



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

UNIVERSITA' DEGLI STUDI DI PADOVA
Dipartimento di Ingegneria Industriale DII

Corso di Laurea Magistrale in Ingegneria dell'Energia Elettrica

Progetto ed implementazione su un microcontrollore delle routine di controllo di un motore brushless e della comunicazione seriale mediante protocollo SPI

Relatore

Prof. Manuele Bertoluzzo

Correlatore

Prof. Nicola Bianchi

Studente con matricola

Maximilian Graf

2005776

Anno Accademico 2021/2022

Indice

Lista delle figure	III
Lista delle tabelle	VI
Sommario	VII
Introduzione	VIII
1. La Formula Student	1
1.1 La competizione di Formula Student	1
1.2 Il RaceUP Team e la Divisione Elettrica	2
1.3 SG-e 04	3
1.4 AMK racing kit	4
2. Strumenti usati	9
2.1 Hardware	9
2.1.1 Microcontrollore	9
2.1.2 Gate driver	11
2.1.3 IGBT	12
2.2 Software	13
3. La comunicazione con i gate drivers	15
3.1 Il protocollo SPI	16
3.2 Il modulo SPI del microcontrollore TMS320F28379D	17
3.3 Sviluppo delle routine di comunicazione	20
3.3.1 Inizializzazione	22
3.3.2 Comunicazione	23
3.3.3 Gestione degli errori	27
3.4 Collaudo	29
4. Algoritmi di controllo di un motore brushless	31
4.1 Il sistema di riferimento sincrono dq	31
4.2 Equazioni del motore a magneti permanenti interni in dq	35
4.3 Regioni di funzionamento del motore brushless anisotropo	37
5. Simulazione degli algoritmi di controllo	41
5.1 Algoritmo di controllo AMK	41
5.2 Modello Simulink del controllo	43
5.2.1 I regolatori PI	45
5.2.2 Funzionamento in MTPA	47
5.2.3 Funzionamento in deflussaggio	53
5.2.4 Modellizzazione dell'inverter	55
5.2.5 La taratura dei regolatori PI	58

5.3	Modello Simulink dello stadio di potenza	62
5.3.1	Algoritmo di Space Vector Modulation	63
5.3.2	Scelta della tecnica di modulazione vettoriale	66
5.3.3	Dimensionamento della capacità di DC-link	73
6.	Implementazione degli algoritmi nel firmware di un microcontrollore	79
6.1	La periferica ePWM	80
6.2	La periferica ADC	86
6.3	Interfaccia per la lettura encoder	92
6.4	Funzionalità di protezione	97
6.5	Accorgimenti per velocizzare l'esecuzione del codice	98
6.6	Collaudo	99
7.	Lavori futuri	107
8.	Conclusioni	109
	Ringraziamenti	111
	Appendice A	112
	Appendice B	113
	Appendice C	114
	Appendice D	115
	Bibliografia	119

Lista delle figure

1.1 Punteggio eventi Formula Student, da [30]	2
1.2 La vettura SG-e 04	3
1.3 powertrain della vettura SG-E 04	4
1.4 Invertitori AMK	5
1.5 Caratteristiche meccaniche dei motori AMK, da [21]	6
2.1 Dispositivi usati nel lavoro di tesi	9
2.2 Microcontrollore [33]	10
2.3 Gate driver 1EDI2004AS [9]	11
2.4 Rappresentazione delle capacità tra i terminali di un IGBT [32]	12
2.5 IGBT sixpack FS380R12A6T4LB [13]	12
2.6 Collegamenti interni dell'IGBT sixpack FS380R12A6T4LB [13]	13
3.1 Gate driver board, versione demo	15
3.2 Collegamenti SPI, da [31]	17
3.3 Funzionamento della SPI [34]	17
3.4 Modulo SPI del microcontrollore TMS320F28379D [16]	18
3.5 Topologia SPI standard [9]	20
3.6 Topologia SPI daisy chain [9]	21
3.7 Trasmissione dati in daisy chain [9]	21
3.8 Struttura del firmware	22
3.9 Segnali tra microcontrollore e gate driver	23
3.10 Stati di funzionamento del gate driver [9]	24
3.11 Domanda e risposta con protocollo SPI [9]	25
3.12 Collegamento in daisy chain [18]	26
3.13 Primo test sulla scheda per singolo gate driver	30
3.14 Segnale di clock misurato con oscilloscopio	30
4.1 Vettore spaziale [19]	32
4.2 Vettore spaziale [19]	33
4.3 Rotore di macchina a magneti permanenti anisotropa [20]	33
4.4 Rotore di macchina a magneti permanenti isotropa [20]	34
4.5 Schema a blocchi della macchina a magneti permanenti interni [20]	36
4.6 Limiti di tensione e corrente nel piano $I_d - I_q$ [20]	38
4.7 Caratteristica meccanica del motore brushless [20]	39
5.1 Algoritmo di controllo dell'azionamento AMK [21]	42
5.2 Algoritmo di controllo dell'azionamento AMK [21]	43
5.3 Algoritmo di controllo sviluppato	44
5.4 Modello Simulink della centralina	45
5.5 Integrazione a tempo discreto [22]	46
5.6 Regolatore PI discreto	46
5.7 Regolatore di velocità	47
5.8 Vettore spaziale di corrente I nel piano dq	48
5.9 Banco prova "lento"	49

5.10 Setup del banco prova	50
5.11 Setup del banco prova	50
5.12 Luogo MTPA, realizzata tramite Matlab	51
5.13 Relazione coppia – corrente, realizzata tramite Matlab	52
5.14 Generazione dei riferimenti di corrente, realizzata tramite Matlab	52
5.15 Regolatore di tensione	53
5.16 Corrente di asse diretto al variare della velocità, da Simulink	55
5.17 Ritardo dovuto all’invertitore	56
5.18 Modello dell’invertitore, realizzata su Simulink	56
5.19 Corrente di asse diretto con compensazione, realizzata con Simulink ..	57
5.20 Schema a blocchi del controllo di motore brushless [23]	58
5.21 Anelli di corrente dopo disaccoppiamento [23]	59
5.22 Anelli di corrente con disaccoppiamento e compensazione FCEM [23]	59
5.23 Controllo della corrente	60
5.24 Corrente di asse diretto in deflussaggio, realizzata con Simulink	61
5.25 Vettori di stato dell’inverter a due livelli [24]	63
5.26 Sintesi di un generico vettore spaziale nel primo settore [25]	64
5.27 Functions che implementano la Space Vector Modulation in Simulink	64
5.28 Modulazione a singolo fronte [6]	67
5.29 Modulazione simmetrica[6]	67
5.30 Modulazione asimmetrica [6]	68
5.31 Modulazione ad aggancio di bus [6]	68
5.32 Schema di single-update PWM [26]	69
5.33 Schema di di double-update PWM [26]	69
5.34 Corrente con SVM simmetrica, realizzata con Simulink	71
5.35 Corrente con SVM asimmetrica, realizzata con Simulink	72
5.36 Coppia erogata con SVM simmetrica, realizzata con Simulink	72
5.37 Coppia erogata con SVM asimmetrica, realizzata con Simulink	73
5.38 Inverter a tensione impressa [27]	74
5.39 Calcolo della corrente efficace nel condensatore in Simulink	76
6.1 Struttura del controllo del motore nel codice	79
6.2 Regolazione della tensione media sul carico mediante PWM [25]	81
6.3 Schema di principio di inverter a due livelli [25]	81
6.4 Rilievo di segnali PWM tramite oscilloscopio	82
6.5 Schema illustrativo dell'Up-Down Count Mode [16]	82
6.6 Controllo dei canali A e B della periferica ePWM [16]	83
6.7 Inserzione del tempo morto [16]	84
6.8 Campionamento sincrono [28]	85
6.9 Sensori letti tramite periferica ADC	87
6.10 Modello dello stadio di ingresso di ADC “Single-ended” [16]	87
6.11 Modello dello stadio di ingresso di ADC “Differential input” [16]	88
6.12 Schema di principio di una sonda ad effetto Hall [35]	88
6.13 Sensore ad effetto Hall HO 150-S-0100 [14]	89
6.14 Stadio di lettura della tensione DC	90
6.15 Stadio di lettura dei termistori	91

6.16 Velocità del motore calcolata dal microcontrollore in fase di debug	94
6.17 Schematico dei collegamenti tra TIDM-1008 e scheda F28379D [29] ...	96
6.18 Scheda TIDM-1008 montata sulla LAUNCHXL-F28379D	96
6.19 Durata delle operazioni	99
6.20 Inverter fornito dall'azienda sponsor	100
6.21 Cavo per alimentazione LV	100
6.22 Cavo di segnale	101
6.23 Scheda millefori	101
6.24 Cavo per la lettura dell'encoder	102
6.25 Pinout del connettore Abs-Enc-1 (J7) sulla scheda TIDM-1008 [29]	103
6.26 Set minimo di variabili da monitorare in debug	103
6.27 Corrente di fase misurata con pinza amperometrica	105
8.1 Prototipo di invertitore assemblato	110
D.1 File Simulink " Control.slx"	115
D.2 File Simulink " Control.slx"	116
D.3 File Simulink " Power_stage.slx"	117
D.4 File Simulink " Power_stage.slx"	118

Lista delle tabelle

1.1 Caratteristiche della cella SLPBA843126	3
1.2 Caratteristiche dell'invertitore KW26-S5-FSE-4Q	5
1.3 Caratteristiche del motore DD5-14-10-POW-18600-17KW11	6
2.1 Caratteristiche dell'IGBT FS380R12A6T4LB	13
3.1 Pinout della periferica SPI	23
3.2 Funzioni usate nella comunicazione con i gate drivers	24
5.1 Gestione dei riferimenti nell'azionamento AMK	43
5.2 Gestione dei riferimenti nell'azionamento progettato	44
5.3 Perdite nel modulo sixpack al variare della frequenza di commutazione	70
5.4 Ripple della tensione DC al variare della frequenza di commutazione	76
5.5 Requisiti per il condensatore di DC-link	77
5.6 Ripple della tensione DC con il valore di capacità scelto	77
6.1 Pinout della periferica ePWM	86
6.2 Pinout della periferica ADC	92
6.3 Collegamenti dell'inverter di prova	102
6.4 Collegamenti del sensore di posizione	103

Sommario

La vettura elettrica, progettata e costruita dalla Divisione Elettrica della squadra di Formula Student (RaceUP Team) che rappresenta l'Università degli Studi di Padova, è mossa da quattro motori sincroni a magneti permanenti, controllati da altrettanti invertitori.

L'obiettivo di questo lavoro di tesi è progettare il firmware per microcontrollore, al fine di controllare un invertitore trifase per uso automotive, nell'ottica di sostituire i dispositivi attualmente in uso.

Il lavoro è articolato in due macrosezioni.

La comunicazione tra il microcontrollore e lo stadio di potenza è realizzato mediante il protocollo di comunicazione SPI. La scrittura di funzioni apposite, in linguaggio C, consente di realizzare una interfaccia in grado di gestire la commutazione degli interruttori di potenza ed eventuali errori che possono sorgere nel funzionamento degli stessi.

Segue lo studio degli algoritmi di controllo delle macchine a magneti permanenti, che sfocia nella produzione di due modelli dell'invertitore in ambiente Simulink. A partire dalle simulazioni, la routine di controllo viene tradotta in codice e questo viene collaudato mediante un invertitore di prova.

Tutte le fasi di progettazione e stesura del codice sono seguite da test. La stretta collaborazione con i progettisti delle schede di potenza ha consentito l'integrazione ottimale del codice con lo stadio di potenza dell'invertitore.

La progettazione e la validazione di un algoritmo di controllo per motore brushless consentirà alla Divisione Elettrica di proseguire con lo sviluppo in proprio degli invertitori di trazione. La documentazione redatta durante il lavoro di tesi, inoltre, contribuirà alla formazione di un know-how in fatto di elettronica di potenza e controllo dei motori elettrici.

Introduzione

In una vettura stradale mossa da motori elettrici è fondamentale la presenza di convertitori di frequenza, o invertitori, che consentono alla stessa di accelerare e frenare, variando la velocità di rotazione delle macchine elettriche mediante la modifica dell'ampiezza e della frequenza delle correnti assorbite.

Da anni la squadra corse RaceUP Team utilizza invertitori prodotti da terzi e, pertanto, è limitata nell'ottimizzazione degli spazi e delle masse a bordo della vettura. Oltretutto, essendo prodotti da esterni, gli invertitori si presentano come dei sistemi chiusi, dei quali non è ben chiaro il funzionamento. In caso di danni ai dispositivi, infatti, è sempre necessario affidarsi al produttore per le riparazioni. Infine, negli anni si sono spesso verificati problemi di compatibilità elettromagnetica tra i convertitori ed i restanti sistemi elettrici presenti in vettura, riducendone drasticamente l'affidabilità complessiva.

Con questo lavoro di tesi, l'autore si è prefissato di risolvere o, quantomeno, di mitigare, alcune delle sopracitate problematiche.

A partire dalla stagione 2019/2020, nel reparto Powertrain della Divisione Elettrica del RaceUP Team si è cominciato a discutere della possibilità di sviluppare in proprio gli invertitori di trazione. Da anni, le squadre più esperte di Formula Student progettano quasi tutti i componenti della loro vettura in proprio, riuscendo così ad ottenere i massimi risparmi in termini di ingombro e masse, con un chiaro vantaggio in termini di prestazioni in pista.

La topologia scelta per gli invertitori è a tensione impressa, a due livelli, con IGBTs al silicio come interruttori di potenza. Nonostante il trend sia il passaggio ai materiali semiconduttori a banda larga, quali il carburo di silicio (SiC) ed il nitruro di gallio (GaN), confrontando le esigenze della squadra corse con le tecnologie disponibili, si è ritenuto opportuno scegliere una tecnologia consolidata, ottimizzata per la frequenza di commutazione scelta. Oltretutto, si è tenuto conto del fattore economico, dato che il costo di moduli SiC MOSFET è molto maggiore di moduli IGBT di taglia simile. Dovendo montare in vettura ben quattro convertitori, la spesa complessiva diventa rilevante.

A maggio 2020 lo studio di fattibilità si è concretizzato nel primo contatto con aziende attive nel settore dell'elettronica di potenza. In luglio 2021 è cominciato il primo lavoro di tesi sugli invertitori sviluppati in proprio, da parte di un laureando in Ingegneria Elettronica.

Questo è quindi il secondo lavoro di tesi sull'argomento.

L'auspicio dell'autore è, da una parte, di riuscire a realizzare un prototipo funzionante di convertitore, dall'altra c'è la volontà di dare avvio ad un progetto destinato a durare negli anni a venire. Fin dai primi studi di fattibilità, infatti, è emerso che sarebbero stati necessari anni per passare dallo sviluppo di un prototipo all'implementazione in vettura di invertitori sviluppati in proprio.

Gli strumenti usati per svolgere questo lavoro sono principalmente software, per simulare e per programmare il microcontrollore, ma non manca la parte pratica del collaudo.

