentre a destra sono raffigurate le corrispondenti immagini di processo di input della sezione Input/Output per il Terminale 8, relative al Task Fast. Analogamente in figura 3.25 si hanno a sinistra le immagini di processo di output della sezione PLC, relative al Task Fast mentre a destra sono raffigurate le corrispondenti immagini di processo di output della sezione Input/Output per il Terminale 8, relative al Task Fast.

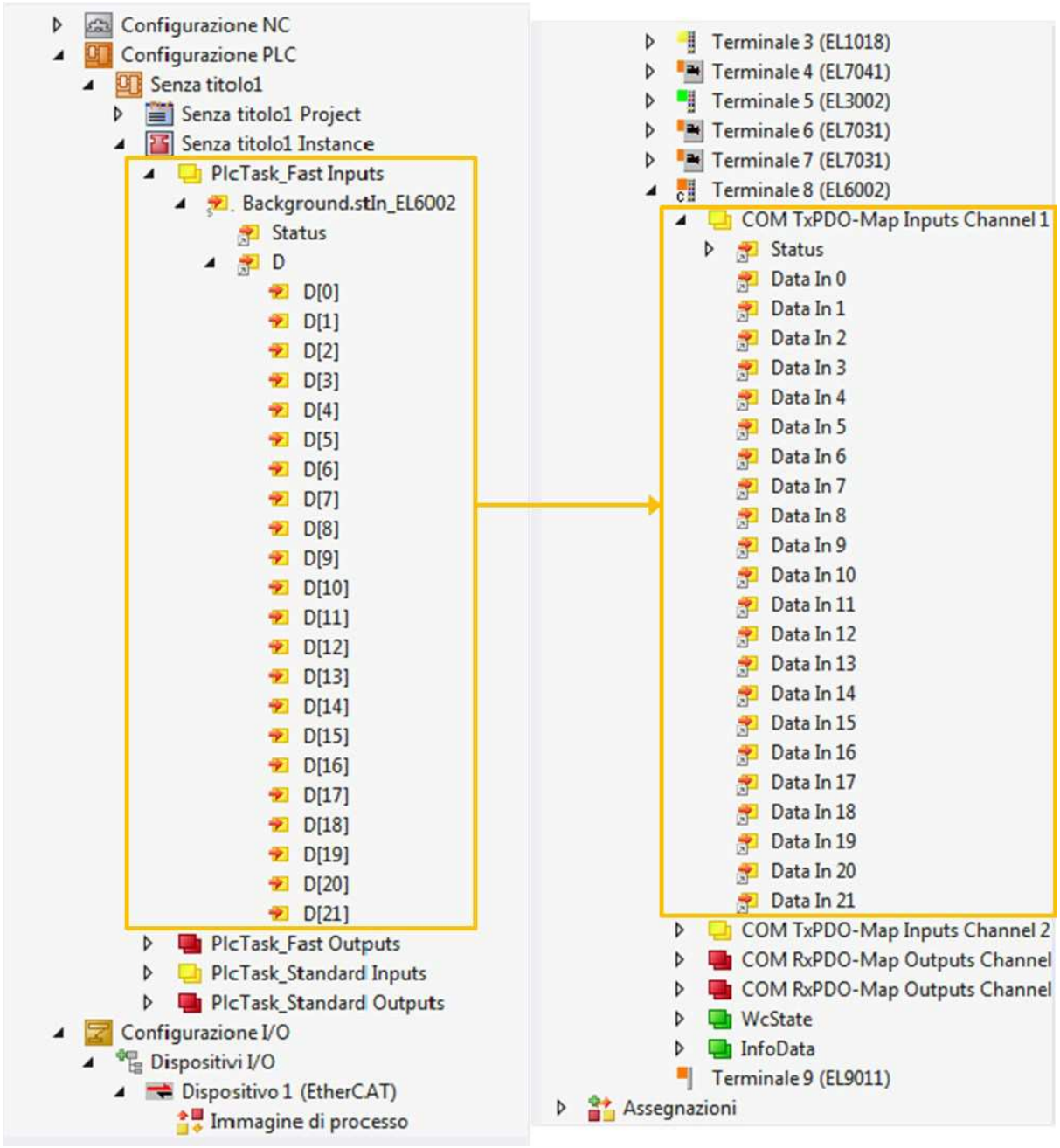

Figura 3.24: Linking input

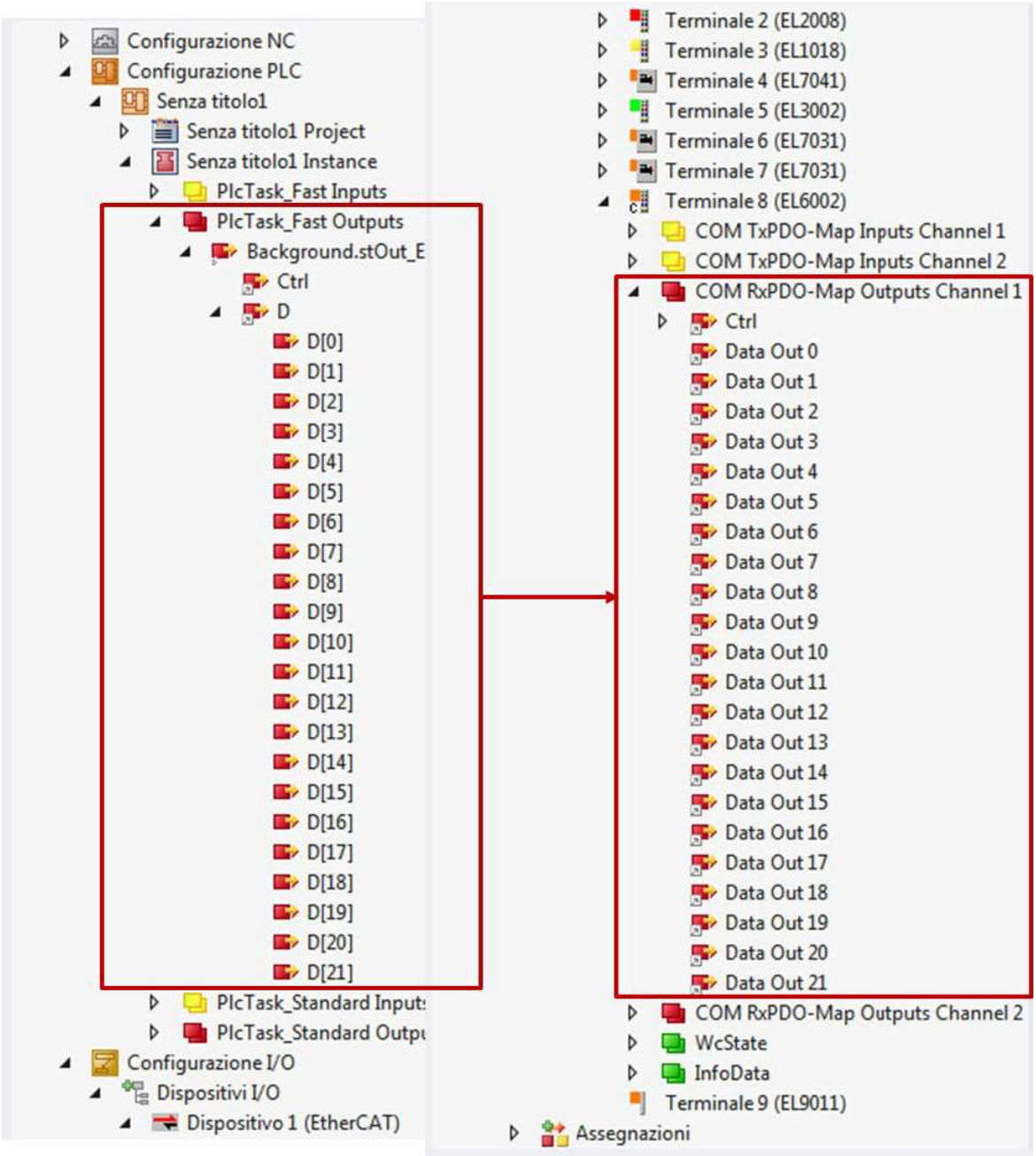

Figura 3.25: Linking output

3.2.5.4 Algoritmi

Le operazioni da effettuare sono le seguenti:

1) Creare i link**s**
2) Impostare le variabili di configurazione per la comunicazione seriale

3) Creare un insieme di algoritmi per la comunicazione tra il CX5020, il modulo EL6002 e il dispositivo tip-tilt.

Dopo aver creato i links, occorre definire all'interno della directory POUs, una cartella Autoguida che conterrà tutti i file inerenti alla comunicazione seriale con il dispositivo T-OMG. Successivamente si definiscono le variabili software globali Buffer di tipo *ComBuffer*, precisamente nel file GVL già creato in precedenza (per il sottosistema polarimetrico) presente nella directory GVLs. Tali variabili sono alla base della comunicazione seriale in quanto saranno utilizzate da tutti i file presenti nella cartella Autoguida. Il codice è il seguente:

```
VAR_GLOBAL

  RxBufferEL    : ComBuffer; (* Receive data buffer; used with all receive function blocks *)
  TxBufferEL    : ComBuffer; (* Transmit data buffer; used with all receive function blocks *)

END_VAR
```

Figura 3.26: Codice GVL per il dispositivo T-OMG

Per quanto esposto nel paragrafo 3.2.4.2, occorre creare due Main che sono regolati da Task a differente priorità.

Si consideri il main principale. Nella parte dichiarativa, si aggiunge l'istanza della FB fbEL6002 di tipo *FB_SerialCom* mediante la seguente istruzione:

```
PROGRAM MAIN
VAR
  fbEL6002      : FB_SerialCom;
END_VAR
```

Figura 3.27: Codice MAIN

mentre nella parte esecutiva del Main, si richiama l'istanza definita in precedenza, insieme alle altre, si veda la figura 3.28:

```
MASCHERA( ); // Terminale 7
SPECCHIETTO( ); // Terminale 6
POLARIMETRO( );
fbEL6002( TxBuffer:=TxBufferEL, RxBuffer:=RxBufferEL);
```

Lo schema generale di chiamate del main ai moduli suddetti è il seguente:

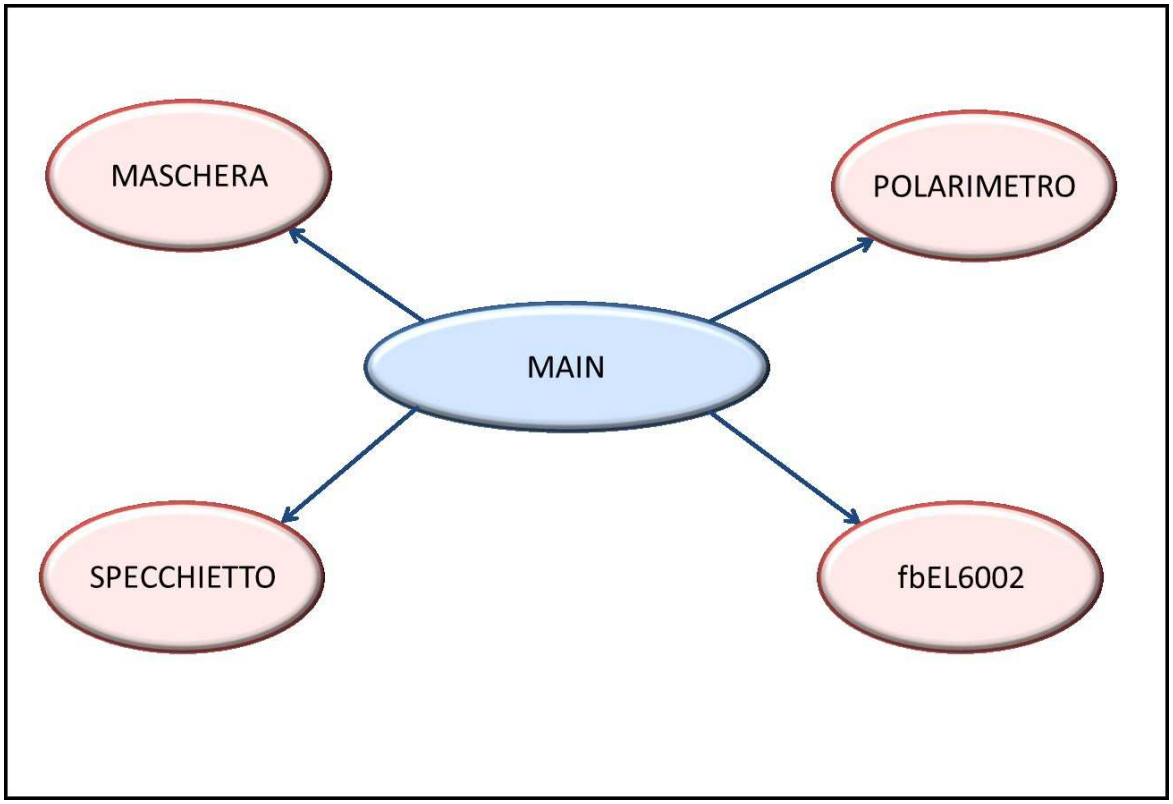

Figura 3.28 : POU richiamati dal MAIN

Poi, per la comunicazione seriale, si ottiene un altro main che è possibile definire Background, regolato dal task più veloce, nel quale si dichiarano le seguenti variabili necessarie per la comunicazione di background tra il PLC e il terminale EL6002:

- la FB fbEL6002Ctrl : SerialLineControl
- bEL6002CtrlError : BOOL;

- eEL6002CtrlErrorID : ComError_t;

Si dichiarano inoltre le variabili di Input/Output tra il PLC e il terminale EL6002:

- stIn_EL6002 AT %I* : EL6inData22B; (* linked to the EL6002 in the TwinCAT System Manager *)
- stOut_EL6002 AT %Q* : EL6outData22B; (* linked to the EL6002 in the TwinCAT System Manager *)

Infine si richiama la FB fbEL6002Ctrl , nella parte esecutiva:

```
(* background communication with the EL6001 terminal *)
fbEL6002Ctrl(
  Mode:= SERIALLINEMODE_EL6_22B,
  pComIn:= ADR(stIn_EL6002),
  pComOut:= ADR(stOut_EL6002),
  SizeComIn:= SIZEOF(stIn_EL6002),
  Error=> ,
  ErrorID=> ,
  TxBuffer:= TxBufferEL,
  RxBuffer:= RxBufferEL );
IF fbEL6002Ctrl.Error THEN
  bEL6002CtrlError := TRUE;
  eEL6002CtrlErrorID := fbEL6002Ctrl.ErrorID;
END_IF
```

Figura 3.29: Codice di chiamata dell'istanza fbEL6002Ctrl

Tra gli algoritmi realizzati, occorre soffermarsi su alcuni, in particolare si considerino i due algoritmi relativi alla trasmissione e alla ricezione dei dati. Per poter trasmettere e ricevere bytes su una porta seriale, non è possibile definire due funzioni in quanto una funzione (FUN) nasce e muore all'interno dello stesso ciclo del PLC e non ha memoria

dello stato precedente, invece una Function block (FB) una volta istanziata, viene eseguita per più cicli consecutivi mantenendo lo stato delle variabili da un ciclo al successivo. Poiché occorre tenere memoria delle variabili per poter leggere e scrivere sui buffer Rx e Tx, in quanto per poter inviare e ricevere uno stream di byte su una comunicazione seriale sono necessari più cicli, una funzione non è adatta per compiere tali operazioni per cui occorre definire due FB, le quali sono:

- send_byte
- receive_byte

Per lo stesso motivo, tutti gli algoritmi legati alla trasmissione di un comando e alla ricezione della risposta non possono essere definiti come funzioni (FUN) ma devono essere delle function block (FB).

Per quanto riguarda l'implementazione delle relazioni matematiche che consentono di convertire byte a partire da valori in gradi o in microstep e viceversa, occorre utilizzare delle funzioni in quanto esse sono adatte per calcoli matematici; le funzioni implementate sono le seguenti:

- get_speed
- set_speed
- traslate_byte_deg
- traslate_byte_ ustep
- traslate_ deg_ byte
- traslate_ deg_ ustep
- traslate _ustep_ byte
- traslate_ ustep_ deg

Un program (PGR) ha le stesse caratteristiche di una FB ma, a differenza delle FB che possono essere istanziate più volte, un program può essere istanziato una sola volta in tutto il progetto, pertanto per la comunicazione seriale, definiamo come program i due Main:

- Main
- Background (presente dentro la directory AUTOGUIDA)

mentre si definisce come function block (FB) il blocco funzione *FB_SerialCom* (presente dentro la directory AUTOGUIDA), istanziata nel Main principale, che ingloba tutte le FUN e FB del progetto AUTOGUIDA, rappresentando il "core" dello stesso.

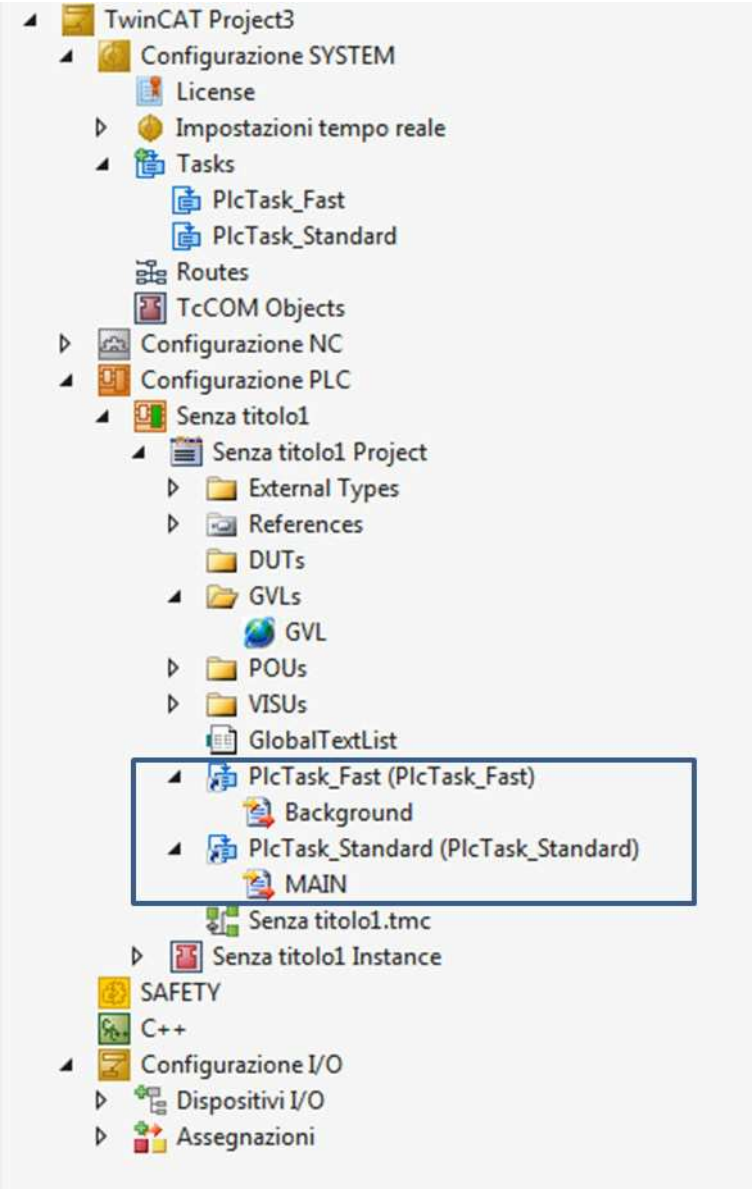

Figura 3.30: Assegnazione di Main e Background ai due task

Occorre notare che il program Main è richiamato e associato al Task_Standard, mentre il program Background è richiamato e associato al Task_Fast come si evince dalla figura 3.30.

Infine è stata implementata una FB dalla quale il Server TwinCAT restituisce tutte le informazioni di cui si ha bisogno. È stata definita tale FB come *Decode_RD* e il suo compito è quello di decodificare le risposte ricevute relative ai comandi inviati e riempire le variabili attese.

3.2.5.5 Macchina a stati

Per l'esigenza di dover sapere periodicamente lo stato e la posizione di ogni singolo attuatore, è necessario implementare un algoritmo mediante il quale si possa inviare un comando di movimento come l'homing e sapere la posizione del motore, non solo alla fine dell'esecuzione della procedura ma anche durante tutta la procedura di homing. Poiché per poter eseguire la FB Receive occorrono più cicli di PLC e accertato che i comandi di richiesta di stato e di posizione, inviati al dispositivo T-OMG, vengono gestiti dallo stesso in modo "non preemptive" (non sono di tipo preemptive), per poter ottenere un monitoraggio sullo stato e sulla posizione durante l'esecuzione di una qualsiasi procedura, occorre implementare un algoritmo in base al quale si possa inviare il comando di movimento, p es. l'homing, all'interno di un ciclo PLC, poi venga inviato il comando di richiesta sullo stato o sulla posizione del motore nel successivo ciclo PLC e infine attendere le risposte nei cicli seguenti. In pratica ciò si realizza implementando una macchina a stati finiti in base alla quale, in un singolo ciclo PLC viene inviata una richiesta o si riceve una risposta. Tale macchina a stati prevede la presenza di un'istruzione CASE con una sequenza che scandisce i cicli e quindi, le operazioni di invio e ricezione, in modo tale che le risposte vengano memorizzate nel buffer Rx. Precisamente si hanno 4 cicli PLC, si veda la figura 3.31:

0) in tale ciclo o valore di sequenza, sono presenti tutti i comandi di movimento, di homing e di invio di un generico comando;
10) in tale ciclo è presente la FB Receive (ricevo le risposte)
20) qui si ottiene la FB Status che invia la richiesta di stato al singolo motore o ad entrambi
30) infine si ottiene un'altra FB Receive (ricevo lo status)

Se la variabile Count (si veda il paragrafo 3.10) è diversa da zero, si rimane nello stato 30, altrimenti si torna nello stato 0.

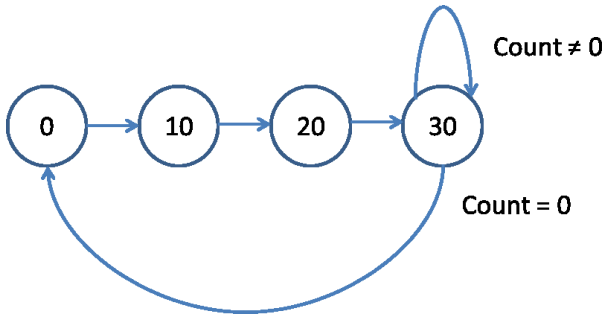

Figura 3.31: Macchina a stati finiti

3.2.6 Pannello di visualizzazione

Il pannello di visualizzazione del server TwinCAT 3 è il seguente:

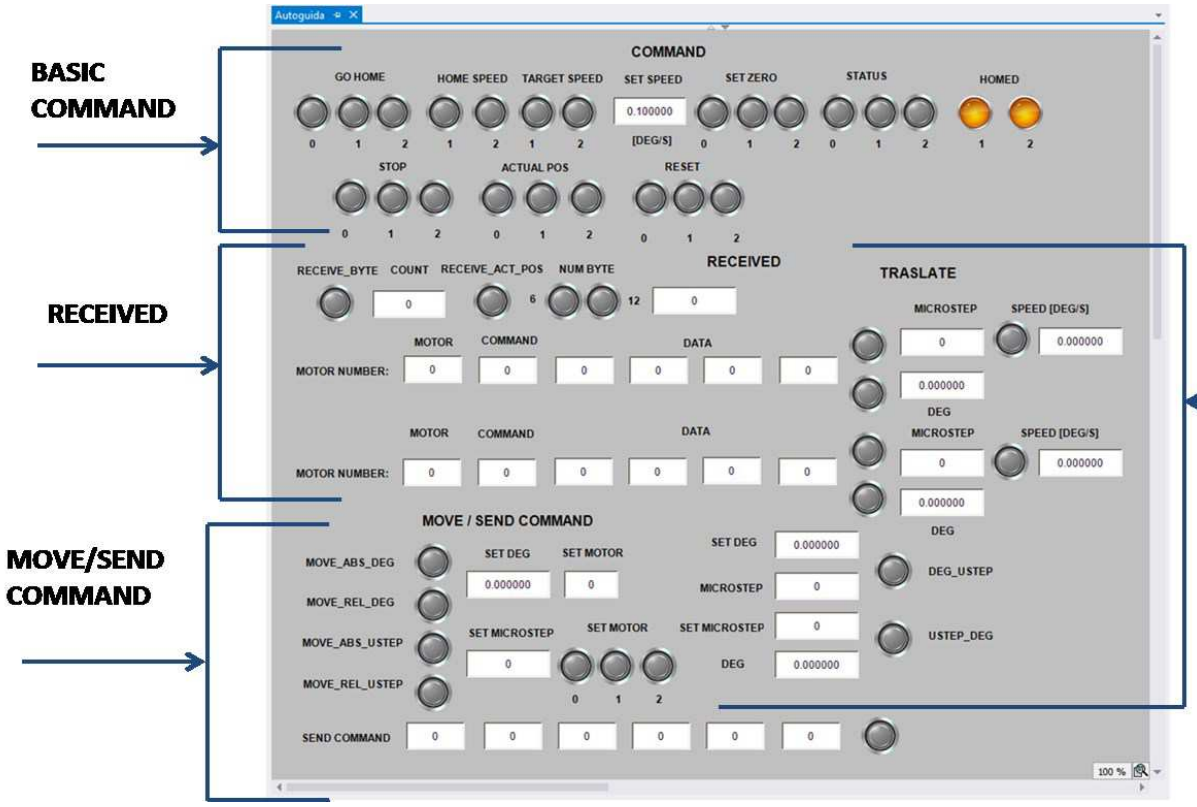

Figura 3.32: Pannello di visualizzazione TwinCAT 3 del sottosistema di autoguida

In questo pannello sono presenti 4 sezioni, come si vede in figura 3.33:

- BASIC COMMAND
- RECEIVED
- MOVE/SEND COMMAND
- TRASLATE

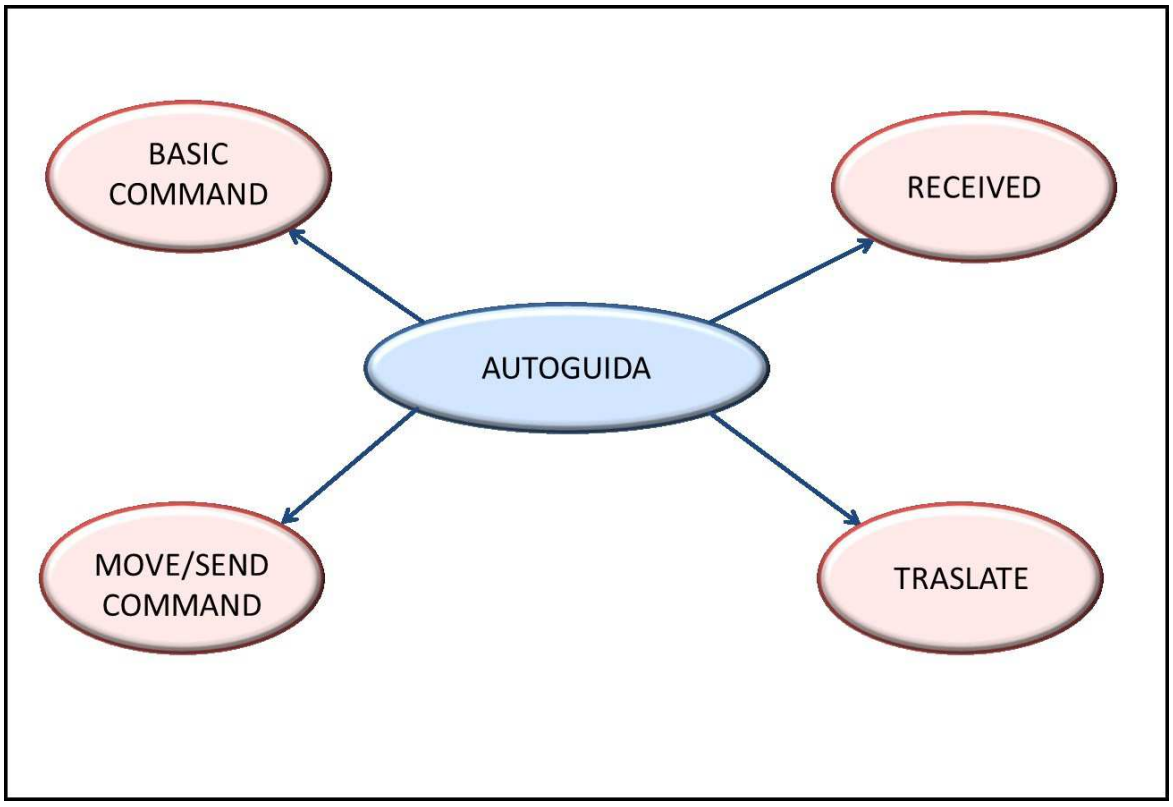

Figura 3.33: Sezioni dell'interfaccia ingegneristica del sottosistema di autoguida

Nella sezione BASIC COMMAND sono presenti dei tasti relativi all'invio dei comandi più importanti e utili per gli obiettivi dell'amministratore. Per ogni comando sono presenti tre tasti etichettati con i numeri 0,1,2 che si riferiscono al target del comando:

- 0 = ad entrambi i motori
- 1 = all'attuatore azimutale o motore 1
- 2 = all'attuatore di elevazione o motore 2

I tasti relativi ai comandi implementati sono i seguenti (figura 3.34):

- GO HOME = invia il comando di homing

- HOME SPEED = invia il comando per impostare la velocità di homing, leggendola dalla casella di testo SET SPEED
- TARGET SPEED = invia il comando per impostare la velocità dei movimenti (assoluto e relativo), leggendola dalla casella di testo SET SPEED
- SET ZERO = invia il comando di posizionamento a 0° e 0 microstep per entrambi i motori o per un singolo motore
- STATUS = invia la richiesta sullo stato a entrambi i motori, o ad ogni singolo motore
- STOP = invia il comando di arresto ad un qualsiasi movimento di entrambi i motori o di un singolo motore
- ACTUAL POS = invia il comando di richiesta della posizione attuale per entrambi i motori o per ogni singolo motore
- RESET = invia il comando di reset ad entrambi i motori o ad un singolo motore, in caso di errore.

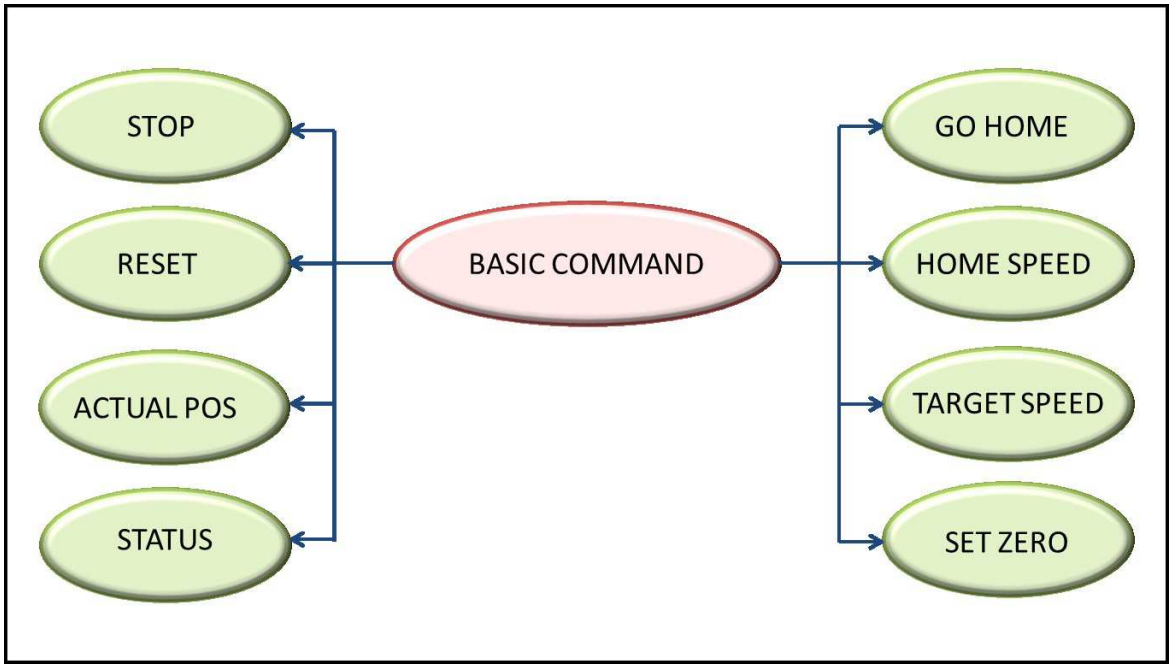

Figura 3.34: POUs della sezione Basic Command

Per quanto riguarda la sezione RECEIVED è presente un tasto che, quando premuto, visualizza le risposte memorizzate nel buffer Rx, nelle 12 caselle di testo relative ai due

motori. Per sapere se tale buffer è parzialmente riempito e quindi ci sono delle risposte (bytes) che non sono state ancora lette, è stata definita la variabile COUNT la quale restituisce tale informazione. Essa ritorna il numero di byte presenti nel buffer Rx e quindi da leggere. La funzione Receive è stata implementata in modo tale che, la risposta ricevuta e memorizzata nel buffer Rx, venga visualizzata sui due gruppi di caselle di testo etichettate "Motor Number" a seconda che il comando corrispondente sia stato inviato ad entrambi i motori oppure ad un solo motore. Nel primo caso ci si aspetta di ricevere 12 bytes di risposta che verranno visualizzati nelle 12 caselle di testo sopra menzionate, mentre nel secondo caso ci si aspetta di ricevere 6 bytes che verranno visualizzati nelle 6 caselle di testo relative al motore corrispondente. Infine è stata implementata una FB RECEIVE_ACT_POS la quale ad un comando di richiesta della posizione attuale inviato ad un motore o ad entrambi, legge i byte presenti sul buffer Rx e li posiziona nelle caselle di testo corrispondenti (figura 3.35).

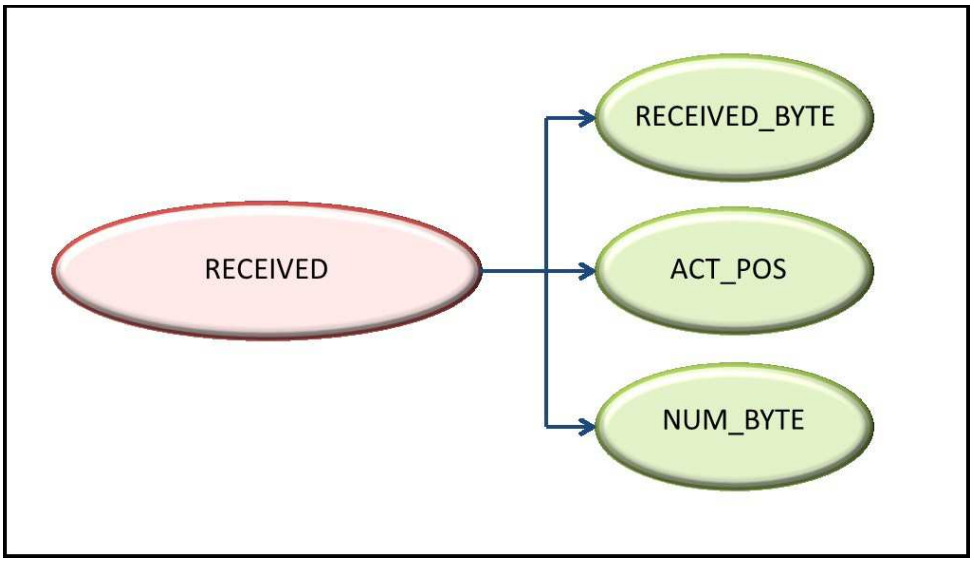

Figura 3.35: POUs della sezione Received

Poi si ha la sezione MOVE/SEND COMMAND. Qui sono presenti le 4 funzioni di movimento assoluto e relativo, precisamente due funzioni di movimento assoluto riferite alla posizione impostata, come dato di ingresso, in gradi e in microstep e altre due analoghe per il movimento relativo. In questa sezione è stata inclusa la possibilità di inviare un

generico comando, ad un qualsiasi motore mediante la FB SEND COMMAND (figura 3.36).