Nel primo capitolo viene presentato in modo sintetico il mondo della Formula Student, con una particolare attenzione alla squadra dell'Università degli Studi di Padova, il RaceUP Team. Segue una descrizione della vettura elettrica, soprattutto dei sistemi elettrici di trazione.

Il secondo capitolo descrive i principali strumenti, sia dispositivi che programmi, coinvolti in questa tesi.

Il terzo capitolo affronta la comunicazione tra il microcontrollore e lo stadio di potenza. Sono trattati nel dettaglio sia le periferiche del microcontrollore che i gate drivers. Sono inoltre riportate parti del codice prodotto.

Lo studio degli algoritmi di controllo delle macchine a magneti permanenti è affrontato nel quarto capitolo. L'applicazione della teoria viene descritta nel quinto capitolo, mediante le realizzazioni di simulazioni al computer, assieme alla descrizione dettagliata degli inverter attualmente in uso nella vettura.

Il sesto capitolo presenta la traduzione in codice degli algoritmi di controllo, seguita dal riporto dei risultati del collaudo con un inverter di prova.

Vengono riportate, infine, le riflessioni sul lavoro finora svolto e sui prossimi passi da compiere, prima di sviluppare il modello definitivo che potrà essere montato a bordo della vettura.

1 La Formula Student

1.1 La competizione di Formula Student

La competizione di Formula Student (anche chiamata Formula SAE) nacque nel 1981, dall'idea di un professore all'Università del Texas. Originariamente aperta solo ad università statunitensi, ora questa attività coinvolge università di tutto il mondo.

Per ogni ateneo, un gruppo di studenti ogni anno progetta e costruisce una vettura monoposto da competizione, per competere nelle categorie Combustion (CV) o Electric (EV), per vetture con motore a combustione interna o elettrico rispettivamente, o Driverless (DV), per vetture a guida autonoma, quindi prive di pilota, che possono avere motorizzazione di uno dei due tipi precedenti.

Ogni anno, nei mesi estivi, si svolgono le competizioni interuniversitarie che servono per validare la qualità della progettazione dei prototipi. Formula Student Germany, Formula Student Austria, Formula Student China sono solo alcuni degli eventi che si svolgono con il patrocinio di numerose aziende, molte ma non tutte legate al settore automotive.

Durante gli eventi, una giuria composta da professionisti dell'industria valuta la bontà della vettura progettata dagli studenti, ma anche la loro preparazione nell'ambito della gestione delle risorse, pianificazione dei lavori, capacità e chiarezza espositive.

Le prove da superare sono divise in dinamiche e statiche. Le prime mirano a verificare, tramite le gare di Skid Pad, Acceleration, Autocross ed Endurance, la validità della progettazione e le performance della vettura presentata. Le seconde, invece, servono per testare le capacità di presentazione, gestione delle risorse e project management all'interno della squadra universitaria.

Mentre la prova di Skid Pad fornisce indicazioni sulla capacità di un prototipo di essere maneggevole nei tratti curvi della pista, per eccellere nell'Acceleration è sufficiente che la vettura abbia buone capacità di accelerazione sul rettilineo da 75 m. L'Autocross consiste in una corsa lunga al massimo 1 km, dotata di numerose curve e rettilinei brevi. L'Endurance, infine, è forse l'evento dinamico più difficile da superare, poiché ogni squadra deve superare 22 km con una sola carica dell'accumulatore ad alta tensione. Al termine di questa prova, inoltre, occorre dimostrare ai giudici che la vettura è in condizioni sicure, ovvero è integra ed in condizioni di essere riaccesa. Viene inoltre valutato il consumo energetico, premiando i prototipi più efficienti.

Per quanto concerne gli eventi statici, essi sono tre. Il Business Plan Presentation (BPP) ed il Cost and Manufacturing Event sono di carattere economico. Il loro scopo è dimostrare quanto una singola squadra è in grado di gestire in maniera dettagliata

un elenco dei pezzi presenti nella vettura e delle lavorazioni necessarie per ottenerli, stimando i relativi costi (Cost), oppure valutare la capacità di attrarre gli investitori, tramite il BPP.

L'Engineering Design, infine, consiste nel presentare in modo conciso una descrizione quanto più dettagliata possibile della vettura.

L'assegnazione dei punti è ripartita come indicato nella seguente tabella.

	CV & EV	DV
Static Events:		
Business Plan Presentation	75 points	75 points
Cost and Manufacturing	100 points	100 points
Engineering Design	150 points	300 points
Dynamic Events:		
Skid Pad	75 points	75 points
Acceleration	75 points	75 points
Autocross	100 points	100 points
Endurance	325 points	-
Efficiency	100 points	75 points
Trackdrive	-	200 points
Overall	1000 points	1000 points

Fig. 1.1 - Ripartizione dei punti assegnati per ogni prova statica e dinamica, nelle varie categorie. Al termine di tutte le prove viene stilata una classifica in base alla somma dei punteggi ottenuti, per incoronare la squadra vincente

La partecipazione alla Formula Student è vista, da parte di molte aziende multinazionali che sponsorizzano le competizioni e le squadre, come una attività formativa essenziale per studenti che aspirano a lavorare nel settore automotive e non. Lo scopo è favorire la transizione dei giovani ingegneri dal mondo accademico a quello lavorativo, attraverso l'applicazione pratica delle nozioni apprese durante il percorso di studi, in combinazione con l'attività di progettazione di parti meccaniche, gestione delle risorse, business planning, project management, ricerca di fondi e soft skills, quali la capacità di lavorare a scadenze e lavorare in squadra.

1.2 Il RaceUP Team e la Divisione Elettrica

L'Università degli Studi di Padova, nelle competizioni nazionali ed internazionali di Formula Student, è rappresentata dal RaceUP Team.

I membri della squadra sono ripartiti circa equamente in Divisione Electric e Divisione Combustion, le quali hanno come obiettivo la costruzione rispettivamente di un prototipo con propulsione elettrica ed uno con motore a combustione interna.

Mentre le vetture a propulsione convenzionale vengono progettate e costruite dal 2006, i primi lavori sulla vettura elettrica risalgono al 2015. Il primo prototipo elettrico (ORIGIN-E 2016) è stato presentato nel 2016. Attualmente, è stato ultimato il quinto veicolo, il quale porta il nome di SG-E 04.

1.3 SG-e 04

La SG-E 04 è la vettura progettata durante la stagione 2019/2020 ma poi costruita soltanto nella stagione 2020/2021, a causa della pandemia di Covid-19.

Si tratta di una monoposto con monoscocca ed appendici aerodinamiche in fibra di carbonio, progettate e realizzate in casa.

Le sospensioni sono di tipo push-rod. Il veicolo è mosso da un apparato di propulsione completamente elettrico, composto da un accumulatore ad alta tensione, quattro inverter ed altrettanti motori sincroni a magneti permanenti. I motori sono montati direttamente nel gruppo ruota (“hub-motors”), tramite un riduttore di velocità epicicloidale, anch’esso progettato internamente. La velocità massima così raggiungibile è pari a 119 km/h. La potenza complessivamente sviluppabile dai quattro motori è pari a 140 kW, mentre la massa complessiva della vettura ammonta a 215 kg.

Convertitori e motori saranno descritti nel dettaglio nel prossimo paragrafo. Tutto l’apparato propulsivo è mantenuto in temperatura da circuiti di raffreddamento ad acqua.



Fig. 1.2 - La vettura SG-e 04

L’accumulatore è composto da celle ai polimeri di litio, modello SLPBA843126, prodotte da Shenzhen Melasta Battery Co. Ltd. Le loro caratteristiche salienti sono riportate nella tabella sottostante.

Tabella 1.1

Capacità nominale [mAh]	6350
Tensione nominale [V]	3.7
Tensione massima [V]	4.2

Tensione minima [V]	3
Corrente continua di scarica [A]	105
Corrente massima di scarica (≤ 2 sec) [A]	127
Temperatura di funzionamento [$^{\circ}$ C]	-20 \div 60
Impedenza interna @ 1kHz [mOhm]	≤ 2
Massa [g]	131 \pm 2

La configurazione dell'accumulatore è 2p144s, per un totale di 288 celle. Ne consegue che la tensione nominale del pacco è 532.5 V, quella massima 604.8 e quella minima 432 V. I limiti pratici della tensione, tuttavia, sono 600 V massimi da regolamento Formula Student e circa 500 V minimi per questioni di sicurezza, dato che la curva di scarica della cella, sotto i 3.5 V, diventa molto ripida. L'accumulatore alimenta tutto il powertrain, quindi i convertitori devono essere in grado di lavorare entro tutto l'intervallo dei valori di tensione lato continua. Il regolamento Formula Student, inoltre, limita la potenza massima in uscita dal pacco batteria a 80 kW. Il raffreddamento delle celle è a convezione forzata tramite aria.

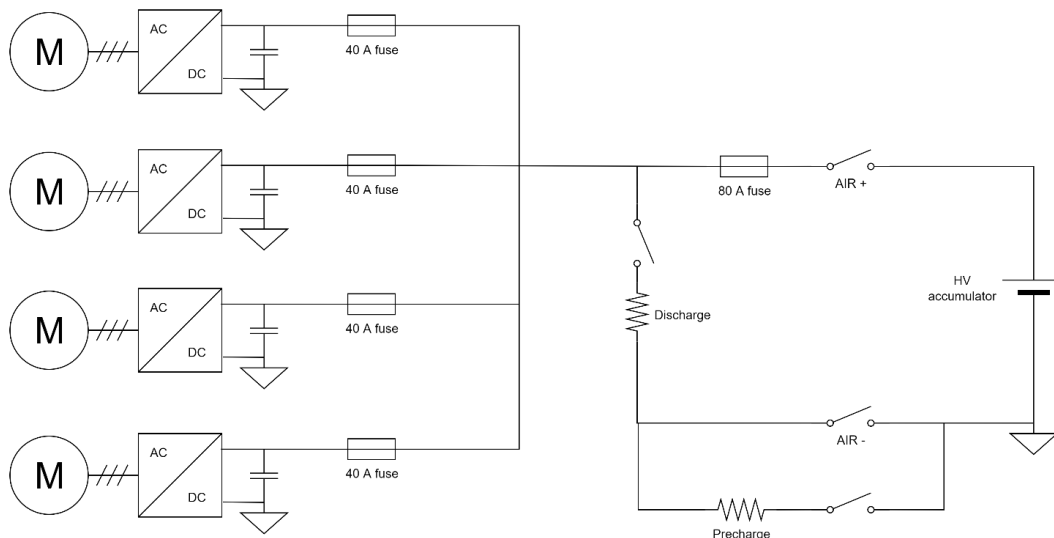


Fig. 1.3 - Schematico del powertrain della vettura SG-E 04

Accanto al circuito ad alta tensione, sono presenti due circuiti a bassa tensione a 24 V e a 12 V, galvanicamente isolati dal primo. Questi circuiti alimentano tutti i dispositivi elettronici quali sensori, schermo del pilota, circuito LV degli inverters ed altri.

1.4 AMK Racing Kit

L'apparato di propulsione della vettura è composto interamente dal "Formula Student Electric Racing Kit", venduto da AMK. Il kit comprende quattro inverters ed altrettanti motori.

Gli inverter, modello KW26-S5-FSE-4Q, sono della tipologia a due livelli, a tensione impressa. Le caratteristiche salienti sono riportate in tabella.

Tabella 1.2

Tecnologia	IGBT
Frequenza di commutazione [Hz]	8000
Frequenza massima in uscita [Hz]	1666.67
Potenza nominale [kVA]	26
Tensione DC [V]	250 ÷ 720
Corrente nominale [A rms]	43
Corrente massima [A rms]	105
Capacità in ingresso [uF]	75
Raffreddamento	Liquido

La massa complessiva dei quattro convertitori è di circa 10 kg, dopo un alleggerimento della piastra di raffreddamento eseguito dal team sul prodotto di fabbrica. Probabilmente tale valore potrebbe essere abbassato ulteriormente, sostituendo la struttura metallica portante con un materiale più leggero ma conduttivo per preservare i collegamenti a massa delle schede. Per quanto riguarda gli ingombri, risulta difficile immaginare di ridurre il volume dei convertitori, a causa sia della struttura metallica portante, sia della già elevata densità di componenti che non possono essere avvicinati ulteriormente senza rendere poi difficoltosa la manutenzione degli stessi, in caso di guasto o sostituzione del cablaggio.

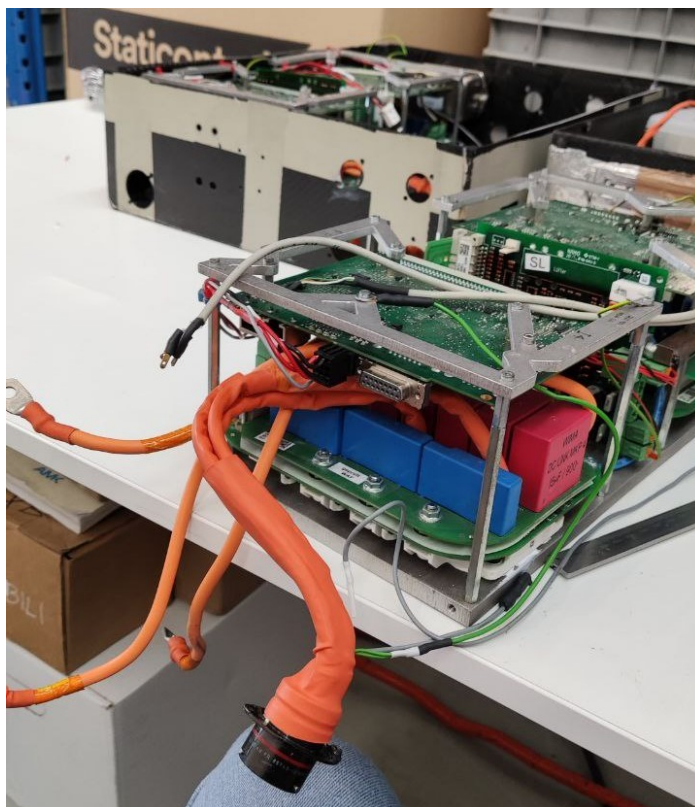


Fig. 1.4 - Coppia di convertitori. Sulla sinistra si scorge la scatola di carbonio nella quale vengono alloggiati le due coppie, per un totale di quattro inverter

I motori, modello DD5-14-10-POW-18600-17KW11, sono del tipo sincro a magneti permanenti interni (IPM). Le caratteristiche salienti sono qui riportate.

Tabella 1.3

Numero di poli	10
Coppia nominale [Nm]	9.8
Coppia massima [Nm]	21
Potenza massima [kW]	35
Velocità massima [rpm]	20000
Corrente nominale [A rms]	41
Corrente massima [A rms]	105
Tensione nominale [V rms]	350
Induttanza sincrona di asse diretto L_d [mH]	0.12
Induttanza sincrona di asse in quadratura L_q [mH]	0.24
Collegamento avvolgimenti	triangolo
Resistenza tra due terminali [Ohm]	0.135
Flusso concatenato dovuto ai magneti Λ_{mg} [V·s]	0.0296
Momento d'inerzia J [kg · m ²]	$2.74 \cdot 10^{-4}$
Massa [kg]	3.55
Raffreddamento	Liquido

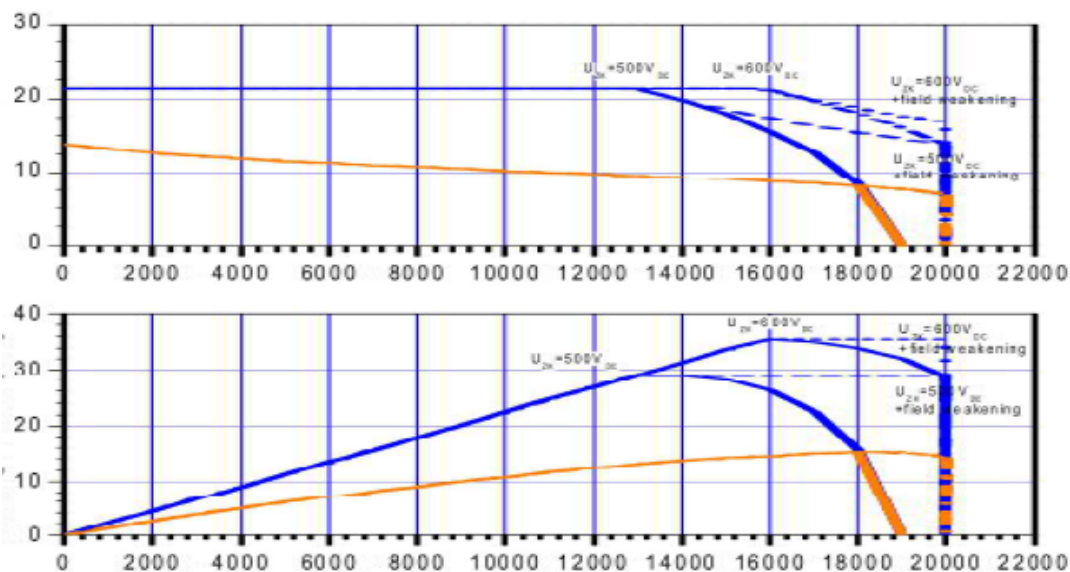


Fig. 1.5 - Caratteristiche coppia-velocità (sopra) e potenza-velocità (sotto) dei motori

Le macchine a magneti permanenti ben si prestano ad essere utilizzate come motori per vetture da corsa, grazie alle loro elevate compattezza e densità di potenza. L'alimentazione tramite inverter consente di ottenere il funzionamento a velocità variabile, grazie al controllo della frequenza di uscita ed elevate prestazioni dinamiche, tramite un preciso controllo della coppia erogata.

Essendo l'intero l'azionamento prodotto dalla stessa azienda, non esistono problemi di compatibilità tra l'elettronica di potenza ed i motori. Allo stesso tempo, il kit si presenta come un sistema chiuso poiché non sono disponibili molte informazioni; i

datasheets forniscono solo una descrizione incompleta. Gli inverters si possono programmare tramite il software proprietario della AMK, Aipex Pro, tuttavia il numero di parametri modificabili è esiguo.

2 Strumenti usati

2.1 Hardware

I dispositivi utilizzati per realizzare il prototipo di inverter in questo lavoro di tesi sono i seguenti:

- Evaluation board con microcontrollore LAUNCHXL-F28379D LaunchPad della Texas Instruments
- Scheda TIDM-1008 “PositionManager BoosterPack” della Texas Instruments
- Gate driver 1EDI2004AS con booster 1EBN1001AE, prodotti da Infineon Technologies
- IGBT sixpack FS380R12A6T4LB, prodotto da Infineon Technologies

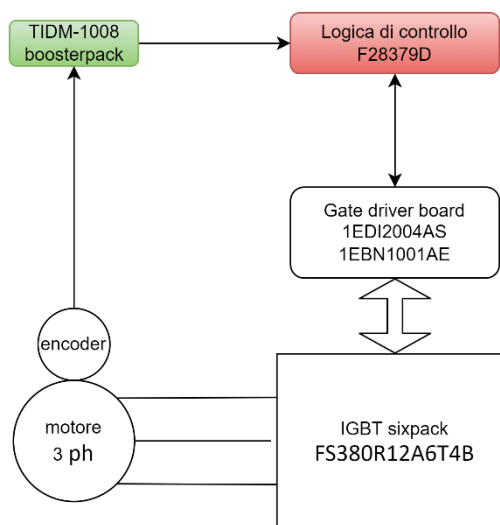


Fig. 2.1 - Schema dei dispositivi coinvolti nel lavoro di tesi

2.1.1 Microcontrollore

La C2000™ LAUNCHXL-F28379D LaunchPad™ è una scheda low-cost per lo sviluppo di progetti basati su dispositivi della serie Texas Instruments Delfino™ F2837xD. È dotata di un dispositivo JTAG per il debug, inoltre l'interfaccia USB fornisce una connessione seriale tra microcontrollore e pc. È così possibile alimentare la scheda direttamente dal pc dal quale la si programma.

La scheda dispone, tra l'altro, di:

- Interfaccia USB galvanicamente isolata per programmare e debuggare
- microcontrollore TMS320F28379D
- due LEDs accessibili dall'utente
- interfaccia CAN con transceiver integrato
- quattro moduli separati per ADC a 12 o 16 bit
- quattro demodulatori Sigma Delta
- alimentazione esterna a 3.3 o 5 V
- moduli SPI, I2C
- dodici moduli PWM con una unità Configurable Logic Block (CLB) che può essere usata per la generazione dei tempi morti.

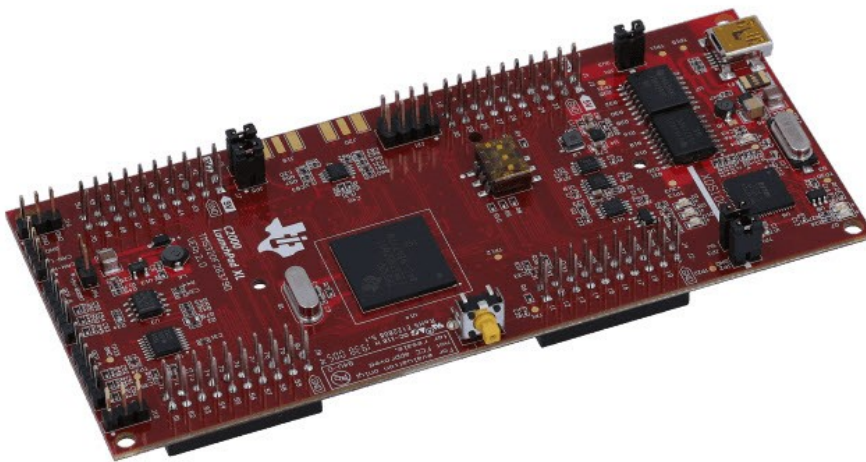


Fig. 2.2 - LAUNCHXL-F28379D LaunchPad

I microcontrollori della serie TMS320F2837xD sono unità floating-point a 32 bit, progettati specificamente per controlli avanzati a catena chiusa come azionamenti elettrici industriali, inverter fotovoltaici, veicoli e trasporti elettrici, lettura di sensori ed elaborazione di segnali.

Il microcontrollore TMS320F28379D è dual core, con una frequenza massima di 200 MHz. La logica di input/output lavora a 3.3 V.

Accanto alle due CPU, sono presenti due coprocessori (CLA) che possono operare indipendentemente dalle prime, portando a quattro il numero di core effettivamente utilizzabili.

Il microcontrollore può essere programmato tramite Code Composer Studio™ IDE. La routine di controllo del motore e la comunicazione SPI viene tutta gestita dal microcontrollore.

2.1.2 Gate Driver



Fig. 2.3 – Gate driver 1EDI2004AS

L' 1EDI2004AS è un gate driver progettato specificamente per pilotare singoli IGBTs (o anche power MOSFETs) con potenze superiori a 5 kW e tensioni fino a 1200 V. Il driver è composto da due chip separati, in modo da ottenere separazione galvanica tra logica al primario e secondario ad alta tensione. La logica a bassa tensione funziona a 5 V, motivo per cui occorre adeguare i segnali a 3.3 V provenienti dal microcontrollore.

Il driver è dotato di una interfaccia SPI a 16 bit, con baud rate fino a 2 Mbaud e la possibilità di collegare più dispositivi in daisy chain. Tramite SPI è possibile scrivere nei registri del dispositivo, leggere gli errori ed abilitare la commutazione dell'IGBT. I segnali pwm, invece, vengono direttamente forniti in ingresso all'integrato.

Tra le funzionalità più importanti vi sono:

- monitoraggio della desaturazione
- protezione da sovraccorrente
- possibilità di selezionare lo spegnimento a due livelli (Two-Level Turn-Off) dello switch
- pin di output NFLTA e NFLTB per una rapida notifica al microcontrollore, in caso di errori

Il compito del gate driver consiste nel trasformare i segnali logici, quindi a bassa potenza, in uscita dal microcontrollore in tensioni con le quali accendere e spegnere il gate dell'IGBT.

Generalmente quest'ultimo viene portato in conduzione applicando una tensione tra gate ed emettitore v_{GE} positiva, mentre per interdire il transistor si applica una v_{GE} negativa. La v_{GE} fa scorrere una corrente che va a caricare o scaricare le capacità tra Gate e Collettore (C_{GC}) e tra Gate ed Emettitore (C_{GE}).

Oltre a questo tipo di alimentazione, detta bipolare, il driver supporta anche una alimentazione unipolare (accensione con v_{GE} positiva e spegnimento con v_{GE} nulla), con la quale però aumenta la probabilità di accensioni indesiderate dello switch, a causa di accoppiamenti parassiti.

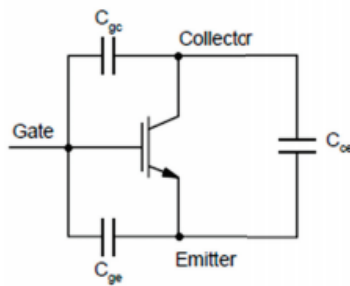


Fig. 2.4 – Rappresentazione delle capacità tra i terminali di un IGBT

Il gate driver, inoltre, separa galvanicamente la logica di controllo in bassa tensione dal circuito di potenza, provvedendo a gestire tutte le funzionalità di sicurezza.

Va notato che l'1EDI2004AS può erogare o assorbire al massimo 1 A, per cui si è scelto di abbinarlo ad un amplificatore di corrente del tipo 1EBN1001AE, il quale è in grado di erogare fino a ± 15 A.

Nell'inverter oggetto di questa tesi l'alimentazione del gate è bipolare con $+15/-9$ V, proveniente da un convertitore DC/DC isolato, che viene collegato alla linea a 24 V presente nella vettura.

2.1.3 IGBT

Il transistor scelto per l'inverter del team di Formula Student è un IGBT, modello FS380R12A6T4LB. Nonostante la tecnologia emergente sia quella dei MOSFETs al carburo di silicio (SiC), per questa prima versione di inverter si è scelta la tecnologia ormai consolidata degli IGBTs, principalmente per motivi economici.

Per una maggiore compattezza si è optato per un package "sixpack", ovvero un unico componente che raccoglie al suo interno tutti i sei interruttori necessari per realizzare un inverter a due livelli. Prevedendo di usare dei trasduttori di corrente ad effetto Hall, la versione "Long Tab" con i terminali di fase (busbars) leggermente più lunghi dovrebbe permettere agevolmente l'integrazione dei sensori.



Fig. 2.5 - Aspetto esteriore dell'IGBT sixpack. Si noti la struttura con alette "pin fin" per lo smaltimento della potenza dissipata sotto forma di calore

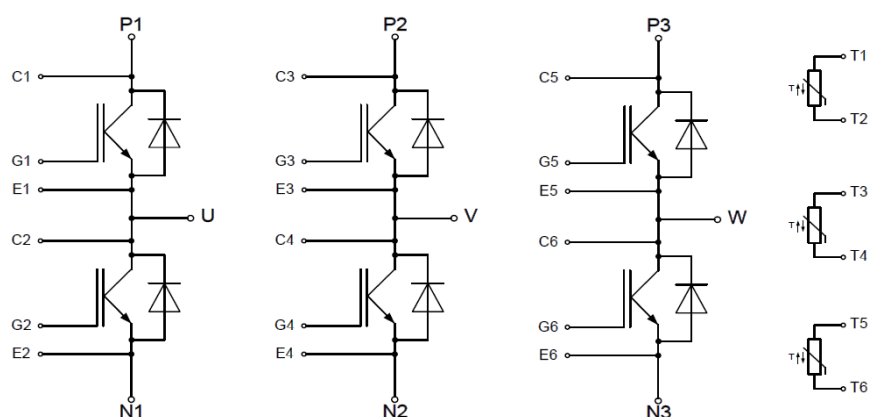


Fig. 2.6 - Collegamenti interni del modulo IGBT. Per ogni fase è disponibile un termistore che consente di monitorarne la temperatura

Tabella 2.1

Massima tensione di blocco v_{CE} [V]	1200
Corrente DC al collettore [A]	250
Corrente ripetitiva al collettore [A]	760
Tensione v_{GE} [V]	± 20
Induttanza parassita [nH]	8
Temperatura operativa [$^{\circ}\text{C}$]	$-40 \div 150$
Massa [g]	720

Si noti come la tensione di blocco massima sia ben superiore alla tensione di lavoro DC prevista ($500 \div 600$ V). È buona norma, quando le commutazioni sono di tipo “hard”, sovradimensionare in tensione l’interruttore di potenza poiché, durante l’apertura dello stesso, si possono verificare delle sovratensioni dovute ad induttanze parassite, come quella del modulo stesso e di tutti i collegamenti.

2.2 Software

Per la programmazione del microcontrollore si è utilizzato Code Composer Studio (CCS, v10.1.0), un software IDE per programmare e debuggare schede della Texas Instruments. Accanto a CCS è necessario installare i pacchetti C2000Ware (v3.02.00.00), un pacchetto per microcontrollori della serie C2000 contenente librerie, esempi e drivers e controlSUITE, un altro pacchetto necessario per la parte riguardante la lettura dell’encoder.

Gli algoritmi di controllo, invece, sono stati simulati tramite Matlab R2020b e la corrispondente versione di Simulink.

Per qualsiasi futuro lavoro sull’inverter, si consiglia di usare, almeno per CCS, la stessa versione indicata sopra, onde evitare i frequenti problemi di compatibilità dei compilatori.

3 La comunicazione con i gate drivers

drivers

La prima parte di questo lavoro di tesi riguarda lo sviluppo della routine di comunicazione tra scheda F28379D e gate driver 1EDI2004AS, tramite il protocollo SPI.

Questa parte si è svolta in stretta collaborazione con un altro tesista, laureando in ingegneria elettronica, il quale si è occupato della parte hardware, cioè della progettazione delle schede elettroniche.

Data la poca esperienza, da parte dei membri del team RaceUP, nell'ambito di elettronica di potenza, si è scelto di progettare e costruire una prima scheda, adatta ad alloggiare un singolo gate driver, dotata di tutti i componenti necessari per consentire la comunicazione SPI col microcontrollore della Texas Instruments. Su questa scheda sono stati eseguiti tutti i test, volti ad implementare una interfaccia basilare che consente di programmare i registri del driver, leggere eventuali errori ed abilitare lo stesso a commutare un IGBT.