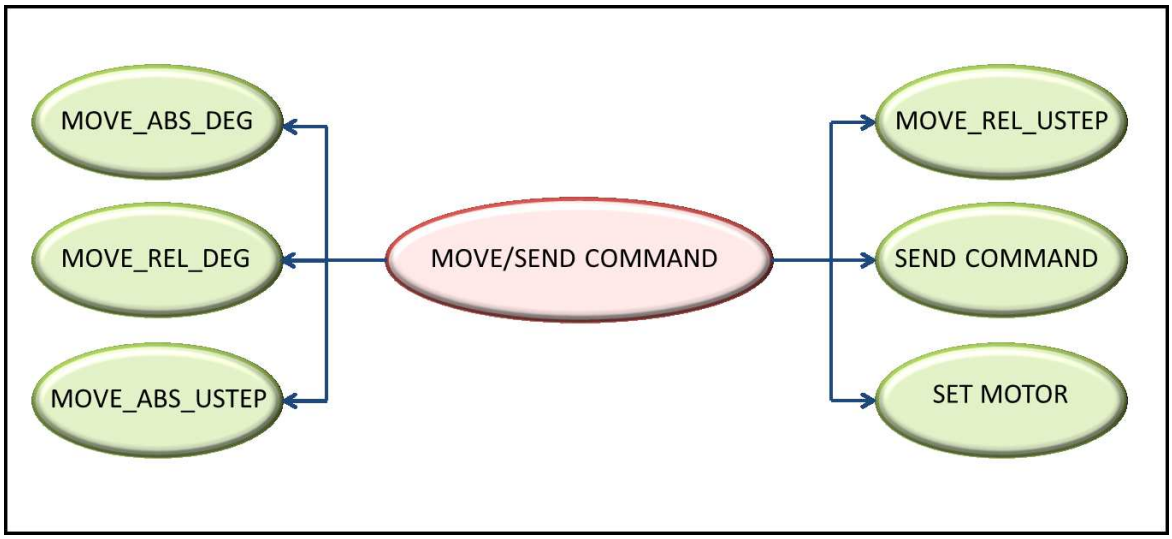

Figura 3.36: POUs della sezione Move/Send Command

Infine nella sezione TRASLATE sono presenti tutte quelle funzioni di supporto che ci consentono di poter tradurre i gradi in microstep e viceversa. Il pulsante SPEED restituisce la velocità di homing o di un movimento (a seconda della procedura fatta precedentemente) con cui si è mosso il singolo motore (figura 3.37).

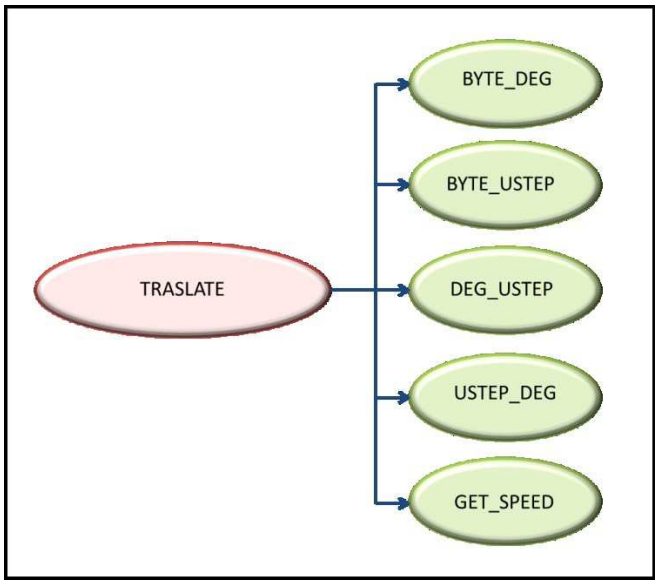

Figura 3.37: POUs della sezione Traslate

3.2.7 Sviluppo del software in linguaggio Java

L'interfaccia utilizzata per il Client in linguaggio Java, è la seguente:

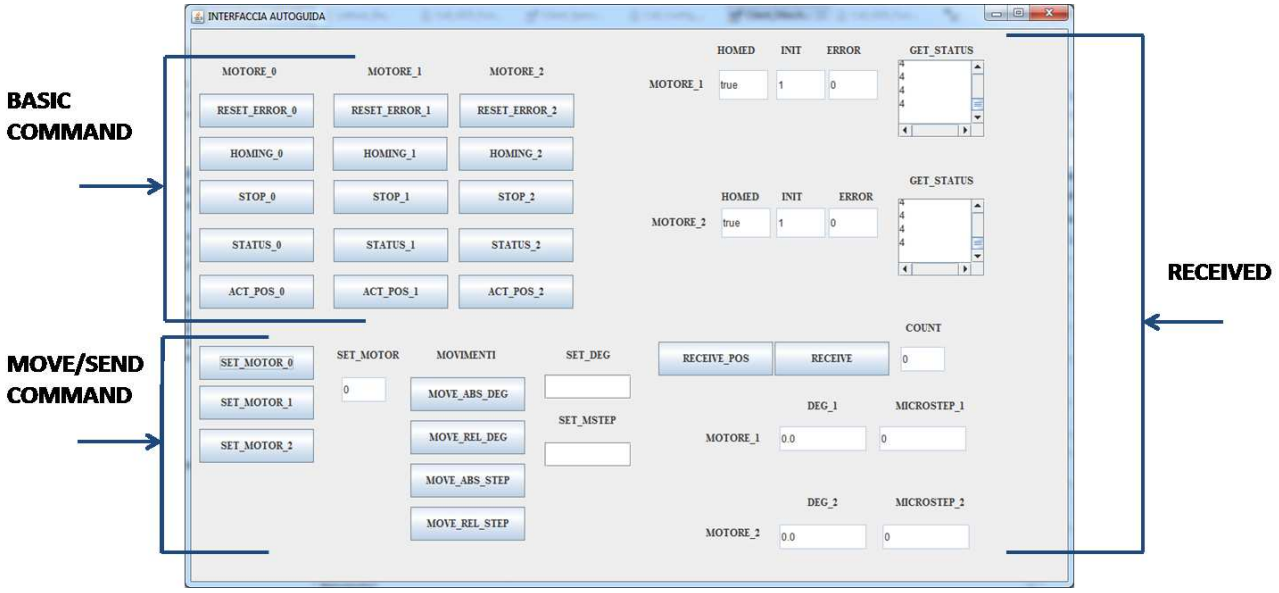

Figura 3.38: Interfaccia Client del sottosistema di autoguida

Come si evince da tale interfaccia, sono state realizzate le seguenti funzioni sia per ogni singolo motore che per entrambi i motori:

- RESET_ERROR
- HOMING
- STOP
- STATUS
- ACT_POS

Per quanto riguarda i comandi di movimento, come nel server TwinCAT 3, inizialmente si imposta il motore o i motori ai quali il comando deve essere inviato mediante le seguenti funzioni:

- SET_MOTOR_0
- SET_MOTOR_1

- SET_MOTOR_2

Inoltre sono presenti due caselle di testo di sola scrittura, mediante le quali è possibile impostare la posizione di destinazione per poi inviare il comando desiderato tra:

- MOVE_ABS_DEG
- MOVE_REL_DEG
- MOVE_ABS_USTEP
- MOVE_REL_USTEP

A destra del pannello è presente la sezione di RICEZIONE con due sub-sezioni:

1) In alto si ottiene, per ogni motore, le variabili che occorre monitorare, le quali vengono sempre aggiornate leggendo le corrispondenti variabili del Server. Occorre sottolineare che tali variabili sono tutte array essendo il dispositivo composto da due motori, pertanto in ogni casella di testo viene visualizzata la componente relativa a quel motore. Esse sono:
    - HOMED = indica se sul motore è stato fatto (true) o meno (false) l'homing
    - INIT = fornisce informazioni che riguardano la procedura di homing
    - ERROR = indica se si sono presentati errori durante la procedura di homing o durante un qualsiasi movimento
    - GET_STATUS = indica lo stato del motore
2) In basso sono presenti i pulsanti relativi alla ricezione delle risposte e COUNT che fornisce l'informazione della presenza di bytes nel buffer Rx che attendono di essere letti. Infine, per ogni singolo motore, viene visualizzata la posizione attuale sia in gradi che in microstep.

3.3 Il sottosistema di calibrazione

Questo sottosistema è composto da due stepper motors, ognuno dei quali aziona uno stage lineare; più precisamente è presente un sistema di controllo per lo specchietto di calibrazione e uno per il gruppo maschera. Il controllo del singolo motore viene effettuato dal PC embedded CX5020 di Beckhoff in combinazione con il singolo modulo Beckhoff EL7031.

In seguito sarà illustrato come configurare il PC CX5020 in modo da poter controllare un dispositivo connesso al bus EtherCAT in modo analogo al sottosistema polarimetrico.

3.3.1 Requisiti

Per il sottosistema di calibrazione, sia per lo specchietto delle calibrazioni che per la maschera, è richiesta una movimentazione lineare ad un grado di libertà con precisone di posizionamento non superiore a 0.5 mm.

La risoluzione del motore come riportata nelle caratteristiche (si veda il paragrafo 2.3.3.1, tabella 2.6) è di circa 0.02 μm.

Le caratteristiche di risoluzione del motore unite alle funzionalità implementate hanno consentito quindi di soddisfare con ampio margine il requisito richiesto.

3.3.2 Sviluppo del software TwinCAT per il sottosistema di calibrazione

3.3.2.1 Configurazione delle variabili

Si consideri il modulo EL7031 relativo allo specchietto: in questa sezione, si procede in modo del tutto analogo al motore Standa utilizzato con il modulo EL7041 (si veda il paragrafo 3.1.2), utilizzando il linguaggio IEC 61131-3 Structured Text (ST). Inizialmente si creano le variabili relative allo switch di fine corsa per lo specchietto e la variabile Asse_2 di tipo AXIS_REF. Essendo delle variabili che devono essere viste da tutto il sistema, esse si definiscono come variabili globali, aggiungendoli al file GVL, precedentemente creato (si veda il paragrafo 3.1.2.1) nella cartella GVLs, presente nella sezione *Configurazione PLC*. In modo analogo si procede per il modulo EL7031 relativo alla maschera. Il codice che tiene conto sia dello specchietto di calibrazione del gruppo lampade, sia della maschera per illuminare o schermare una delle due fibre ottiche è il seguente:

```
VAR_GLOBAL

    Asse_2              :   AXIS_REF;
    Asse_3              :   AXIS_REF;
    hs_maschera    AT%I* :  BOOL; // lo switch 1 è un input=ingresso
    hs_specchietto AT%I* :  BOOL; // lo switch 1 è un input=ingresso

END_VAR
```

Figura 3.39: Codice GVL per i motori del sistema di calibrazione

Il codice complessivo che tiene conto del motore Standa per il sottosistema polarimetrico, dei due motori per il sistema di calibrazione e del motore T-OMG per il sottosistema di autoguida è il seguente:

```
VAR_GLOBAL

    Asse_1          :   AXIS_REF;  // asse_1 deve essere di tipo AXIS_REF
    Asse_2          :   AXIS_REF;  // asse_2 deve essere di tipo AXIS_REF
    Asse_3          :   AXIS_REF;  // asse_3 deve essere di tipo AXIS_REF
    hs1          AT%I* :  BOOL; // lo switch 2 è un input=ingresso)
    hs2          AT%I* :  BOOL; // lo switch 1 è un input=ingresso)
    hs_maschera  AT%I* :  BOOL; // lo switch 1 è un input=ingresso)
    hs_specchietto AT%I* : BOOL; // lo switch 1 è un input=ingresso)

    RxBufferEL  : ComBuffer;  (* Receive data buffer; used with all receive function blocks *)
    TxBufferEL  : ComBuffer;  (* Transmit data buffer; used with all receive function blocks *)

END_VAR
```

Figura 3.40: Codice GVL per tutti i motori

Seguendo la procedura esposta in precedenza per il sottosistema polarimetrico, si ottengono i links tra le variabili “software” di immagini di processo PLC, e le variabili

"hardware" di immagini di processo dei terminali fisici I/O (di ingresso/uscita). Alle variabili globali che vengono definite inizialmente, Asse_2, Asse_3, hs_specchietto, hs_maschera, si associano i vari Input/Output dei vari moduli e lo status degli switches di fine corsa. Nel caso delle variabili Asse_2 e Asse_3 i vari links vengono creati automaticamente mentre per quanto concerne gli switches di fine corsa, questi vengono realizzati manualmente.

In particolare, alla scansione dei vari moduli del PC embedded, tutti i links principali tra il modulo *Configurazione NC* e il Terminale 6 (quello corrispondente ad un modulo EL7031) sono automaticamente creati. Le variabili "software" associate al Terminale 6, vengono linkate automaticamente con le variabili "hardware" presenti nella sezione Encoder del modulo *Configurazione NC*. Analogamente tutte le variabili "software" presenti in nella sezione Status di Terminale 6, vengono linkate con le variabili "hardware" presenti nella sezione Drive del modulo *Configurazione NC* (Links L1).

Inoltre vengono create due cartelle:

- *PLC Task_Standard Inputs*
- *PLC Task_Standard Outputs*

Facendo il download della configurazione fin qui implementata, vengono creati automaticamente tutti i links L2, tra le variabili "software" di Input e Output e le variabili "hardware" di ingresso e di uscita presenti nelle sottocartelle *From PLC* e *To PLC* di Asse 2.

Procedendo analogamente per il Terminale 7 e per l'altro modulo EL7031, al quale si ottiene associato la variabile Asse_3 di tipo AXIS_REF, si ottengono analoghi risultati.