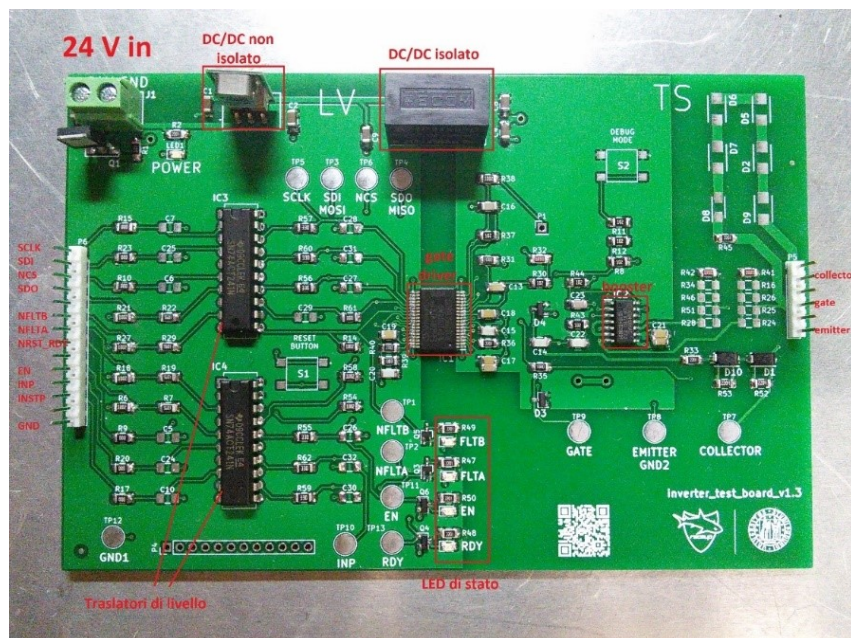


Fig. 3.1 – Scheda usata per lo sviluppo della comunicazione SPI con gate driver della Infineon

La scheda è divisa in due parti: quella sinistra a bassa tensione, galvanicamente isolata dallo stadio di potenza e quella a destra che è direttamente collegata a collettore, emettitore e gate dell'IGBT.

L'alimentazione proviene da un alimentatore esterno a 24 V. Sulla scheda poi dei convertitori DC/DC provvedono ad abbassare il livello a 5 V per il primario e +15/-9 V

per il secondario. Il secondo DC/DC è isolato, per soddisfare ai requisiti di sicurezza che impongono una separazione galvanica tra circuito di alta e di bassa tensione. I livelli di tensioni forniti da questo convertitore alimentano il secondario del gate driver ma sono anche usati per accendere (+15 V) e spegnere (-9 V) l'interruttore allo stato solido.

Al primario sono presenti quattro led per facilitare il debugging. Nel normale funzionamento della scheda solo il led RDY dovrebbe essere acceso se l'IGBT non commuta, se invece commuta dovrebbero essere accesi sia RDY che EN. FLT B e FLTA notificano la presenza di un errore.

Si noti, inoltre, come siano presenti due traslatori di livello, realizzati con i buffers SN74ACT241N. Questi sono necessari dal momento che il microcontrollore Texas lavora con logica 0-3.3 V, mentre il driver con logica 0-5 V. I segnali in uscita dal microcontrollore vengono elevati a 5 V, quelli in ingresso provenienti dalla scheda vengono invece abbassati a 3.3 V tramite partitori di tensione ma passano comunque attraverso i buffers per realizzare disaccoppiamento, poiché si è notato che i partitori esterni possono interferire con eventuali resistenze di pull-up o pull-down presenti all'interno del driver.

Nel seguito verrà fornita una descrizione basilare del protocollo SPI, alla quale seguirà l'analisi dettagliata della routine di comunicazione sviluppata in ambiente Code Composer Studio.

3.1 Il protocollo SPI

Il Serial Peripheral Interface (SPI) è un sistema di comunicazione seriale tra microcontrollori o tra microcontrollore e circuiti integrati, inizialmente ideato dalla Motorola.

L'interfaccia SPI è di tipo sincrono. Questo significa che il dispositivo ricevente, durante la trasmissione, si sincronizza con quello trasmettitore per campionare il segnale ricevuto in corrispondenza dei singoli bit. La trasmissione dei dati avviene contemporaneamente in entrambe le direzioni, essendo la SPI una trasmissione seriale di tipo full-duplex.

I dispositivi coinvolti sono sempre un Master ed uno o più Slaves, dove il primo controlla il bus, emette il segnale di clock, decide quando iniziare e terminare la comunicazione.

Questo tipo di trasmissione necessita di cinque connessioni:

- 1) il segnale di clock, ad onda quadra con duty-cycle del 50%, scandisce la trasmissione di bit, per cui più la sua frequenza è elevata, più velocemente vengono trasferiti i dati
- 2) il segnale MOSI (Master Output Slave Input) o SDO (Serial Data Output) contiene le informazioni trasmesse dal Master allo Slave
- 3) il segnale MISO (Master Input Slave Output) o SDI (Serial Data Input) contiene le informazioni trasmesse dallo Slave al Master

- 4) il Chip Select viene portato a zero dal Master per comunicare allo Slave che è in corso la trasmissione, quindi è attivo basso
- 5) il riferimento dei potenziali (GND) è indispensabile, dato che i livelli logici “alto” e “basso” di tutti i dispositivi vanno riferiti ad una tensione comune

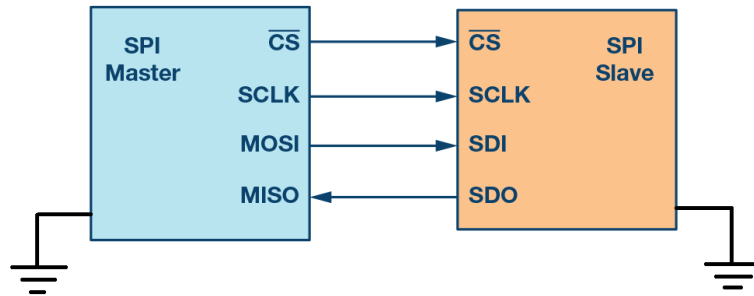


Fig. 3.2 – Schema di principio dei collegamenti tra un Master ed uno Slave

L'immagine sottostante è esplicativa del funzionamento della SPI. Quando il Master decide di cominciare la trasmissione, Chip Select (SS in figura) viene portato basso. Contemporaneamente il clock (SCK in figura) comincia a oscillare alla frequenza prestabilita, consentendo allo Slave di sincronizzarsi per ricevere le informazioni tramite il segnale MOSI. Contemporaneamente alla ricezione, lo Slave può trasmettere a sua volta al Master tramite il segnale MISO. Per terminare la trasmissione, il Master porta a 1 il Chip Select.

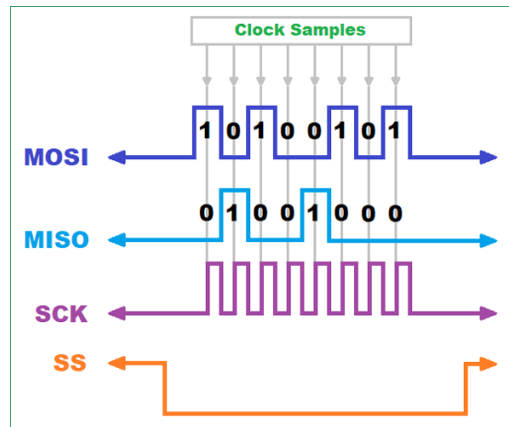


Fig. 3.3 – Funzionamento della SPI

3.2 Il modulo SPI del microcontrollore TMS320F28379D

Il microcontrollore è dotato di una periferica SPI con tre diversi moduli (indicati come A, B,C). La comunicazione con i gate drivers sfrutta il modulo A, in quanto il B viene già utilizzato per la lettura dell'encoder.

Il modulo consente la trasmissione di parole (sequenze di bit) di lunghezza variabile, da 1 a 16 bit.

È possibile scegliere tra quattro diversi schemi di clocking, ma quello selezionato in questo lavoro di tesi è “Rising edge without phase delay”: la trasmissione dati avviene sul fronte di salita del segnale di clock e la ricezione sul fronte di discesa.

La massima frequenza di clock consentita dal microcontrollore è pari a 50 MHz, mentre per valori superiori occorre abilitare l’High-Speed Mode. La SPI può essere configurata per generare interrupts, tuttavia non se ne è fatto uso.

Di interesse particolare è la modalità FIFO (First In First Out), che consente di gestire l’invio e la ricezione di più parole durante la stessa trasmissione, ad esempio quando si fa uso della topologia Daisy Chain.

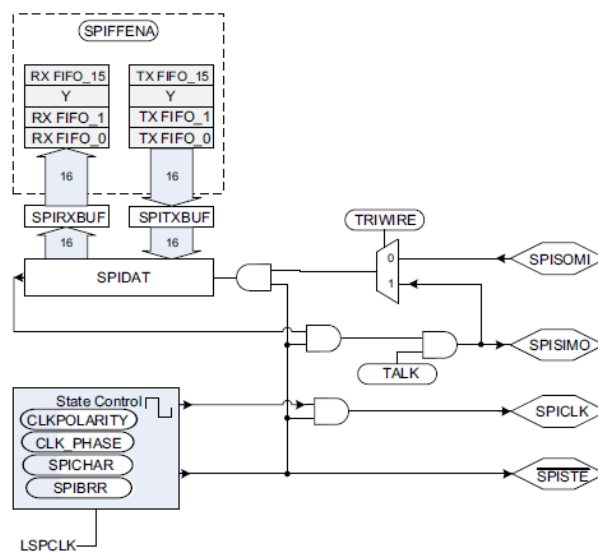


Fig. 3.4 – Diagramma a blocchi del modulo SPI configurato come Master

Come riportato dall’immagine superiore, i registri SPITXBUF e SPIRXBUF agiscono da buffers tra i FIFO di trasmissione e ricezione ed il registro di scorrimento SPIDAT, nel quale transitano tutti i dati in ingresso ed uscita.

Ad ogni trasmissione, se nel registro SPITXBUF è presente un dato valido, questo viene copiato in SPIDAT per essere inviato, a partire dal bit più significativo (Most Significant Bit, MSB). Si può immaginare il MSB come il bit più a sinistra in SPIDAT. Ad ogni periodo di clock, il MSB viene trasmesso, facendo traslare tutta la parola verso sinistra. Contemporaneamente, i bit ricevuti riempiono SPIDAT transitando attraverso il suo LSB, ovvero il bit più a destra. A trasmissione conclusa, il contenuto di SPIDAT, che costituisce la parola ricevuta, viene trasferito in SPIRXBUF, pronto per essere letto dalla CPU. Si sottolinea come la lettura di SPIRXBUF non sia opzionale: la lettura equivale a fare “pulizia” del buffer, svuotandolo e rendendolo quindi pronto per essere riempito dal nuovo dato. Si è verificato che, se non si esegue la lettura di SPIRXBUF dopo ogni trasmissione, si possono avere malfunzionamenti del modulo SPI.

Dato che in questa tesi il microcontrollore agisce sempre da Master, “decidendo” quando comincia e quando termina ogni trasmissione, nel seguito si tralascia il funzionamento da Slave del modulo SPI. In modalità Master (MASTER_SLAVE = 1), la SPI fornisce il clock sul pin SPICLK per tutta la durata della trasmissione. I dati sono

trasmessi dal pin SPISIMO e ricevuti dal pin SPISOMI. La frequenza del clock e, quindi, la velocità di trasmissione, si imposta scrivendo nel registro SPIBRR.

Nel seguito si riporta il codice per la trasmissione di una singola parola.

```
Spi_xmit(a); // salvo a nel registro
SPITXBUF
while(SpiaRegs.SPIFFRX.bit.RXFFST !=1) { } // aspetto la ricezione dei
dati
a = SpiaRegs.SPIRXBUF; // pulisco SPIRXBUF
```

Per superare il limite della trasmissione di una parola per volta è possibile utilizzare la modalità FIFO, che viene abilitata ponendo pari a 1 il bit SPIFFENA nel registro SPIFFTX.

Il FIFO è dotato di un gruppo di 16 buffers di trasmissione e 16 di ricezione, ognuno capace di contenere una parola lunga al massimo 16 bit. È quindi possibile scrivere 16 parole diverse nei registri FIFO di trasmissione, sarà poi il microcontrollore a gestire il loro invio.

Attraverso le funzioni **SPI_writeDataBlockingFIFO** e **SPI_readDataBlockingFIFO** è possibile scrivere dati rispettivamente nel buffer di trasmissione e leggere dati da quello di ricezione. Entrambe le funzioni sono disponibili all'interno dell'header spi.h, già disponibile nella libreria del sistema di sviluppo. La modalità FIFO si rende necessaria dal momento che il microcontrollore deve comunicare con tanti gate drivers quanti sono gli IGBTs da controllare, ovvero sei.

Nel seguito si riportano le righe di codice che stanno alla base della trasmissione via FIFO, trasmissione che coinvolge un numero di dispositivi pari a `NUMBER_OF_GATE_DRIVERS`.

```
Int i;
for (i = 0; i < NUMBER_OF_GATE_DRIVERS; i++) // ripeto per N
volte
{
    SPI_writeDataBlockingFIFO(SPIA_BASE, a); // scrivo nel FIFO
N-esimo
}
// Block until data is received and then return it
for (i = 0; i < NUMBER_OF_GATE_DRIVERS; i++) // ripeto per N
volte
{
    rData[i] = SPI_readDataBlockingFIFO(SPIA_BASE); // leggo dal FIFO
N-esimo
}
```

La peculiarità del funzionamento in daisy chain, descritto nel paragrafo successivo, sta nel fatto che il Chip Select deve stare basso durante tutta la trasmissione, ovvero finché non sono state inviate tutte le parole a tutti i gate drivers, in modo sequenziale. Le funzioni **SPI_writeDataBlockingFIFO** e **SPI_readDataBlockingFIFO** consentono fare ciò in una unica trasmissione, mentre usando **spi_xmit** più volte di seguito si ha sempre un intervallo di tempo in cui Chip Select viene portato ad 1 dal microcontrollore, segnalando di fatto agli Slaves che la comunicazione è terminata, nonostante sia stata trasmessa una sola parola.

3.3 Sviluppo delle routine di comunicazione

L'interfaccia SPI si compone di tre parti: l'inizializzazione del modulo SPI, l'abilitazione dei gate drivers e la gestione degli errori.

Tutto il codice relativo alla SPI si trova nei files `SPI_management.c` e `gate_drivers_management.c`, assieme agli omonimi headers. Alcune righe di codice compaiono anche in `main.c` ed in `epwm_management.c`.