In questo modo, nel caso del modulo EL7031, tali links vengono realizzati automaticamente dal sistema. Si ricordi che le variabili GVL.hs_specchietto e GVL.hs_maschera, relative agli switches di fine corsa non sono linkate, per cui occorre linkarle manualmente.

3.3.2.2 Configurazione dei parametri meccanici

Anche qui occorre impostare correttamente i parametri meccanici della movimentazione del singolo motore. Per realizzare ciò, occorre posizionarsi sulla scheda *Parametro* della sezione Asse 2 (specchietto). Qui, sono presenti diverse sezioni, tra cui "Velocities" che saranno viste in seguito (paragrafo 3.3.2.2.1).

Dai dati di targa del motore [24], è noto che:

- R = Default Resolution = numero di microstep per 1 step = 64 microstep/step
- 1 microstep = 0.0238125 µm
- Movimento lineare per 1 giro = 0.3048 mm
- Numero di steps per 1 giro = 200 steps

Si calcola la velocità di riferimento e la velocità massima.

Per quanto riguarda la v_{max}, si ottiene:

$$v_{max} = base\ frequency/motor\ frequency \tag{3.43}$$

In base alle considerazioni fatte nel paragrafo 3.1.1.2 ed avendo una base frequency o speed range di default pari a 2000 step/s, si ottiene:

$$v_{max} = \frac{2000\ step/s}{200\ step/rev} = 10\ rev/s \tag{3.44}$$

Per quanto riguarda il parametro Scaling Factor (SF), per i moduli EL7031 non è previsto alcun ingresso per un encoder, pertanto occorre impostare tale parametro:

$$SF = \frac{distance\ per\ rev}{full\ step * micro\ step} \tag{3.45}$$

$$SF = \frac{360°/rev}{200\ step/rev * 64\ micro\ step/step} = 0.028125°/step \tag{3.46}$$

Se si vuole esprimere lo SF in mm/step, dalla (3.46) si ottiene:

$$SF = \frac{0.3048\ mm/rev}{200\ step/rev * 64\ micro\ step/step} = 0.0000238125\ mm/microstep \quad (3.47)$$

La velocità di riferimento, espressa in gradi al secondo, è data da:

$$v_{ref} = \frac{rev}{s} * 360° \quad (3.48)$$

In mm/s si ottiene:

$$v_{ref} = \frac{step}{rev} * \frac{Mov\ lin}{step} * \frac{rev}{s} \quad (3.49)$$

Occorre calcolare il movimento lineare per 1 step, sapendo che

$$1\ step = 64\ microstep \quad (3.50)$$

Per quanto visto in precedenza, occorre calcolare la distanza lineare corrispondente a 64 microstep:

$$64\ microstep = \frac{Mov\ lineare}{microstep} * \frac{microstep}{step} \quad (3.51)$$

$$64\ microstep = 0.0238125\ \mu m * 64 = 1.524\ \mu m \tag{3.52}$$

Pertanto si ottiene:

$$1\ step = 1.524\ \mu m \tag{3.53}$$

Quindi:

$$v_{ref} = \frac{200}{1} * \frac{1.524\ \mu m}{1} * \frac{10}{1} = 0.03048\ \mu m/s = 3.048\ mm/s \tag{3.54}$$

Come detto nel paragrafo 3.1.1.2, tale parametro viene utilizzato dal software per ricavarsi le seguenti informazioni:

1) Il valore di un fullstep in mm

$$1\ fullstep = \frac{3.048\ mm/s}{2000\ step/s} = 0.001524\ mm/step \tag{3.55}$$

2) Il valore del microstep in mm

$$1\ microstep = \frac{0.0025\ mm/step}{64\ micro\ step/step} = 0.0000238125\ mm/microstep \tag{3.56}$$

3) Il numero di fullstep di cui muoversi corrispondenti a x mm, per un movimento di 1 mm, in base alla seguente proporzione, è dato da:

$$2000\ fullstep : 3.048\ mm = x\ fullstep : 1\ mm \tag{3.57}$$

da cui

$$x = \frac{1\ mm * 2000\ Fs}{3.048\ mm} \cong 656.1678\ step = 41994.75\ microstep \tag{3.58}$$

Pertanto, per un movimento di 1 mm, il rotore si muoverà di 656.1678 step. Invertendo si ha che il motore si è mosso di 1 step per ogni 0.001524 mm.

$$\frac{1}{656.1678\ fstep} = 0.001524\ mm \tag{3.59}$$

In microstep si ottiene:

$$\frac{1}{41994.75\ \mu step} = 0.0000238125\ mm = \ 23.8125 * \ 10^{-6}\ mm \tag{3.60}$$

Per ogni $23.8125 * \ 10^{-6}\ mm$ il motore si muove di un microstep.

4) Il numero di fullstep al secondo di cui muoversi corrispondenti a x mm/s, per un movimento di 1 mm/s, in base alla seguente proporzione:

$$3.048\ mm/s : 2000\ fstep/s = 1mm/s : x\ fstep/s \tag{3.61}$$

è pari a:

$$x = \frac{2000\ fstep/s * 1\ mm/s}{3.048\ mm/s} \cong 656.1678 \frac{fstep}{s} = 41994.75\ \frac{microstep}{s} \quad (3.62)$$

Di conseguenza, per un movimento di 1 mm/s, il rotore si muoverà di circa 656.1678 step/s, quindi di 41994.75 microstep/s.

Invertendo si ha che il motore si muove di 1 step per ogni 0.001564 s.

$$\frac{1}{656.1678\ fstep/s} \cong 0.001564\ s \quad (3.63)$$

In microstep, si ottiene:

$$\frac{1}{41994.75\ \mu step/s} \cong 0.0000238125\ s = 23.8125 * 10^{-6}\ s \quad (3.64)$$

Pertanto per ogni 23.8125 * 10^{-6} s il motore si muove di un microstep. In conclusione occorre impostare:

- v_{ref} = 3.048 mm/s
- v_{max} = 3 mm/s
- SF = 0.0000238125 mm/microstep

Il Rapporto del Motore definito come:

$$R_M \triangleq \frac{base\ frequency}{v_{ref}} \quad (3.65)$$

è pari a:

$$\frac{base\ frequency}{v_{ref}} = \frac{2000\ fstep/s}{3.048\ mm/s} \cong 656.1678 \frac{step}{mm} \cong 41994.75\ \frac{microstep}{mm} \quad (3.66)$$

Per ottenere una movimentazione più veloce ed egualmente precisa, è possibile modificare i suoi parametri, base frequency o speed range e v_{ref}, ma tale rapporto deve rimanere inalterato. Si considerino quali sono i vantaggi e gli svantaggi impostando solo una v_{ref} più grande e di lasciando inalterato il parametro base frequency *o* speed range:

Vantaggio: avere una velocità massima teorica più grande comporta che il motore si muove più velocemente;

Svantaggio: il motore si muove più velocemente ma la movimentazione è meno fine perché modificando il v_{ref} di 4 volte è come se modificassimo il rapporto di risoluzione *R* da 64 microstep/step a 64/4=16 micro step/step, risultando una diminuzione del il rapporto di risoluzione. Più alto è il rapporto di risoluzione, più fine sarà la movimentazione.

3.3.2.2.1 Parametri Asse 2 e Asse 3

I parametri seguenti, presenti nella sezione Velocities vengono impostati ad un valore che è stato scelto sulla base della relazione (3.54) e in base a prove in laboratorio.

- velocità manuale (rapida) = 2.0 mm/s
- velocità manuale (lenta) = 1.0 mm/s
- velocità calibrazione (in avanti) = 2.0 mm/s
- velocità calibrazione (all'indietro) = 0.1 mm/s
- incremento spinta (in avanti) = 1.0 mm/s

3.3.2.2.1.2 Parametri Encoder

Come tipo di Encoder si sceglie un Encoder simulato non essendo presente alcun encoder collegato al modulo EL 7041. Il fattore di scala si sceglie in base alla relazione mentre per l'homing , avendo scelto hs2 come sensore di fine corsa o di homing e poiché lo stage si muove di default verso hs1, occorre selezionare al valore TRUE la voce "Inverti

direzione ricerca eccentrico calibrazione". Una spiegazione di quanto affermato sarà illustrata nel paragrafo 3.3.2.5.

- Type: Codificatore simulazione
- Scaling Factor (fattore di scala) = 0.0000238125 mm/step
- Sezione Homing: Inverti direzione ricerca eccentrico calibrazione: FALSE

3.3.2.3 Configurazione dei parametri elettrici

Nel manuale on-line relativo al modulo EL7031, [24], è riportato sia il valore predefinito (default) sia l'unità di misura per ogni parametro. Tra tutti, occorre impostare i seguenti:

- Maximal Current = 240 mA/Phase
- Nominal Voltage = 5000 mV
- Motor Coil Resistance = 2040 (unità = 0.01 Ohm) → 20.4 Ohm*100
- Motor Fullsteps = 200
- Speed range = 2000 Fullsteps/s

3.3.2.4 Configurazione calibrazione (homing)

Come per il motore Standa, anche per questo tipo di motore, la procedura di homing deve essere programmata. Si consideri il modulo EL7031 per lo specchietto delle calibrazioni, in questo caso si può definire un'istanza della FB MC_Home nel file (PRG) SPECCHIETTO:

```
Home        : MC_Home;
```

Occorre verificare quali sono le direzioni positiva e negativa del movimento. Andando sulla scheda "In linea" di Asse 1 e selezionando la casella "Movimento in avanti", che per convenzione rappresenta la direzione positiva, si osserva che lo stage si muove verso la direzione opposta rispetto alla posizione del sensore cioè si allontana dal sensore,

pertanto si può affermare che la direzione positiva è la direzione per la quale lo stage si allontana dal sensore mentre quella negativa è la direzione per la quale lo stage si avvicina al sensore.

Jog- = direzione verso il sensore
Jog+ = direzione opposta al sensore

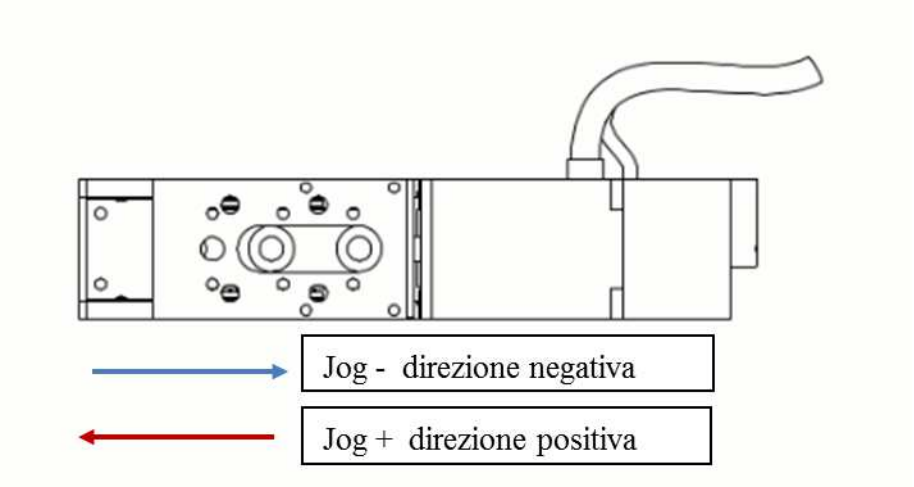

Figura 3.41: Direzioni movimento *stage* Zaber LSA

In base alle considerazioni fatte nel paragrafo 3.1.2.4 e poiché questo tipo di motore presenta un solo sensore di homing, il codice è il seguente:

```
IF NOT Home.Active THEN
   IF hs_specchietto THEN
  Enable_asse2.Enable_Negative:=FALSE;
   ELSE
  Enable_asse1.Enable_Negative:=TRUE;
   END_IF
END_IF
```

Figura 3.42: Codice homing per lo specchietto

Infine si richiama l'istanza della FB MC_Home:

```
Home(
   Axis:=Asse_2,
   HomingMode:=MC_DefaultHoming,
   bcalibrationcam:=hs_specchietto,
   Execute:=bhome,
   Busy => bBusy);
```

Figura 3.43: Codice di chiamata dell'istanza Home per lo specchietto

In modo analogo si procede per la procedura di homing per il motore relativo alla machera di illuminazione/oscuramento delle fibre ottiche. Per completezza si riporta il codice di homing e della FB MC_Home:

```
IF NOT Home.Active THEN
  IF hs_maschera THEN
        Enable_asse3.Enable_Negative:=FALSE; // disattivo la direzione negativa(verso destra)
  ELSE
        Enable_asse3.Enable_Negative:=TRUE;
  END_IF
END_IF
```

Figura 3.44: Codice homing per la maschera

```
Home(
   Axis:=Asse_3,
   HomingMode:=MC_DefaultHoming,
   bcalibrationcam:=hs_maschera,
   Execute:=bhome,
   Busy => bBusy);
```

Figura 3.45: Codice di chiamata dell'istanza Home per la maschera

3.3.2.5 Configurazione del movimento con controllo della velocità (jog)

Valgono le stesse considerazioni fatte per il motore Standa, per cui si riporta soltanto il codice. Dopo aver istanziato la FB:

```
Move_JOG        : MC_Jog;
```

si implementa il seguente codice:

```
//codice-condizioni per le funzioni jog (movimento a impulsi)

IF bjogFW OR bjogBW THEN
  Move_JOG.Mode:= MC_JOGMODE_STANDARD_SLOW;
END_IF

IF bjogFFW OR bjogFBW THEN
  Move_JOG.Mode:= MC_JOGMODE_STANDARD_FAST;
END_IF

IF bjogFW OR bjogFFW THEN
  Move_JOG.JogForward:= TRUE;
ELSE
  Move_JOG.JogForward:= FALSE;
END_IF

IF bjogBW OR bjogFBW THEN
  Move_JOG.JogBackwards:= TRUE;
ELSE
  Move_JOG.JogBackwards:= FALSE;
END_IF
```

Figura 3.46: Codice per il movimento jog

Infine si richiama l'istanza della FB MC_Jog:

```
Move_JOG(Axis:=Asse_2);
```

Sostituendo Asse_2 con Asse_3, si ottiene il codice per il motore relativo alla maschera.

3.3.2.6 Configurazione del movimento con controllo della posizione (assoluto e relativo)

3.3.2.6.1 Movimento assoluto

Anche in questo caso, valgono le stesse considerazioni fatte per il motore Standa, per cui si riporta soltanto il codice. Inizialmente si istanzia la FB:

```
Move_ABS        : MC_Moveabsolute;
```

successivamente si richiama la sua istanza:

```
Move_ABS(
    Axis:=Asse_2,
    execute:= bABS,
    position:=lpositionABS ,
    velocity:=lvelocityABS );
```

Figura 3.47: Codice di chiamata dell'istanza Move_ABS

Sostituendo Asse_2 con Asse_3, si ottiene il codice per il motore relativo alla maschera.

3.3.2.6.2 Movimento relativo

Si riporta soltanto il codice. Inizialmente si istanzia la FB:

```
Move_REL        : MC_Moverelative;
```

successivamente si richiama la sua istanza:

```
Move_REL(
    Axis:=Asse_1,
    execute:=bREL,
    distance:=ldistanceREL,
    velocity:=lvelocityREL );
```

Figura 3.48: Codice di chiamata dell'istanza Move_REL

Nella tabella seguente sono riportate le Function Blocks definite e la relativa istanza.

ISTANZA	FUNCTION BLOCK
Enable_asse_2	MC_Power
Move_ABS	MC_Moveabsolute
Move_REL	MC_Moverelative
Move_JOG	MC_Jog
Reset	MC_Reset
Home	MC_Home
Stop	MC_Stop
Write_parameter	MC_Writeparameter
Setpos	MC_Setposition
Status_parameter	MC_ReadParameter

Tabella 3.4: Function Blocks istanziate per lo specchietto

In figura 3.49 sono rappresentate le Function Blocks definite per lo specchietto

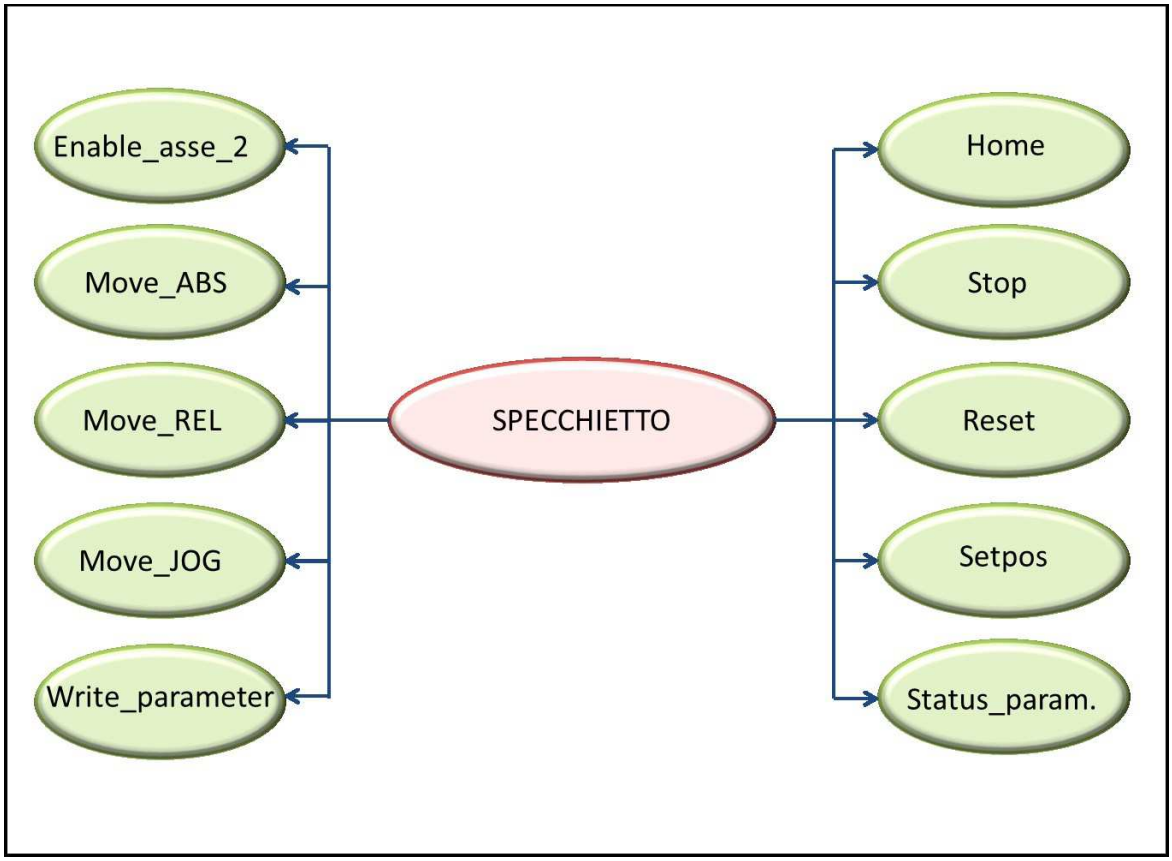

Figura 3.49 : Function Blocks definite per la sezione Specchietto

Sostituendo Asse_2 con Asse_3, si ottiene il codice per il motore relativo alla maschera

Si riporta tabella e figura.

ISTANZA	**FUNCTION BLOCK**
Enable_asse_3	MC_Power
Move_ABS	MC_Moveabsolute
Move_REL	MC_Moverelative
Move_JOG	MC_Jog
Reset	MC_Reset
Home	MC_Home
Stop	MC_Stop
Write_parameter	MC_Writeparameter
Setpos	MC_Setposition
Status_parameter	MC_ReadParameter

Tabella 3.5: Function Blocks istanziate per la maschera

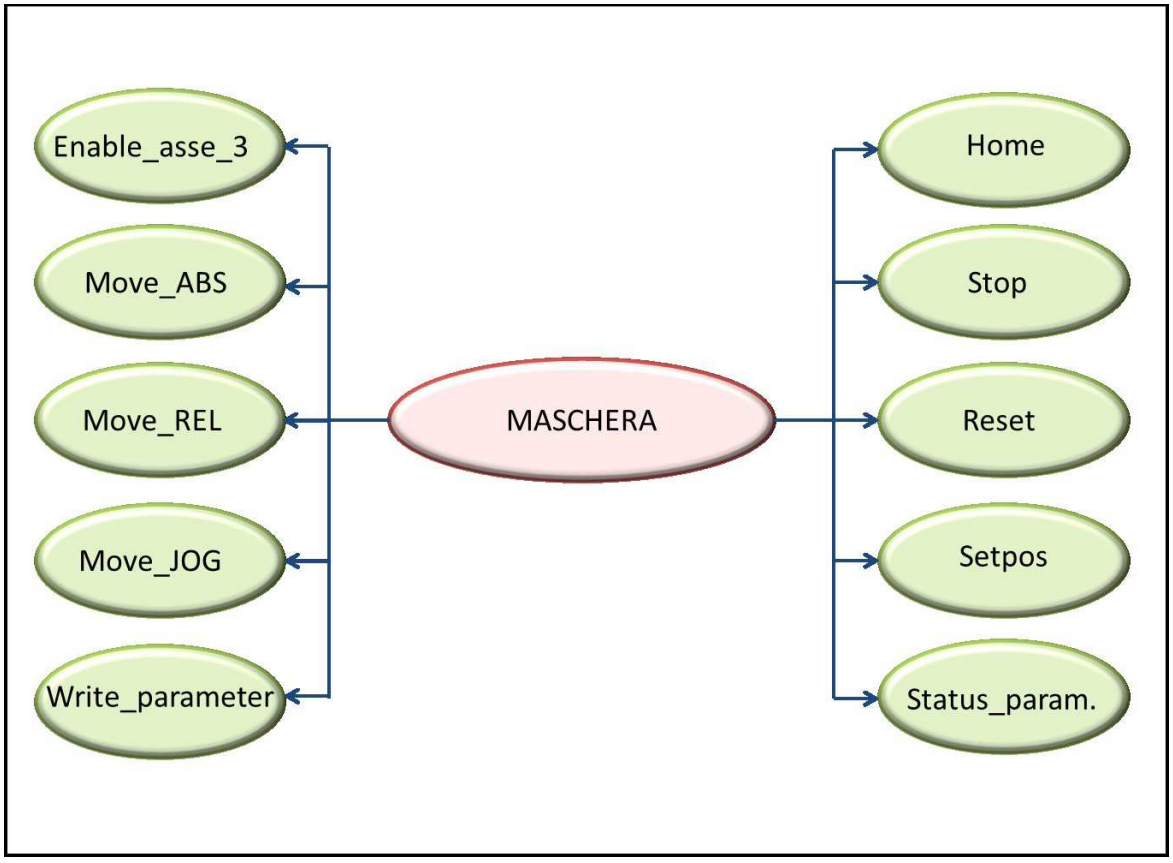

Figura 3.50 : Function Blocks definite per la sezione Maschera

3.3.2.7 Implementazione dello stato del motore

Sono state definite le stesse variabili con gli stessi possibili valori che esse possono assumere, come già illustrato per il motore Standa (si veda il paragrafo 3.1.2.7).

1) Set Init

In tale sezione è definita la variabile *iSetInit* che restituisce il valore 0 oppure 1 a seconda che sia stato ricevuto correttamente il comando di homing.

2) Get Init

In tale sezione è definita la variabile *iGetInit* che restituisce il valore 0 oppure 1 a seconda che il comando di homing sia stato eseguito correttamente.

3) Get Status

In tale sezione è definita la variabile *iGetStatus* che restituisce il valore compreso tra (0÷5) a secondo dello stato del motore.

4) Get Error

In tale sezione è definita la variabile *iGetError* che restituisce il valore compreso tra (0÷5) a secondo del tipo di errore.

3.3.2.8 Pannelli di visualizzazione

Anche in questo caso viene implementata un'interfaccia grafica dalla quale poter visualizzare i comandi da inviare e gli stati delle variabili più significative, sia per lo specchietto che per la maschera, del tutto simili al pannello per il motore Standa (paragrafo 3.1.2.8).