Nella prima fase della tesi è stato sviluppato un codice per la comunicazione SPI tra il microcontrollore ed un singolo driver. Per controllare un intero inverter a due livelli, tuttavia, sono necessari ben sei drivers. Questo comporta un elevato numero di collegamenti fisici poiché, per abilitare uno Slave alla comunicazione, il Master deve mettere a zero il corrispondente segnale di Chip Select. È quindi necessario un Chip Select per ogni Slave, per un totale di 6 pin, oltre a clock, MOSI, MISO. Per ridurre l'uso di GPIOs (General Purpose Input/Output) del microcontrollore, è possibile collegare i drivers in "daisy chain". In tal modo, i pin del Master necessari si riducono da nove a quattro.

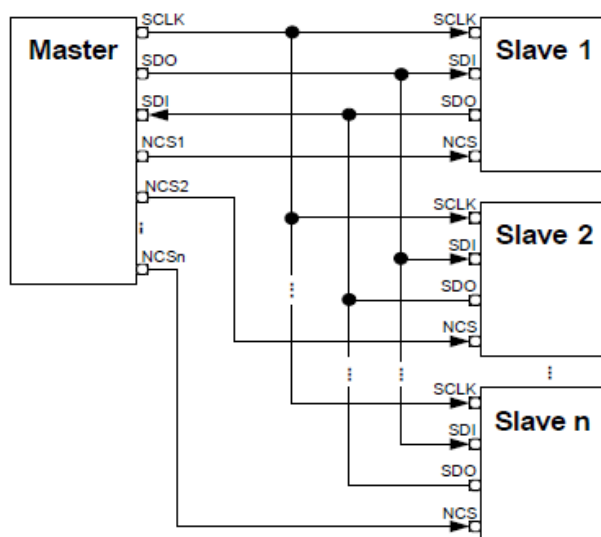


Fig. 3.5 - Topologia SPI standard

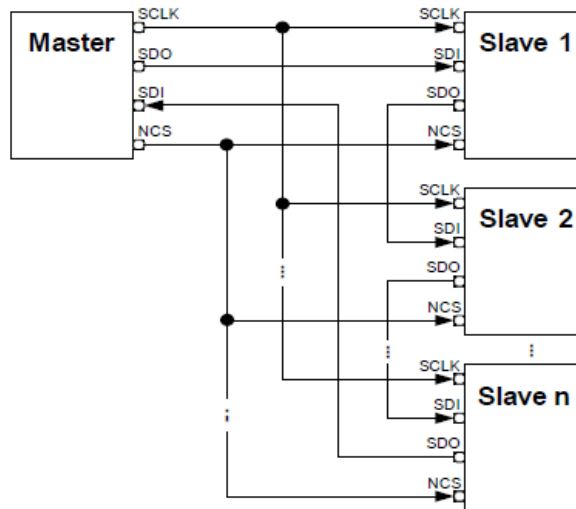


Fig. 3.6 - Topologia SPI daisy chain

Lato software, non vengono più trasmesse singole parole a 16 bit, poiché ora tutti i drivers sono collegati in modo seriale. Il primo driver, il cui SDI è direttamente collegato al microcontrollore, riceve tutte le parole destinate ai drivers successivi e le inoltra attraverso il suo SDO, che è collegato all'SDI del driver successivo.

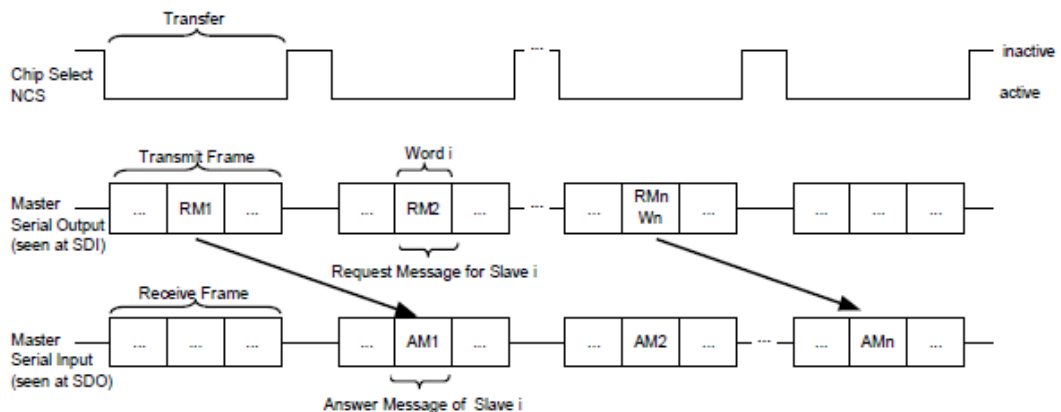


Fig. 3.7 - Trasmissione dati con interfaccia SPI, collegamento in daisy chain

Durante tutta la trasmissione il segnale Chip Select viene mantenuto basso dal Master, in modo che gli Slaves stiano pronti a ricevere altri messaggi finché la comunicazione non è terminata. Ciò equivale alla trasmissione di N parole, dove N corrisponde al numero di Slaves collegati.

Non essendo disponibili codici esempio prodotti dalla Texas, è stata sviluppata da zero una interfaccia in grado di gestire fino a 6 drivers. Il numero di Slaves collegati al microcontrollore si imposta nella macro `NUMBER_OF_GATE_DRIVERS`, all'interno del file `gate_drivers_management.h`.

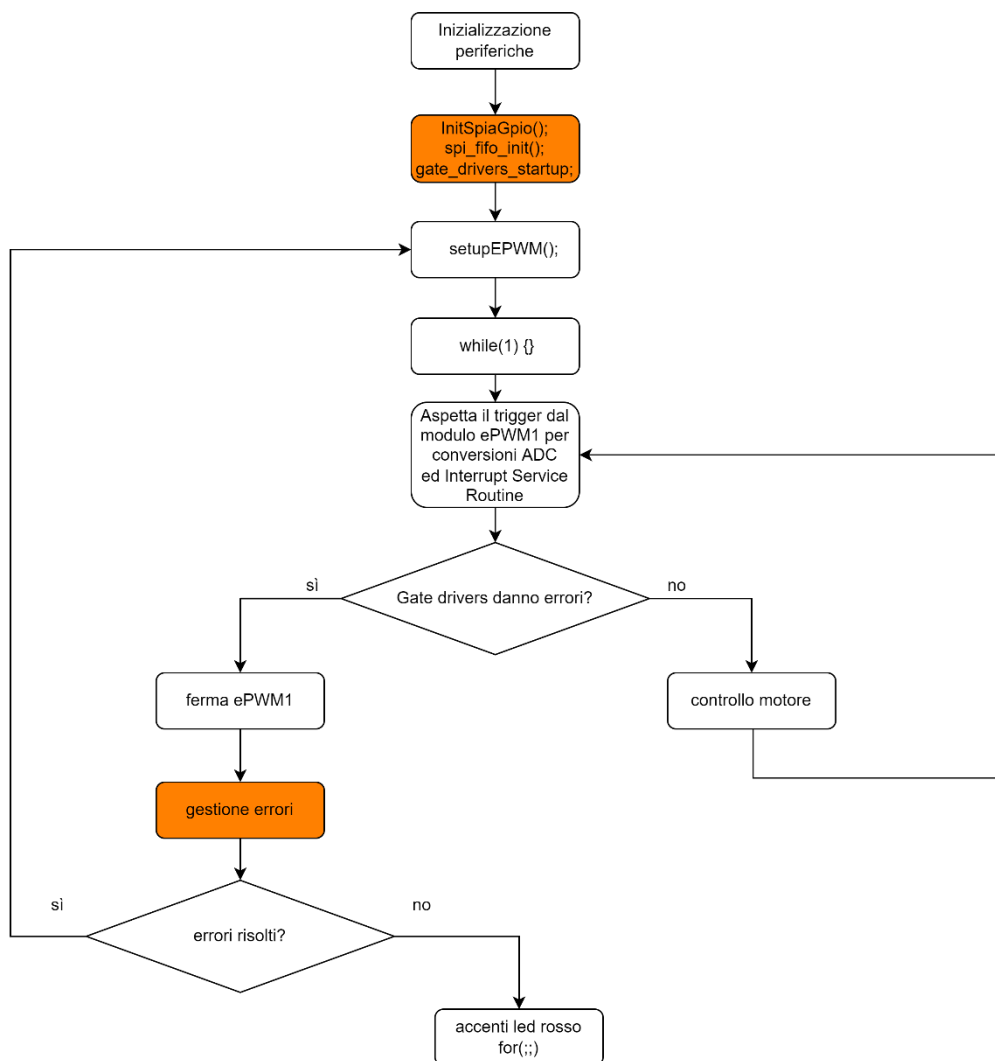


Fig. 3.8 - Struttura del firmware. In arancione sono evidenziate le operazioni che coinvolgono la comunicazione SPI

3.3.1 Inizializzazione

L'inizializzazione del modulo A della periferica SPI avviene nel main.c tramite le funzioni **InitSpiaGpio** e **spi_fifo_init**.

La prima inizializza i pin del modulo A della SPI. La SPI può funzionare in due modi: in High-Speed Mode e in modalità normale. L'High-Speed Mode può essere attivato solo su alcuni GPIOs, come indicato nel manuale del microcontrollore. Questi ultimi GPIOs possono essere utilizzati sia in modalità High-Speed che in modalità normale, tuttavia necessitano di una riga di codice aggiuntiva nella loro inizializzazione. Si riporta, a titolo esemplificativo, l'inizializzazione del GPIO59.

```

GpioCtrlRegs.GPBMUX2.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA
GpioCtrlRegs.GPBMUX2.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA (only
HS Mode pin)
  
```

La prima di codice sopra riportata va scritta per ogni pin usato per la SPI, mentre la seconda è particolare per questo tipo di pin. La seconda funzione di inizializzazione è **spi_fifo_init**, nella quale si configura il funzionamento vero e proprio del modulo SPI-A: funzionamento in modalità FIFO, schema di clocking, High-Speed Mode ed altro.

Tabella 3.1

Segnale	GPIO	Pin
Clock	18	4
Chip Select	19	3
MOSI (SDI del primo driver)	58	15
MISO (SDO dell'ultimo driver)	59	14

3.3.2 Comunicazione

Per sviluppare le funzioni usate dalla routine di comunicazione col gate driver, si è fatto riferimento a [9] e [10].

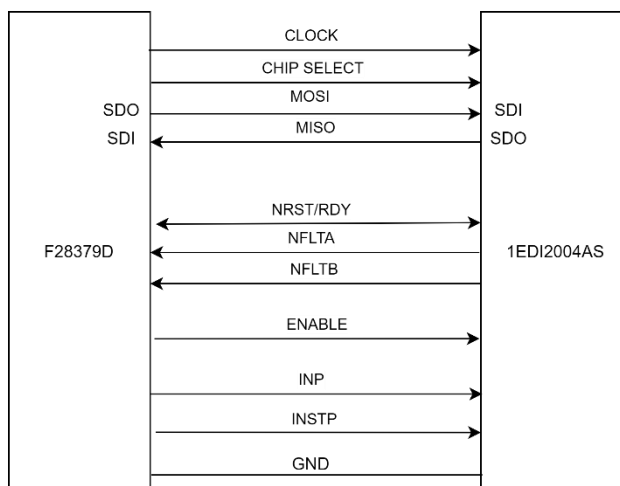


Fig. 3.9 - Segnali scambiati tra microcontrollore e gate driver

La parte di codice relativa alla comunicazione SPI con il gate driver è contenuta all'interno di quattro files:

1. spi_management.c contiene l'inizializzazione del modulo SPI-A
2. spi_management.h contiene i prototipi delle funzioni definite nell'omonimo file C
3. gate_drivers_management.c contiene tutte le funzioni di lettura e scrittura dei registri del gate driver, configurazione del dispositivo, gestione degli Operating Modes e calcolo del bit di parità
4. gate_drivers_management.h contiene i prototipi delle funzioni definite nell'omonimo file C, oltre ad un elenco di tutti i comandi e gli indirizzi dei registri disponibili per il gate driver 1EDI2004AS.

Il funzionamento del gate driver dipende dallo “stato” (Operating Mode, OPM) nel quale si trova il dispositivo. Per passare da OPM all’altro è necessario inviare determinate istruzioni ai drivers, sotto forma di numeri in formato esadecimale. Tutti i comandi sono salvati come macro in `gate_drivers_management.h`.

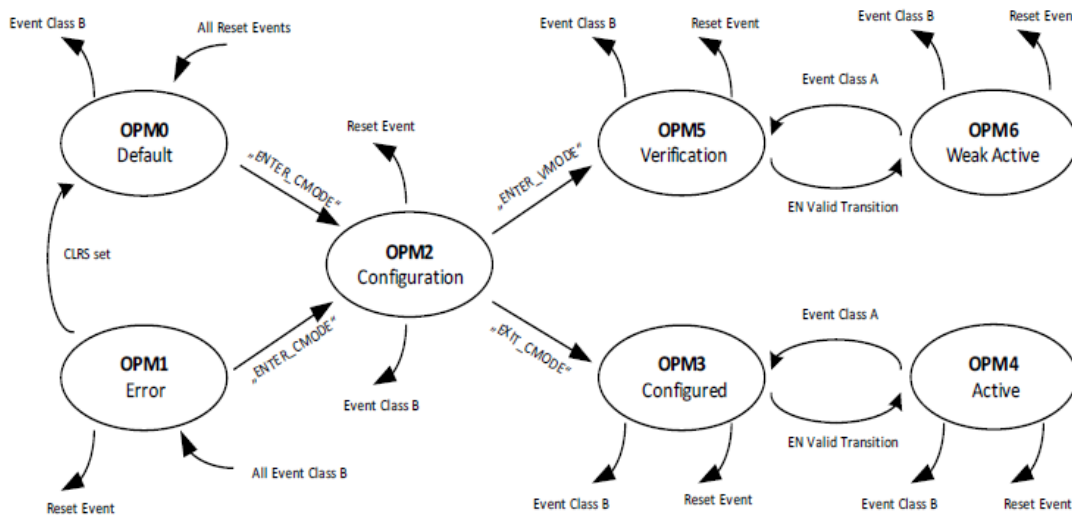


Fig. 3.10 - Diagramma degli stati di funzionamento del gate driver

All’accensione, il dispositivo di default si trova in OPM0. Tramite un opportuno comando (inviando via SPI il comando `ENTER_CMODE`) si può passare alla modalità di configurazione (OPM2), nella quale è possibile impostare numerosi parametri per gestire, ad esempio, le modalità di accensione e spegnimento dello switch controllato. Col comando `EXIT_CMODE` si passa in OPM3 e da lì, tramite l’attivazione di un apposito segnale di enable, si passa in OPM4, nel quale è possibile far commutare l’interruttore di potenza.

Gli errori del dispositivo (eventi) si distinguono in eventi di classe A, B, C. Gli eventi di classe A sono generalmente legati all’intervento di protezioni, quali la desaturazione (DESAT) o la protezione da sovracorrente (OCP). Gli eventi di classe B riguardano perlopiù il verificarsi di problemi nell’alimentazione del primario o del secondario del driver. Gli eventi di classe C, infine, generalmente non sono legati a transizioni di Operating Mode e possono essere dovuti, ad esempio, ad errori con la SPI.

Tutti gli eventi di classe B portano il dispositivo in OPM1. È possibile forzare la transizione ad OPM0 eseguendo un reset manuale tramite il segnale NRST/RDY, portandolo a zero per almeno 10 μs.

Le funzioni scritte per comunicare con i drivers sono riportate in tabella.

Tabella 3.2

parity_bit_calculator	Calcola il bit di parità dispari da sommare alla trasmissione
Read	Restituisce il contenuto del registro specificato di tutti i drivers
writeH	Scrive nel byte più significativo del registro specificato

writel	Scrive nel byte meno significativo del registro specificato
getOPM	Restituisce l'operating Mode di tutti i drivers
isSPIworking	Indica se la comunicazione SPI funziona correttamente
OPM0_to_OPM2	Manda i drivers da OPM0 ad OPM2
configuration_in_OPM2	Configura i drivers mentre sono in OPM2
OPM2_to_OPM3	Manda i drivers da OPM2 ad OPM3
OPM3_to_OPM4	Manda i drivers da OPM3 ad OPM4
gate_drivers_startup	Manda i drivers da OPM0 ad OPM4
drivers_error_management	Gestisce gli errori relativi a drivers ed SPI

Come già accennato nel paragrafo precedente, l'interfaccia SPI consente al Master di ricevere dati simultaneamente alla trasmissione di altri dati allo Slave. Ne consegue che, quando il Master interpella lo Slave, ad esempio richiedendo il valore di un determinato registro, lo Slave possa rispondere solo nella successiva trasmissione.

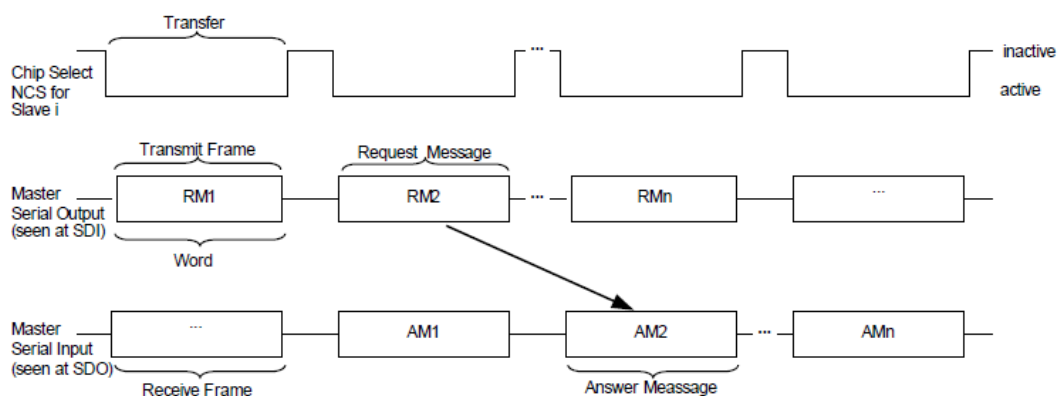


Fig. 3.11 - Principio di domanda e risposta con interfaccia SPI

Sono quindi necessarie due trasmissioni per ogni interrogazione dello Slave da parte del Master. Le due trasmissioni sono gestite dalla funzione **read**. Essa prende in ingresso l'indirizzo del registro del quale si intende conoscere il valore e restituisce il contenuto scrivendo nell'array rData.

L'indirizzo, composto da 5 bit, viene passato alla funzione sotto forma di intero a 16 bit, nel formato 0000 0000 000X XXXX. Questo viene poi traslato di 7 bit verso sinistra e poi ad esso si somma un offset, come indicato nella guida Infineon. Occorre poi sommare l'eventuale bit di parità dispari.

Il bit di parità è un semplice meccanismo di controllo dell'integrità dei dati ricevuti: il trasmettente conta il numero di bit a 1 nel dato da trasmettere e ad esso somma il bit di parità. Prendendo ad esempi la parità dispari, il bit di parità vale 1 se i bit a 1 sono pari, in caso contrario vale 0. Il ricevente verifica che i bit ad 1 del dato ricevuto siano dispari, se così non fosse il messaggio ricevuto verrebbe scartato in quanto corrotto. Al contrario, con parità pari il bit di parità vale 1 se il numero di bit ad 1 è dispari e 0 in caso contrario.

Successivamente la variabile a viene inviata via SPI, come già visto, tramite **SPI_writeDataBlockingFIFO**. Così si effettua la prima trasmissione. Con la seconda trasmissione si potrebbe mandare un qualsiasi messaggio al driver, poiché ciò che interessa effettivamente è la risposta alla prima trasmissione. Per semplicità si manda quindi un **NOPE**, al quale il driver risponde sempre restituendo il valore del registro PSTAT. Tra la prima e la seconda trasmissione si è volutamente introdotto un ritardo tramite un semplice ciclo for, poiché il driver potrebbe essere lento a rispondere se interpellato sul valore di un registro secondario (il cui indirizzo in memoria si trova al secondario isolato). Nella pratica, senza questo delay, si sono ripetutamente osservati malfunzionamenti.

Dato che i dispositivi collegati tramite daisy chain nella gate driver board definitiva sono sei, sarebbe opportuno definire un metodo per poter interrogare con precisione i singoli drivers. I dispositivi, tuttavia, non contengono in memoria una registro che consente di identificarli univocamente. Per ovviare a questo problema, si può pensare di sfruttare i primi 16 bit del registro PRW, nel quale si può scrivere un valore a piacere. Si può quindi di assegnare un valore diverso ad ogni driver, per poterlo poi identificare univocamente, perdendo però la possibilità di usare PRW per verificare il corretto funzionamento della comunicazione SPI, come indicato nel seguito. Un sistema più semplice e usato in questo lavoro di tesi è il seguente. Una volta definito il layout della scheda con i drivers e tracciate le piste di collegamento, è immediato realizzare che il primo dispositivo della daisy chain che riceve il messaggio a lui destinato è il più “distante” dal microcontrollore, visto dal segnale MOSI o, alternativamente, il più “vicino” dal punto di vista del segnale MISO. Sarà quindi anche il primo a rispondere al master, poiché il suo messaggio di risposta non deve attraversare altri drivers. L’array rData è composto da sei elementi: quello con indice 0 contiene quindi la risposta del driver più “vicino”, quello con indice 5 la risposta del driver più “lontano”. Allo stesso modo, nell’invio dei messaggi, il primo messaggio scritto nel FIFO di uscita è destinato al driver più “lontano”.

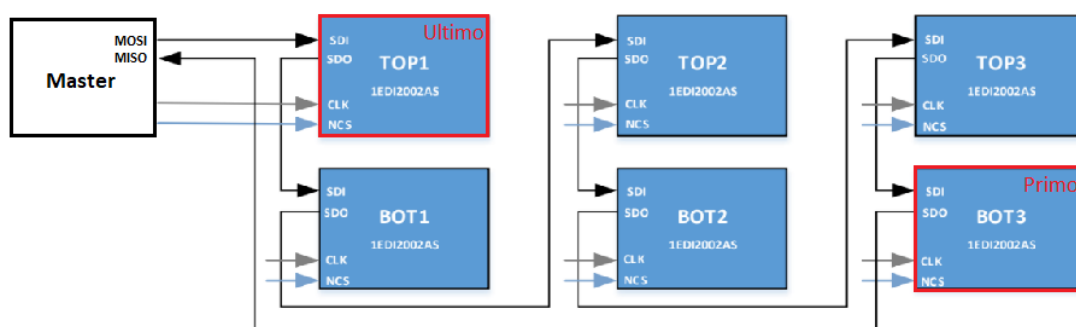


Fig. 3.12 - Collegamento in daisy chain, in evidenza il primo (BOT3) e l'ultimo (TOP1) dispositivo a rispondere

Le altre due funzioni basilari nella comunicazione con il driver sono **writeH** e **writel**, le quali consentono di scrivere rispettivamente nel primo byte (più a sinistra) e nel secondo byte (più a destra) di un determinato registro.

Va specificato che l’effettiva scrittura in un qualsiasi registro del gate driver avviene solo a mezzo della funzione **writel**. Ciò significa che, anche se si intende scrivere solo nei primi 8 bit di un registro, richiamare la sola funzione **writeH** non ha alcun effetto

se subito dopo non si richiama la funzione **writeL**. Le due righe di codice sottostanti sono esemplificative di quanto appena spiegato: si supponga di voler porre a 1 tutti i bit del byte più significativo del registro PRW. Affinché la scrittura abbia effetto, è necessario che **writeH** sia seguito da **writeL**, anche se l'input di **writeH** è costituito da soli zeri.

```
writeH(0x00FF);  
writeL(PRW, 0x0000);
```

Nel main.c, subito dopo l'inizializzazione SPI tramite **InitSpiaGpio** e **spi_fifo_init**, viene eseguito un check se la SPI funziona per tutti i dispositivi connessi, tramite **isSPIworking**. Questa funzione richiede a tutti i drivers connessi di comunicare al microcontrollore il contenuto di due registri (PID e SID), i cui primi 12 bit assumono dei valori sempre costanti, come indicato in [9]. In particolare, in notazione esadecimale, i primi 12 bit di PID valgono sempre 0x4910 e quelli di SID 0x8910. Se ciò che il microcontrollore legge come risposta a questa interpellazione è diverso, **isSPIworking** restituisce 0, altrimenti 1, indicando che tutto funziona correttamente. Questa funzione consente quindi sia di verificare che la SPI funzioni, sia che i lati primario e secondario dei drivers siano funzionanti.

Dopo questa verifica, utile durante il debug, viene eseguita **gate_drivers_startup**. Al suo interno essa raccoglie le funzioni che gestiscono tutti i passaggi da OPM0 ad OPM4, nel quale il driver può effettivamente far commutare l'IGBT, e la configurazione dei drivers. È stato verificato che conviene inizializzare il modulo PWM dopo questi passaggi relativi alla SPI; in tal modo si hanno meno problemi di compatibilità tra le varie periferiche.

```
InitSpiaGpio(); // Setup only the GP I/O only for SPI-A functionality  
spi_fifo_init(); // Initialize the SPI FIFO  
  
//check sulla SPI, questi flag sono utili in debug  
isSPIok = isSPIworking(rData);  
  
gate_drivers_startup(OPMs, rData);
```

3.3.3 Gestione degli errori

Una volta inizializzati i gate drivers ed abilitati a commutare l'IGBT, non è più necessario comunicare con essi via SPI, a meno che non si verifichino errori o malfunzionamenti.

Ogni gate driver si serve dei segnali RDY, NFLTA e NFLTB per comunicare al microcontrollore eventuali errori o malfunzionamenti. Questi segnali sono tutti ad 1 quando il funzionamento è corretto. Nello specifico, RDY nello stato alto indica che il primario del driver funziona correttamente, mentre NFLTA ed NFLTB sono bassi quando si verifica un evento anomalo.

È importante, quindi, monitorare questi tre segnali durante il funzionamento dell'inverter, al fine di intervenire tempestivamente in caso di malfunzionamenti. Il

microcontrollore, pertanto, legge i loro valori e, se almeno uno di essi è zero, esegue una apposita funzione di gestione degli errori (**drivers_error_management**).

La verifica dei tre segnali viene eseguita dal microcontrollore con cadenza regolare, all'inizio della Interrupt Service Routine (ISR) richiamata dalla PWM.

Prima di descrivere il funzionamento della gestione errori si sottolinea che, nel caso in cui uno dei tre segnali sia nello stato logico basso, è opportuno disabilitare il contatore del modulo ePWM1. Questo va fatto poiché la comunicazione SPI con i drivers richiede un tempo maggiore di quello a disposizione tra una chiamata e la successiva della Interrupt Service Routine (pari all'inverso della frequenza del controllo f_c). In altre parole, se si verificasse un errore, il microcontrollore andrebbe a comunicare con i drivers per stabilire quanto accaduto, leggendo i loro registri, ma nel frattempo la ISR contenente il controllo motore verrebbe chiamata nuovamente poiché il contatore dell' ePWM1 sarebbe ancora attivo. Avrebbe quindi inizio nuovamente la routine di controllo, verrebbe eseguita un'altra volta la gestione degli errori e l'intero processo si ripeterebbe alla successiva chiamata della ISR.

Con le seguenti righe di codice, quindi, si ferma il contatore del modulo ePWM1 e si disabilitano le uscite PWM della LAUNCHXL-F28379D.

```
EPWM_setTimeBaseCounterMode(EPWM1_BASE, EPWM_COUNTER_MODE_STOP_FREEZE);
disablePWM(); //blocco le uscite PWM della scheda
pwm_status = 0; //segnalo l'attivazione della trip-zone
```

Successivamente si esegue la funzione **drivers_error_management**. Essa prevede innanzitutto di disabilitare il segnale di enable ai gate drivers, in modo che questi non possano commutare. Potrebbe accadere infatti che solo un driver sia in errore, mentre gli altri continuano a commutare l'IGBT. Si vuole inoltre evitare in qualsiasi caso la ripresa accidentale delle commutazioni. Il secondo passo consiste nel verificare che il primario dei drivers sia funzionante, tramite la lettura del segnale RDY. Se lo stato logico letto al pin corrispondente è 1, si prosegue con i passi successivi, altrimenti il microcontrollore entra nello stato di errore, ovvero accende il LED rosso della scheda LAUNCHXL-F28379D ed entra in un ciclo for infinito, in quanto se il circuito primario di almeno un gate driver non funziona, probabilmente è in corso un problema abbastanza grave che non può essere risolto dal software. Il segnale RDY che viene portato al microcontrollore è in comune a tutti i drivers per una questione di praticità.

Il segnale RDY nel driver prende il nome di NRST/RDY, in quanto bidirezionale. Può essere letto per monitorare lo stato del dispositivo (RDY) ma può anche essere forzato a zero dall'esterno per resettare (NRST). Nello schematico della gate driver board, riportato in [8], i segnali sono chiamati NRST_3V3_OUT (corrispondente a RDY) e NRST_3V3_IN (corrispondente a NRST).

Si verifica poi che la comunicazione SPI sia funzionante. Se anche questa condizione è verificata, si richiede ai drivers i contenuti dei registri PER e SER di ognuno, altrimenti si entra nello stato di errore descritto precedentemente. PER contiene informazioni sugli errori al primario, SER sugli errori al secondario di ogni dispositivo. I contenuti dei registri vengono salvati nell'array errors, che si compone di sei elementi a 32 bit, ognuno corrispondente ad un driver. Ogni elemento di errors

contiene, nei primi 16 bit a sinistra, il contenuto di PER e nella restante porzione il contenuto di SER. Errors è visibile da debug e consente pertanto di risalire, attraverso l'interpretazione dei singoli bit con il manuale dei drivers, all'errore che ha portato al malfunzionamento ed all'individuazione del dispositivo interessato.