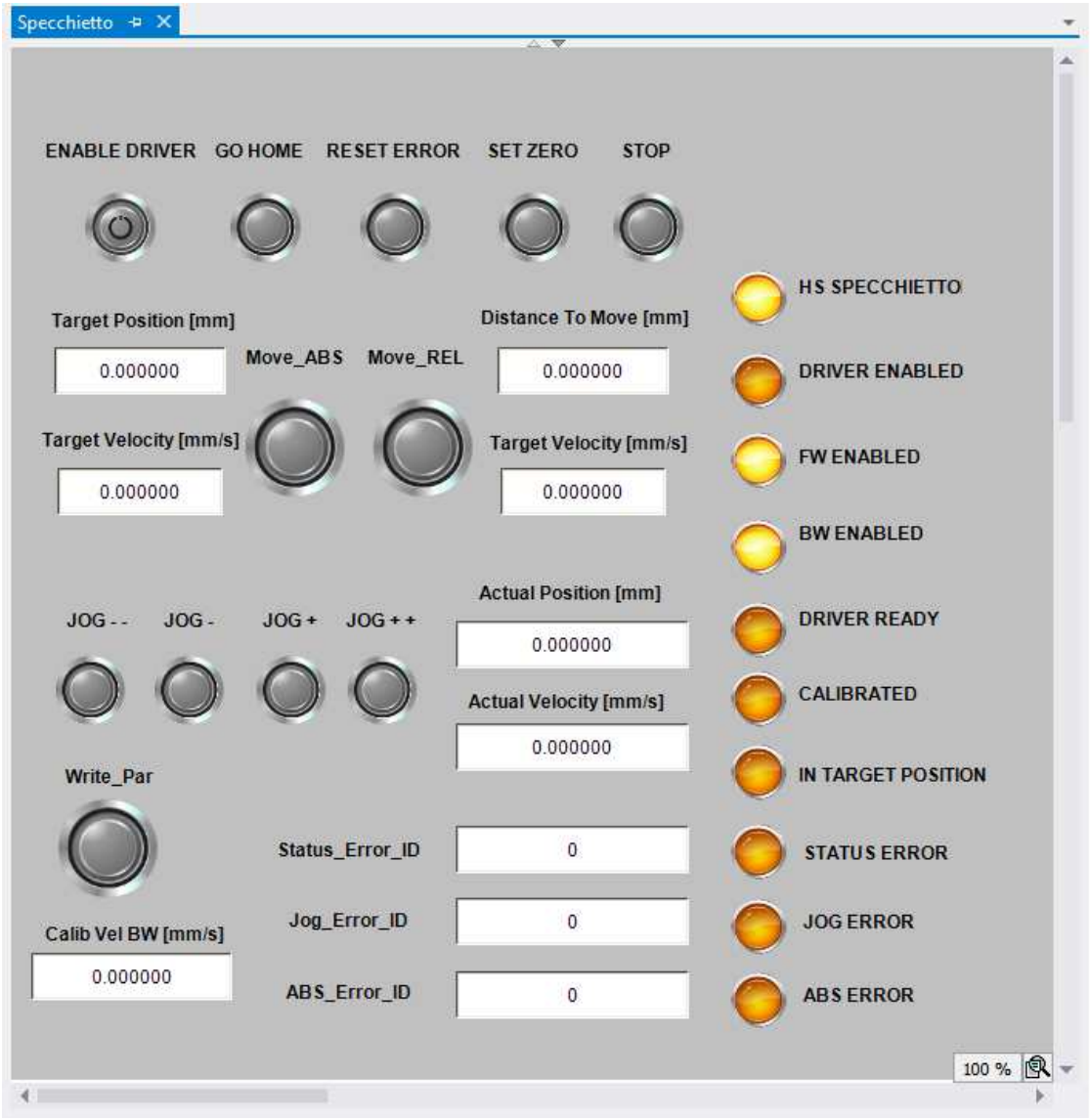

Figura 3.51: Pannello di visualizzazione TwinCAT 3 dello specchietto

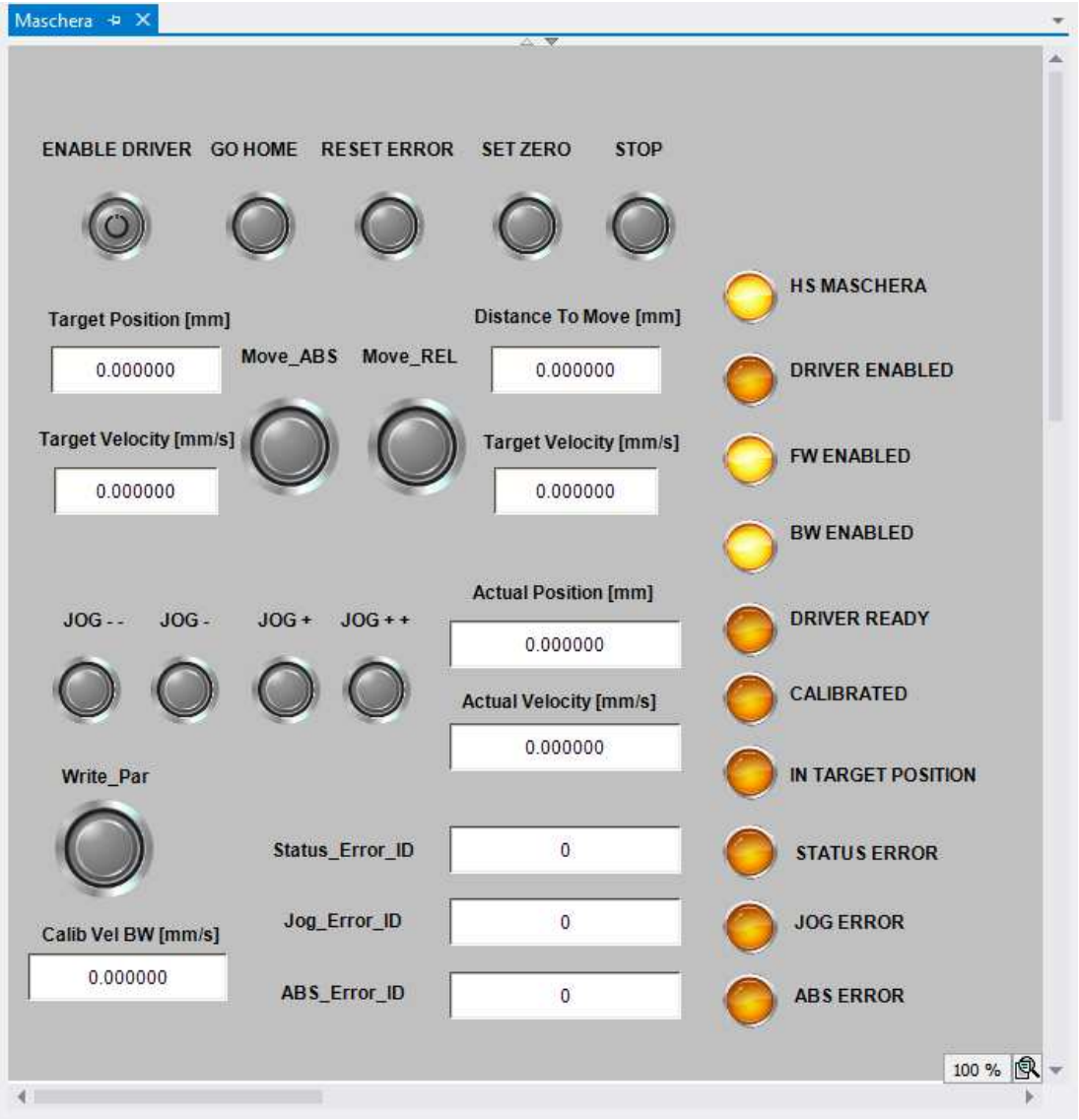

Figura 3.52: Pannello di visualizzazione TwinCAT 3 della maschera

3.3.2.9 Sviluppo del software in linguaggio Java

Facendo uso delle stesse librerie Java discusse nel paragrafo 3.1.2.9, è possibile realizzare le interfacce per lo specchietto e per la maschera.

L'interfaccia dello specchietto è la seguente:

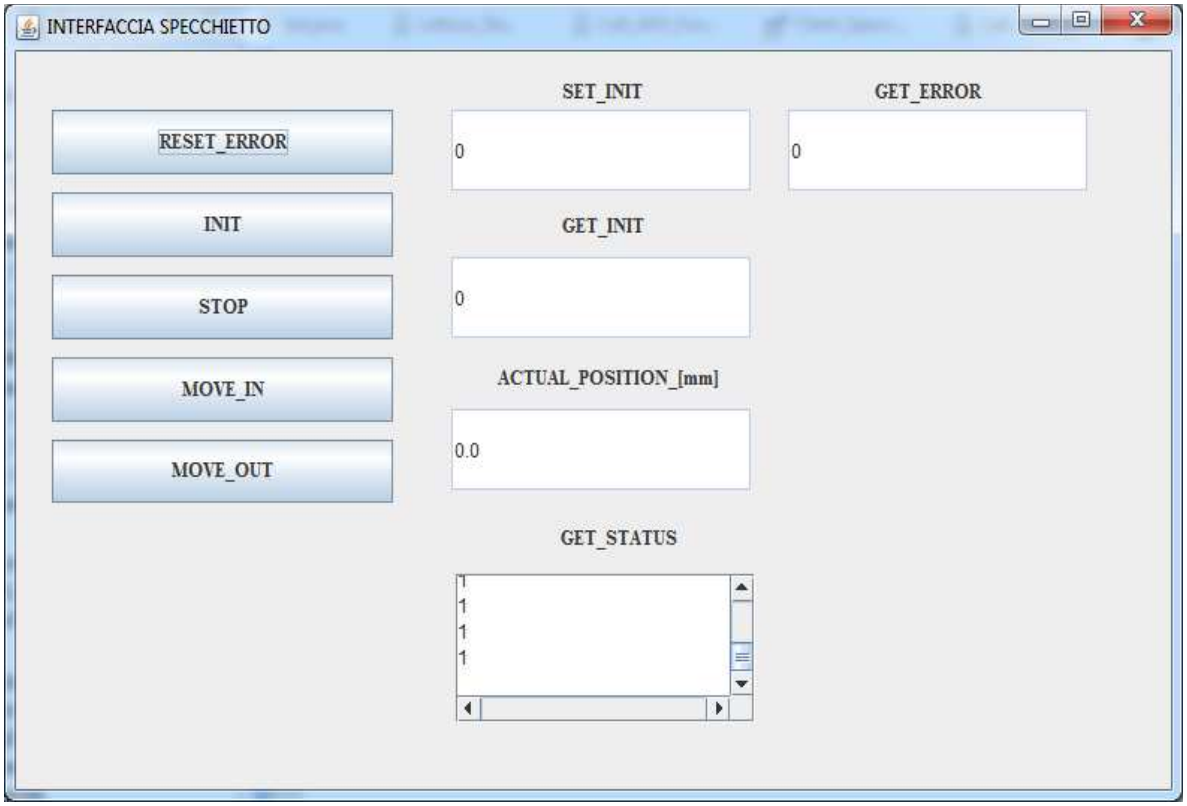

Figura 3.53: Interfaccia Client dello specchietto

Per l'interfaccia dello specchietto, sono state realizzate le seguenti funzioni:

- RESET_ERROR

Tale funzione è necessaria quando si verificano degli errori durante l'homing o durante il funzionamento del motore, a causa dei quali il motore stesso va nello stato di "stallo". Per poter sbloccare il motore occorre inviare tale comando.

- INIT

Questa funzione ingloba vari comandi. Il primo comando che viene inviato al Server TwinCAT è il comando Enable che consente di abilitare il motore. Successivamente viene inviato il comando di Homing.

- STOP

Questa funzione invia il comando di arresto al motore.

- MOVE_IN

Questa funzione invia al motore un comando per effettuare un movimento assoluto verso la posizione di 25 mm, a partire dalla posizione di zero impostata dall'Homing, che

in prima approssimazione dovrebbe essere la posizione di intercettazione del fascio luminoso.

- MOVE_OUT

Questa funzione invia al motore un comando per effettuare un movimento assoluto verso la posizione di 0 mm, in modo tale che lo specchietto non intercetti il fascio luminoso.

Inoltre sono presenti delle caselle di testo che ci informano sulle variabili impostate nel Server e viste in precedenza; tra queste è presente l'ACTUAL POSITION che restituisce la posizione corrente del motore.