Quando si verifica un errore, ci si aspetta che uno o più drivers siano finiti dall'OPM4 ad OPM1 o OPM3, tuttavia verificare in quale Operating Mode si trovi ogni dispositivo rappresenterebbe uno sforzo inutile. Si esegue pertanto un reset di tipo "hardware", forzando a zero il pin NRST (collegato sempre a NRST/RDY di tutti i drivers). Applicando lo stato logico basso su NRST, si forzano tutti i drivers ad entrare in OPM0. Lo stesso effetto si ottiene interrompendo l'alimentazione in bassa tensione e successivamente ripristinandola. Una volta in OPM0, i drivers possono essere riconfigurati e portati nuovamente in OPM4 tramite la funzione **gate_drivers_startup**. In fase di collaudo ci si è accorti che la gestione bidirezionale del NRST/RDY dei gate drivers è alquanto complicata. Unitamente a ciò, ogniqualvolta si verifica un errore, cioè almeno uno dei tre segnali di notifica RDY, NFLTA e NFLTB va basso, è più sicuro fermare l'attività del microcontrollore. Per questi motivi, in seguito alla lettura del contenuto dei registri PER e SER il microcontrollore ha istruzione di fermarsi completamente.

Ogniqualvolta che si verifica un errore, la variabile faults incrementa di una unità, consentendo così di monitorare la comparsa di anomalie che, se venissero risolte autonomamente dal microcontrollore, non sarebbero altrimenti visibili all'esterno. Qualora si decidesse di implementare nel codice le istruzioni per permettere al microcontrollore di risolvere autonomamente alcuni errori, terminata l'esecuzione di **drivers_error_management** occorrerebbe riattivare la PWM in modo che la ISR venga chiamata con cadenza regolare.

```
setupEPWM(); //faccio ripartire tutti i moduli PWM per sincronizzati
```

3.4 Collaudo

Il funzionamento della comunicazione SPI è stato verificato dapprima con la scheda di figura 3.1 e, successivamente, con la scheda destinata ad alloggiare i sei gate drivers, collegati in daisy chain. I test con la seconda scheda si sono rivelati complicati in quanto i componenti chiave, ovvero i drivers ed il traslatore di livello non funzionano correttamente se saldati a mano. Facendoli saldare da un'azienda sponsor, tutti i problemi di saldatura sono stati risolti.

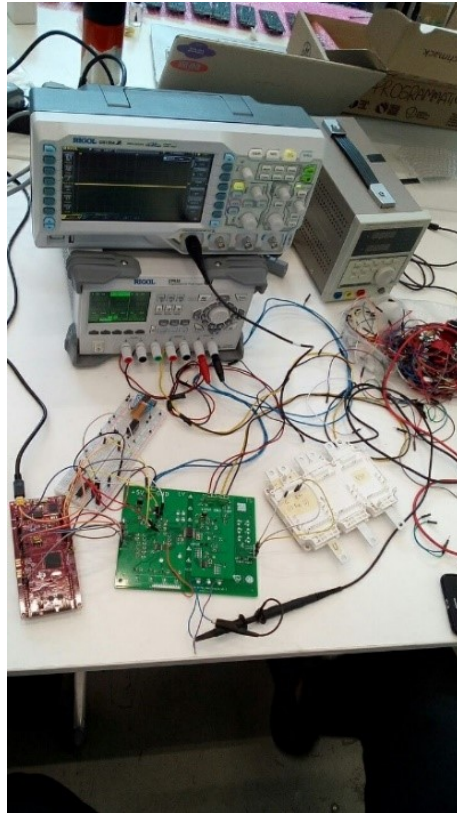


Fig. 3.13 – Primo test sulla scheda per singolo gate driver

Per tutti i test, la frequenza del clock della SPI è stata impostata a 1.72 MHz, rispettando così il limite di 1.8 MHz indicato nel datasheet per il collegamento in daisy chain. Nella scheda da 6 drivers, il funzionamento della SPI è stato testato dapprima saldando un solo driver e poi aggiungendo man mano i restanti componenti. In figura 3.14 è riportato l'andamento del segnale di clock in ingresso ai drivers.

Per verificare il funzionamento della SPI è sufficiente monitorare da debug la variabile `isSPiok`, nella quale viene salvato il valore restituito da `isSPIworking`. Leggendo 1 si verifica il corretto funzionamento.

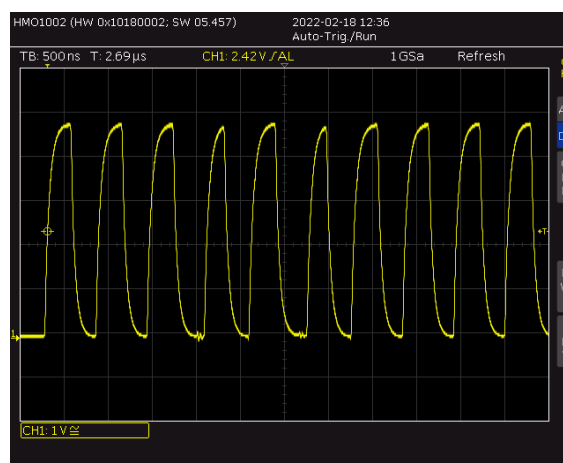


Fig. 3.14 - Segnale di clock misurato tra il traslatore di livello e l'ingresso del primo gate driver. L'andamento del segnale risulta essere smussato rispetto all'onda quadrata, in quanto è misurato a valle del filtro passa-basso analogico del primo ordine che serve a filtrare il segnale

4 Algoritmi di controllo di un motore brushless

4.1 Il sistema di riferimento sincrono dq

Un sistema elettrico trifase è descritto da una terna di grandezze (tensioni, correnti, flussi..). Sarebbe conveniente descrivere lo stesso sistema tramite una coppia di grandezze, cioè trattare il sistema come se fosse “bifase”. Lo strumento matematico che permette di fare questo passaggio è chiamato “vettore spaziale”.

Si supponga di avere una terna di grandezze $g_a(t)$, $g_b(t)$, $g_c(t)$. Il vettore spaziale ad essa associato è definito come:

$$\mathbf{g}(t) = \frac{2}{3} \left\{ g_a(t) + g_b(t)e^{j\frac{2\pi}{3}} + g_c(t)e^{j\frac{4\pi}{3}} \right\} = g_\alpha(t) + jg_\beta(t) \quad (4.1)$$

Il vettore spaziale è, quindi, una grandezza “bidimensionale” associata ad una terna trifase di qualsiasi forma d’onda. Esso si può scomporre in una componente reale ed una immaginaria:

$$\begin{aligned} g_\alpha(t) &= \text{Re}[\mathbf{g}(t)] \\ g_\beta(t) &= \text{Im}[\mathbf{g}(t)] \end{aligned} \quad (4.2)$$

quindi è intuitivo che il vettore spaziale può essere rappresentato sia nel sistema di riferimento “tradizionale” trifase abc sia in quello complesso $\alpha\beta$. Nel caso delle macchine elettriche rotanti trifasi, se si fanno coincidere gli assi magnetici degli avvolgimenti di fase a, b, c con gli omonimi assi del sistema di riferimento trifase abc, si ottiene una rappresentazione “spaziale” e temporale delle grandezze elettriche.

Il piano $\alpha\beta$ è detto “stazionario” in quanto, nel caso delle macchine rotanti, è fisso rispetto al piano abc e quindi anche rispetto all’avvolgimento di statore.

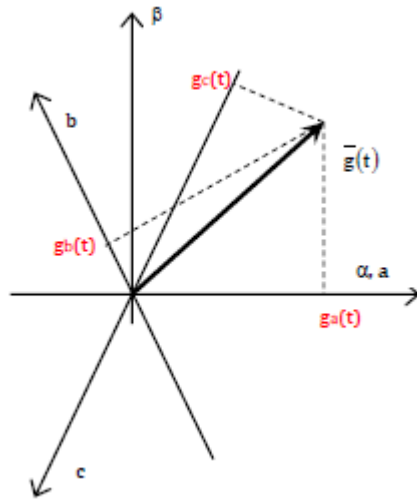


Fig. 4.1 - Vettore spaziale nei sistemi di riferimento abc ed αβ. Si noti come i due sistemi sono "fissi" tra di loro, inoltre l'asse a coincide con l'asse α. Gli assi a, b, c sono sfasati di 120° tra di loro, al pari delle grandezze elettriche delle tre fasi

Per passare da un sistema di riferimento all'altro si fa uso della seguente espressione matriciale (compare anche la coordinata "0" che è di interesse per grandezze omopolari):

$$\mathbf{T}_{abc \rightarrow \alpha\beta 0} = \frac{2}{3} \begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \\ 1/2 & 1/2 & 1/2 \end{bmatrix} \quad (4.3)$$

in modo che:

$$\begin{bmatrix} g_\alpha \\ g_\beta \\ g_0 \end{bmatrix} = \mathbf{T}_{abc \rightarrow \alpha\beta 0} \begin{bmatrix} g_a \\ g_b \\ g_c \end{bmatrix} \quad (4.4)$$

Per effettuare la trasformazione inversa:

$$\begin{bmatrix} g_a \\ g_b \\ g_c \end{bmatrix} = \mathbf{T}_{\alpha\beta 0 \rightarrow abc} \begin{bmatrix} g_\alpha \\ g_\beta \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ -1/2 & \sqrt{3}/2 & 1 \\ -1/2 & -\sqrt{3}/2 & 1 \end{bmatrix} \begin{bmatrix} g_\alpha \\ g_\beta \\ g_0 \end{bmatrix} \quad (4.5)$$

Se la terna $g_a(t)$, $g_b(t)$, $g_c(t)$ è bilanciata, cioè la componente omopolare $g_0(t) = \frac{g_a(t) + g_b(t) + g_c(t)}{3}$ è nulla o non è di interesse, allora l'ultima riga della $\mathbf{T}_{abc \rightarrow \alpha\beta 0}$ e l'ultima colonna della $\mathbf{T}_{\alpha\beta 0 \rightarrow abc}$ possono essere ignorate.

Nell'ambito delle macchine rotanti e degli azionamenti elettrici, spesso è d'interesse esprimere le grandezze elettriche in un sistema di riferimento solidale con il rotore, per motivi spiegati nel seguito. Si introduce un nuovo sistema di assi, composto dagli assi d e q e pertanto chiamato "sistema sincrono dq". Esso ruota rispetto allo statore

e, quindi, rispetto al sistema $\alpha\beta 0$ con una velocità pari a ω_{dq} . L'angolo di sfasamento tra l'asse d è l'asse α si calcola come:

$$\theta_{dq} = \int_0^t \omega_{dq}(t) dt + \theta_{dq}(0) \quad (4.6)$$

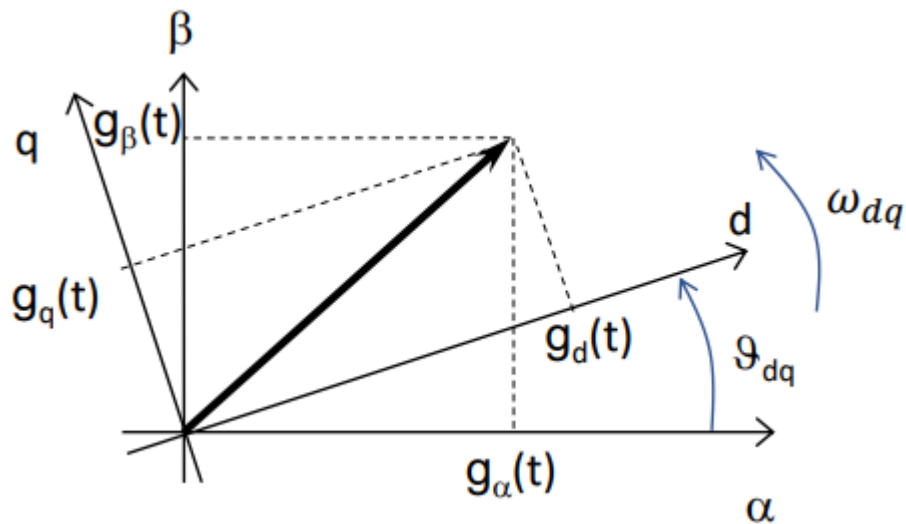


Fig. 4.2 - Sistemi di assi $\alpha\beta$ e dq , $g(t)$ è la generica grandezza rappresentata tramite il vettore spaziale

Nelle macchine sincrone, la velocità di rotazione è direttamente proporzionale alla frequenza f . La pulsazione elettrica è $\omega_{me} = 2\pi f = p\omega_m$, dove p sono le coppie polari e ω_m la velocità meccanica. È conveniente, ai fini del controllo, far coincidere la ω_{dq} con la ω_{me} [rad.elettrici/s], detta velocità meccanica-elettrica.

Convenzionalmente, vale che $\theta_{dq} = \theta_{me} = 0$ quando l'asse d di rotore, coincidente con il polo Nord di rotore, è allineato con l'asse magnetico della fase a di statore. La disposizione degli assi diretto (d) ed in quadratura (q) è rigidamente collegata con la struttura magnetica della macchina rotante.

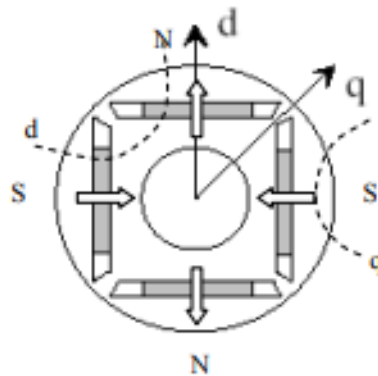


Fig. 4.3 - Esempio di rotore di macchina a magneti permanenti anisotropi a 4 poli ($p = 2$). I magneti (in grigio scuro) sono annegati nel ferro di rotore. L'asse d coincide con il polo Nord di rotore, mentre l'asse q si trova in quadratura sull'asse d. Si noti come, in una macchina a p coppie polari, gli assi d e q siano presenti p volte (una per ogni coppia di poli N-S)

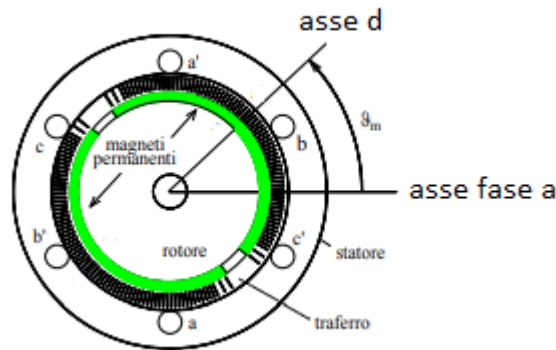


Fig. 4.4 - Macchina isotropa a due poli ($p = 1$). In questo caso, l'angolo elettrico-meccanico θ_{me} coincide con l'angolo meccanico θ_m . Vale infatti la relazione $\theta_{me} = p\theta_m$

Tra le grandezze nel sistema di riferimento dq ed in quello $\alpha\beta$, ignorando la componente omopolare, sussiste la seguente relazione:

$$\mathbf{g}_{\alpha\beta}(t) = \mathbf{g}_{dq}(t) e^{j\theta_{dq}} \quad (4.7)$$

Le trasformazioni da $\alpha\beta$ a dq e viceversa sono conservative sia per le ampiezze che per le potenze, mentre da abc a $\alpha\beta/dq$ e viceversa si ha soltanto la conservazione delle ampiezze, in quanto solo l'ampiezza del vettore spaziale è costante nei tre sistemi di riferimento.