L'interfaccia della maschera è la seguente:

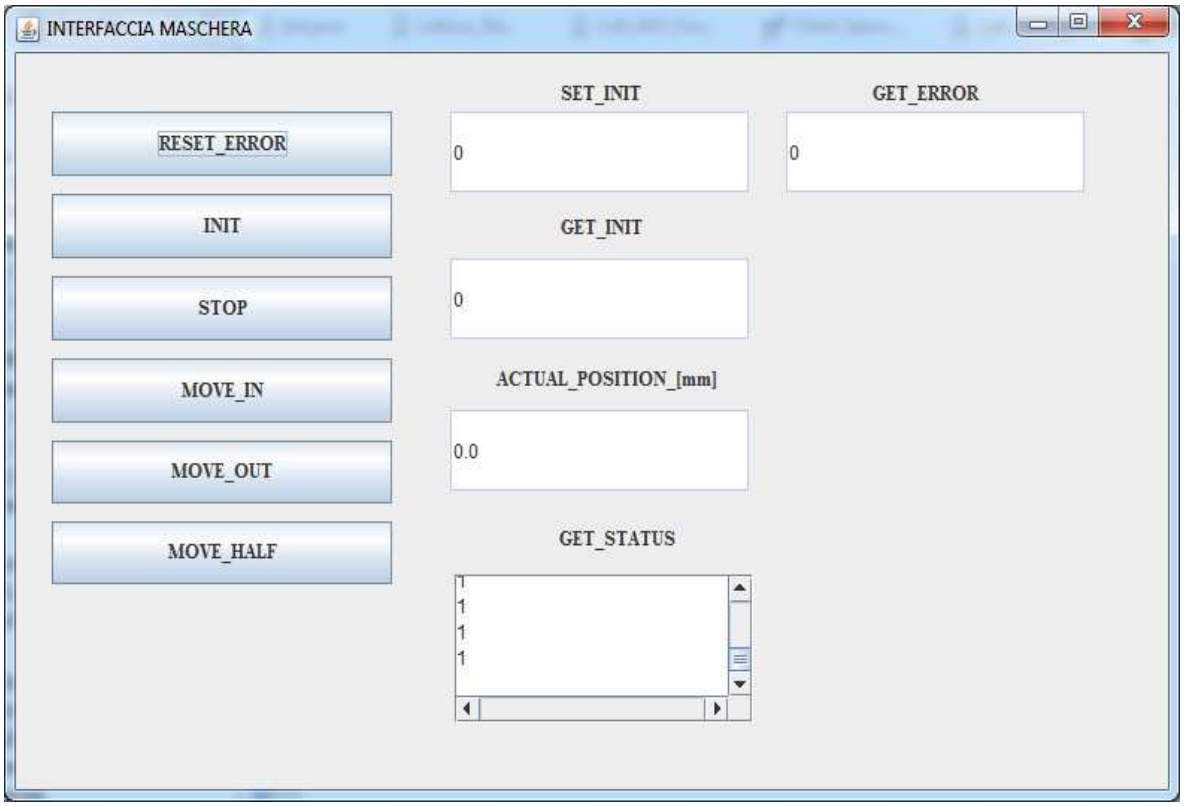

Figura 3.54: Interfaccia Client della maschera

Per l'interfaccia della maschera, sono state realizzate le seguenti funzioni:

- RESET_ERROR

Tale funzione è necessaria quando si verificano degli errori durante l'homing o durante il funzionamento del motore, a causa dei quali il motore stesso va nello stato di "stallo". Per poter sbloccare il motore occorre inviare tale comando.

- INIT

Questa funzione ingloba vari comandi. Il primo comando che viene inviato al Server TwinCAT è il comando ENABLE che consente di abilitare il motore. Successivamente viene inviato il comando di Homing.

- STOP

Questa funzione invia il comando di arresto al motore.

- MOVE_IN

Questa funzione invia al motore un comando per effettuare un movimento assoluto, a partire dalla posizione di zero impostata dall'Homing, verso la posizione di 25 mm che in prima approssimazione dovrebbe essere la posizione di intercettazione del fascio luminoso.

- MOVE_OUT

Questa funzione invia al motore un comando per effettuare un movimento assoluto verso la posizione di 0 mm, in modo tale che lo specchietto non intercetti il fascio luminoso.

- MOVE_HALF

Questa funzione invia al motore un comando per effettuare un movimento assoluto verso la posizione di 12.5 mm, in modo tale che la maschera illumini soltanto una fibra ottica. Tale posizione rappresenta il prototipo delle posizioni "Fibra A" e "Fibra B".

Inoltre sono presenti delle caselle di testo che ci informano sulle variabili impostate nel Server e viste in precedenza; tra queste è presente l'ACTUAL POSITION che restituisce la posizione corrente del motore.

Conclusioni

Per la realizzazione della nuova interfaccia tra il telescopio 91 cm e lo spettrografo CAOS sono stati selezionati dei prodotti che, attualmente, rappresentano lo stato dell'arte nell'ambito del controllo e dell' automazione industriale, in modo da avere il massimo delle prestazioni tecnologiche con la massima compattezza di assemblaggio, alta affidabilità e ridotta manutenzione. I dispositivi elettronici e i controllori dedicati al movimento e al controllo dei sottosistemi costituenti la nuova interfaccia al telescopio sono stati scelti dall' industria di automazione Beckhoff, altamente specializzata in questo settore, in modo da avere la migliore omogeneità della nuova interfaccia insieme con le migliori prestazioni in tempo reale e ci ha consentito di effettuare dei test su di un'elevata varietà di funzionalità sviluppate dall' azienda selezionata. Allo stesso modo i dispositivi elettronici per il supporto ottico-meccanico per ogni singolo sottosistema, sono stati scelti dalle industrie Zaber e Standa che essendo all'avanguardia, ci hanno consentito di ottenere un'elevata precisione così come richiesta dalle specifiche del progetto.

In riferimento alla precisione, considerando che l'immagine di una stella sul piano focale del telescopio dovuta all'atmosfera è dell'ordine di 2" (secondi d'arco) e che le dimensioni proiettate in cielo dalla fibra ottica sono minori di 5 secondi d'arco, lo scopo dell'attività relativamente al tip-tilt è stato il raggiungimento di una precisione tale da seguire un moto apparente della stella pari a 0.5", pertanto la precisione richiesta per il riposizionamento dell'oggetto da osservare nella posizione di riferimento, per la movimentazione dello specchietto tip-tilt, è pari a 0.5" equivalente a 2.4 μrad.

Elenco delle Figure

Elenco delle Tabelle

Elenco degli Acronimi

- ADS= Automation Device Specification
- ASIC= Application Specific Integrated Circuit
- ASTRI= Astrofisica a Tecnologia Replicante Italiana
- BF = Base Frequency
- BPS = Byte Per Second
- BW = Backward
- CAOS= Catania Astrophysical Observatory Spectropolarimeter
- CCD= Charge Coupled Device
- DC= Direct Current
- DLL = Dynamic Link Library
- DVI-D= Digital Visual Interface-Digital
- E-Bus = EtherCAT Bus
- EIA RS-232= Electronic Industries Alliance Recommended Standard 232
- ENI = Engineering Interface
- ESC = EtherCAT Slave Controller
- ESO = European Southern Observatory
- EtherCAT = Ethernet for Control Automation Technology
- FB = Function Block
- FBD = Function Block Diagram
- FEROS = Fiber-fed Extended Range Optical Spectrograph
- FPGA = Field Programmable Gate Array
- FRESCO = Fiber-optic Reosc Echelle Catania Observatory
- FUN = Function
- FW= Forward
- GUI = Graphical User Interface
- GVL = Global Variable List
- HARPS = High Accuracy Radial velocity Planet Searcher
- HS = Home Switch
- HW = Hardware

- IEC = International Electrotechnical Commission
- IL = Instruction List
- INAF = Istituto Nazionale di Astrofisica
- I/O = Input/Output
- LD = Ladder Diagram
- LVDS = Low Voltage Differential Signalling
- M1 = Azimuth Axes
- M2 = Elevation Axes
- NC= Numerical Control
- PDO = Process Data Object
- PLC = Programmable Logical Controller
- POU = Program Organization Units
- PRG = Program
- PTP = Point To Point
- PWM = Pulse-Width Modulation
- RAM = Random Access Memory
- RJ-45 = Registered Jack type 45
- SF = Scaling Factor
- SFC = Sequential Function Chart
- SLN = Serra La Nave
- ST = Structured Text
- SW = Software
- T-OMG = Motorized Two-Axis Optic Mount
- TCP/IP = Tasmission Control Protocol (TCP) Internet Protocol (IP)
- TwinCAT = The Windows Control and Automation Technology
- USB = Universal Serial Bus
- UVES = Ultraviolet and Visual Echelle Spectrograph
- UPS = Uninterruptible Power Supply
- XAE = eXtended Automation Engineering
- XAR = eXtended Automation Runtime

Bibliografia

[1] Salvo Scuderi, http://www.oact.inaf.it/weboac/Rapp_Int_Tec/Scuderi_RIT_13-2015.pdf

[2] David F. Gray, *The observation and analysis of stellar photospheres*, Cambridge University Press, 1992

[3] F. Leone et al., *A method to calibrate the high-resolution Catania Astrophysical Observatory Spectropolarimeter*, The Astronomical Journal, vol. 151, pp. 116-124, May 2016

[4] D. Clarke, J.F. Grainger, *Polarized Light and Optical Measurement*, Pergamon Press Oxford, 1971

[5] H. Dekkel et al., *Design, construction and performance of UVES, the echelle spectrograf for the UT2 Kueyen Telescope at the ESO Paranal Observatory*. SPIE 4008, 534-545, 2000

[6] A. Kaufer et al., *Commissioning FEROS, the New High-Resolution Spectrograph at La Silla*. The Messenger 95, 8-12, 1999

[7] M. Mayor et al., *Setting new Standard with Harps*, The Messenger 114, 20-24

[8] J. Tinbergen, R. Rutten, *A User's Guide to WHT Spectropolarimetry*, ING, La Palma User Manual No. 21, 1992

[9] F. Leone et al, *Spectroscopic study of the HgMn star 49606: the quest for binarity, abundance stratifications and magnetic field*, MNRAS 460, 1999-2007, 21 Aprile 2016

[10] E. Landi Degl'Innocenti, M. Landolf, *Polarization in spectral lines*, Kluwer Academic Publishers, 2004

[11] R. Goodrich et al., *Spectropolarimetry. II. Circular Polarization Optics and Techniques*, PASP 107, 179-183, 1995

[12] Beam Sampler, https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=913

[13] Beckhoff Automation, *Hardware Documentation for CX5020 Embedded-PC*, https://www.beckhoff.com/english.asp?embedded_pc/cx5010_cx5020.htm

[14] TwinCAT System – guida on-line, https://infosys.beckhoff.com/english.php

[15] Beckhoff Automation, *TwinCAT 3 | eXtended Automation (XA)*, https://www.beckhoff.com/english.asp?twincat/twincat-3.htm

[16] Beckhoff Automation, *TwinCAT 3 Getting Started*, ftp://ftp.beckhoff.com

[17] Beckhoff Automation, *TwinCAT Quick Start*, https://download.beckhoff.com

[18] Il protocollo ADS, https://infosys.beckhoff.com/english.php?content=../content/1033/tcadscommon/html/tcadscommon_intro.htm&id=

[19] Karl-Heinz John, Michael Tiegelkamp, *IEC 61131-3: Programming Industrial Automation Systems*, Springer, 1995

[20] Beckhoff Automation, *Documentation for Digital Input Terminals EL1018,* https://www.beckhoff.com/english.asp?ethercat/el1018.htm

[21] Beckhoff Automation, *Documentation for Digital Output Terminals EL2008,* https://www.beckhoff.com/english.asp?ethercat/el2008.htm

[22] Beckhoff Automation, *Documentation for Analog Input Terminals (12 Bit) EL3002*, https://www.beckhoff.com/english.asp?ethercat/el3001_el3002.htm

[23] Beckhoff Automation, *Documentation for Serial Interface Terminals EL6002,* https://www.beckhoff.com/english.asp?ethercat/el6002_el6022.htm

[24] Beckhoff Automation, *Documentation for Stepper Motor Terminals EL7031*, http://www.beckhoff.com/english.asp?ethercat/el7031.htm

[25] Beckhoff Automation, *Documentation for Stepper Motor Terminals EL7041,* https://www.beckhoff.com/english.asp?ethercat/el7041.htm

[26] Beckhoff Automation – guida on-line https://www.beckhoff.com

[27] Beckhoff Automation, Manual for TC3 Serial Communication TwinCAT, https://download.beckhoff.com

[28] Manuale motore ZABER T-OMG, www.zaber.com/wiki/Manuals/T-OMG

[29] Caratteristiche Motore ZABER T-OMG, www.zaber.com/products/product_group.php?group=T-OMG

[30] Manuale motore ZABER LSA, www.zaber.com/wiki/Manuals/LSA

[31] Caratteristiche Motore ZABER LSA,

www.zaber.com/products/product_group.php?group=LSA

[32] Motore STANDA, http://www.altechna.com/product_details.php?id=141, www.standa.lt

[33] Claudio De Sio Cesari, *Manuale di Java 7*, Hoepli, 2011

[34] Paolo Camagni, Riccardo Nikolassy , *Corso di Java – Dalla Programmazione ad Oggetti alle Applicazioni Grafiche*, Hoepli, 2013

[35] https://www.eso.org/sci/libraries/SPIE2014/9152-6.pdf

[36] E. Antolini et al., *Mount Control System of the ASTRI SST-2M prototype for the Cherenkov Telescope Array*, SPIE 9913, 2016