La potenza istantanea $p(t)$ nel sistema stazionario abc è $\frac{3}{2}$ volte quella nel sistema di riferimento sincrono dq:

$$p(t) = u_a(t)i_a(t) + u_b(t)i_b(t) + u_c(t)i_c(t) = \frac{3}{2}[u_d(t)i_d(t) + u_q(t)i_q(t)] \quad (4.8)$$

Esiste, tuttavia, anche la versione invariante alle potenze della trasformazione da abc a $\alpha\beta/dq$ e viceversa, qui non trattata.

Per concludere, i motivi per cui si sviluppano algoritmi di controllo nel sistema di riferimento sincrono sono i seguenti. Innanzitutto, le grandezze elettriche alternate, quali tensioni e correnti, se descritte nel piano dq, a regime sono continue e non sinusoidi. È possibile pertanto, in presenza di azione integrale nel controllo, ottenere un errore a regime nullo in risposta ad un riferimento a gradino.

Se il controllo fosse realizzato in uno dei due sistemi di riferimento stazionari (abc o $\alpha\beta$), sarebbero i regolatori (ad esempio di tipo proporzionale-integrale) a dover produrre le sinusoidi caratteristiche dei sistemi in alternata, con conseguenti requisiti in termini di banda passante. Nel sistema sincrono, la banda passante dei regolatori può essere ben inferiore alla massima frequenza elettrica del motore.

Dal punto di vista computazionale, infine, eseguire l'azione di controllo su due grandezze risulta essere più semplice rispetto al caso trifase.

4.2 Equazioni del motore a magneti permanenti interni in dq

Il motore oggetto del controllo in questo lavoro di tesi è del tipo sincrono, a magneti permanenti interni. Di conseguenza, questo paragrafo tratterà soltanto le macchine anisotrope.

Le equazioni generali di bilancio delle tensioni delle fasi a, b, c di statore sono, nell'ipotesi di assenza di saturazione dei circuiti magnetici:

$$\begin{cases} u_a(t) = Ri_a(t) + L(\theta_{me}) \frac{di_a(t)}{dt} + e_a(t) \\ u_b(t) = Ri_b(t) + L(\theta_{me}) \frac{di_b(t)}{dt} + e_b(t) \\ u_c(t) = Ri_c(t) + L(\theta_{me}) \frac{di_c(t)}{dt} + e_c(t) \end{cases} \quad (4.9)$$

dove R è la resistenza di fase, L l'induttanza sincrona ed $e(t)$ la forza contro elettromotrice indotta dal movimento dei magneti permanenti rispetto allo statore. L'induttanza L è data dalla somma dell'autoinduttanza di fase e la mutua induttanza con le altre fasi. Essa è funzione della posizione reciproca θ_{me} tra rotore e statore, poiché le macchine qui trattate sono di tipo anisotropo. Applicando le trasformazioni matriciali trattate nel paragrafo precedente, si giunge alla formulazione delle equazioni di tensione nel sistema di riferimento dq, sincrono con il rotore (per semplicità, non si indica la dipendenza temporale di tensioni e correnti):

$$\begin{cases} u_d = Ri_d + L_d \frac{di_d}{dt} - \omega_{me} L_q i_q \\ u_q = Ri_q + L_q \frac{di_q}{dt} + \omega_{me} L_d i_d + \omega_{me} \Lambda_{mg} \end{cases} \quad (4.10)$$

dove Λ_{mg} è il flusso concatenato dovuto ai magneti permanenti. Le induttanze L_d ed L_q che qui compaiono rappresentano rispettivamente il valore minimo ed il valore massimo dell'induttanza sincrona al variare di θ_{me} . Queste induttanze caratterizzano i due avvolgimenti fittizi (disposti lungo gli assi diretto ed in quadratura) nei quali si immagina di scomporre la macchina. Quando L_d viene attraversata da corrente, essa produce un flusso che agisce soltanto lungo l'asse polare (asse d), mentre il flusso associato a L_q agisce lungo l'asse in quadratura (asse q). Moltiplicando entrambi i membri della prima equazione per $i_d dt$, quelli della seconda per $i_q dt$ e poi sommando membro a membro si ottiene:

$$\begin{aligned} (u_d i_d + u_q i_q) dt &= R(i_d^2 + i_q^2) dt + L_d i_d di_d + \\ &L_q i_q di_q + \omega_{me} [\Lambda_{mg} i_q + (L_d - L_q) i_d i_q] dt \end{aligned} \quad (4.11)$$

Il primo membro è proporzionale all'energia elettrica assorbita dal motore nell'intervallo di tempo infinitesimo dt , mentre a secondo membro si hanno quattro termini. Da sinistra verso destra, essi sono proporzionali rispettivamente all'energia dissipata sulla resistenza di fase, all'energia associata ai campi magnetici dei due assi ed infine all'energia meccanica resa disponibile all'albero, nel funzionamento da

motore. Ricordando che la potenza meccanica è $P_m = m \omega_m$, si ricava l'equazione della coppia m :

$$m = \frac{3}{2}p\Lambda_{mg}i_q + \frac{3}{2}p(L_d - L_q)i_d i_q \quad (4.12)$$

Le induttanze L_d e L_q vengono considerate costanti durante il funzionamento della macchina e vengono chiamate apparenti. In realtà, quando la corrente nell'avvolgimento di statore è elevata, esse variano in modo non lineare a causa della saturazione del ferro, pertanto l'equazione della coppia sopra riportata è valida solo per correnti modeste. Una equazione sempre valida, invece, tiene conto dei flussi concatenati dai due assi Λ_d e Λ_q :

$$m = \frac{3}{2}p[\Lambda_d(i_d)i_q - \Lambda_q(i_q)i_d] \quad (4.13)$$

Si noti come la coppia sia legata alle correnti di entrambi gli assi: sfruttando l'anisotropia magnetica ($L_d \neq L_q$), si ottiene la coppia di riluttanza (termine proporzionale a $L_d - L_q$) che si va a sommare a quella dovuta ai magneti (termine proporzionale a Λ_{mg}).

Il carico, generalmente di tipo inerziale-viscoso lineare, si può descrivere con la seguente equazione:

$$m = m_L + B\omega_m + J \frac{d\omega_m}{dt} \quad (4.14)$$

dove m è la coppia erogata dal motore, m_L è la coppia di carico, il secondo termine è la coppia d'attrito viscoso ed infine si ha la coppia inerziale, legata alla variazione di energia cinetica e quindi ad accelerazione o decelerazione del rotore.

Si può quindi tracciare lo schema a blocchi del sistema motore con carico.

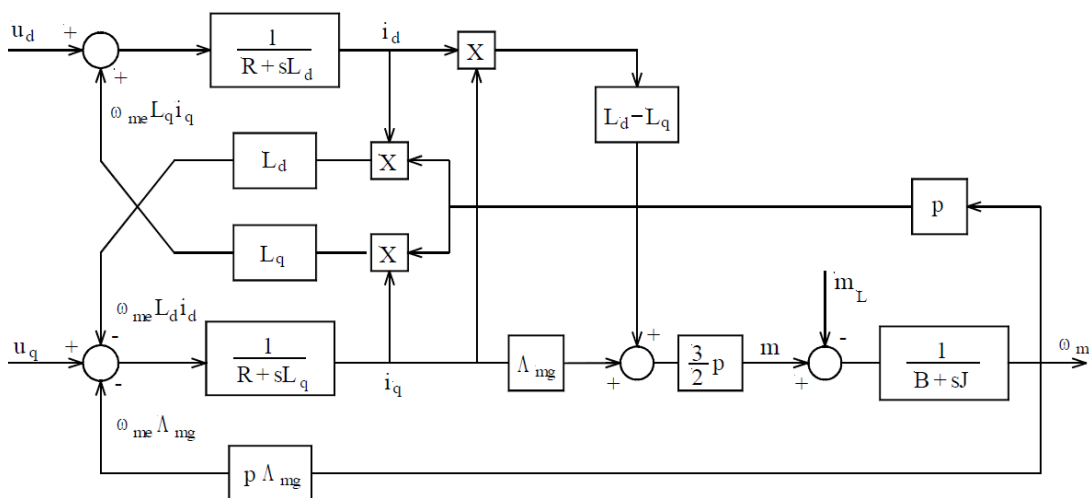


Fig. 4.5 - Schema a blocchi della macchina a magneti permanenti anisotropa

4.3 Regioni di funzionamento del motore brushless anisotropo

Si considera il funzionamento a regime del motore, ovvero con tensioni e correnti di fase sinusoidali, con ampiezza e frequenza costanti. La velocità Ω è costante, così come le grandezze trasformate U_d, U_q, I_d, I_q che pertanto verranno indicate in maiuscolo.

La corrente di fase non deve superare la corrente nominale I_N , espressa in valore di picco:

$$I_d^2 + I_q^2 \leq I_N^2 \quad (4.15)$$

Allo stesso modo, la tensione di fase non deve superare il valore nominale U_N , anch'esso espresso in valore di picco:

$$U_d^2 + U_q^2 \leq U_N^2 \quad (4.16)$$

Dato che solitamente la targa del motore riporta la tensione nominale concatenata U_{nom} , in valore efficace, vale che $U_N = \sqrt{\frac{2}{3}} U_{nom}$.

Riprendendo le equazioni di tensione, assumendo valori costanti di tensioni e correnti e trascurando la caduta di tensione resistiva, si può scrivere:

$$\begin{cases} U_d = -\Omega_{me} L_q I_q \\ U_q = \Omega_{me} L_d I_d + \Omega_{me} \Lambda_{mg} \end{cases} \quad (4.17)$$

Sostituendo queste equazioni nel limite di tensione, si ottiene:

$$(\Omega_{me} L_q I_q)^2 + (\Omega_{me} L_d I_d + \Omega_{me} \Lambda_{mg})^2 \leq U_N^2 \quad (4.18)$$

Dividendo tutto per $(\Omega_{me} L_d)^2$ si ottiene:

$$\left(I_d + \frac{\Lambda_{mg}}{L_d}\right)^2 + \left(\frac{L_q}{L_d} I_q\right)^2 \leq \frac{U_N^2}{(\Omega_{me} L_d)^2} \quad (4.19)$$

Si possono ora rappresentare graficamente i limiti di corrente e tensione. Il primo, nel piano I_d - I_q , descrive una circonferenza di raggio I_N , centrata nell'origine. Il limite di tensione, invece, descrive una insieme di ellissi concentriche, i cui assi si rimpiccioliscono al crescere della velocità meccanica-elettrica Ω_{me} . Quest'ultima corrisponde alla velocità di rotazione dei vettori spaziali a regime e, quindi, alla pulsazione di tensioni e correnti statoriche.

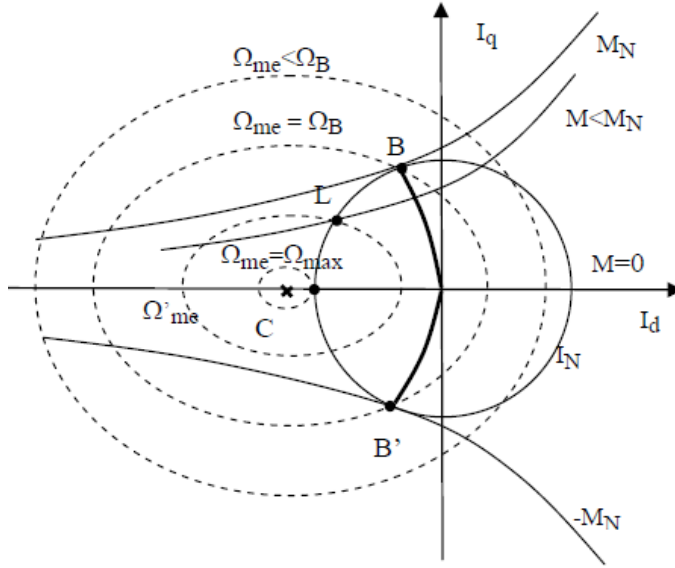


Fig. 4.6 - Rappresentazione grafica dei limiti di tensione e corrente nel piano $I_d - I_q$

Le ellissi sono centrate nel punto che ha coordinate $(-\frac{\Lambda_{mg}}{L_d}, 0)$. Quando $|\frac{\Lambda_{mg}}{L_d}| < I_N$, il centro cade all'interno della circonferenza limite di corrente, pertanto la velocità massima del motore, dal punto di vista puramente elettrico, è infinita. In caso contrario, come accade per i motori AMK, per i quali vale:

$$\left| \frac{0.0296 \text{ Vs}}{0.00012 \text{ H}} \right| = 246.67 \text{ A} > 41\sqrt{2} \text{ A}$$

la massima velocità raggiungibile si calcola, imponendo $I_q = 0$ e $I_d = -I_{d,max}$, dove $I_{d,max}$ è il massimo valore di corrente di asse diretto che può essere sopportato dal motore senza smagnetizzarsi:

$$\Omega_{max} = \frac{U_N}{\Lambda_{mg} - L_d I_{d,max}} \quad [\text{rad.elettrici/s}] \quad (4.20)$$

Essa è dipendente dalla tensione U_N e quindi anche dalla tensione lato DC. Può valere $I_{d,max} = I_N$ oppure la massima corrente smagnetizzante può essere limitata ad un valore inferiore.

Un altro valore di velocità importante da calcolare è la velocità base:

$$\Omega_B = \frac{U_N}{\sqrt{(\Lambda_{mg} + L_d I_d)^2 + (L_q I_q)^2}} \quad [\text{rad.elettrici/s}] \quad (4.21)$$

Rielaborando l'equazione della coppia m , si ricava che i luoghi a coppia costante, nel piano $I_d - I_q$, sono delle iperboli:

$$I_q = \frac{2}{3p} \frac{m}{\Lambda_{mg} + (L_d - L_q) I_d} \quad (4.22)$$

Il funzionamento del motore, ad una data velocità e con una data coppia, è descritto da un certo punto interno al limite di corrente e dato dall'intersezione tra l'iperbole isocoppia ed il limite di tensione corrispondente alla data velocità. Alle basse velocità

di funzionamento, il limite di tensione è molto ampio e pertanto è più restrittivo il limite di corrente. In tali condizioni appare conveniente far funzionare il motore in modo che la corrente assorbita sia minima a parità di coppia erogata (curve che uniscono i punti B e B' con l'origine degli assi), riducendo quindi al minimo le perdite. Questo funzionamento del motore prende il nome di MTPA (Max Torque Per Ampere) e si può attuare dalla velocità nulla fino alla velocità base. In corrispondenza di quest'ultima, la corrispondente ellisse che esprime il limite di tensione passa per i punti B e B'. Al di sotto della velocità base è sempre disponibile la coppia nominale del motore.

Superata Ω_B , si può far aumentare la velocità di rotazione spostandosi, nel piano I_d - I_q , lungo la circonferenza corrispondente alla corrente nominale. Occorre quindi decrementare la corrente I_q per poter far scorrere una corrente I_d negativa, mantenendo costante il modulo della corrente. La I_d negativa produrrà un flusso diretto in senso opposto a quello del magnete. Indebolendo il flusso dei magneti, la velocità può salire. Si parla pertanto di funzionamento in deflussaggio.

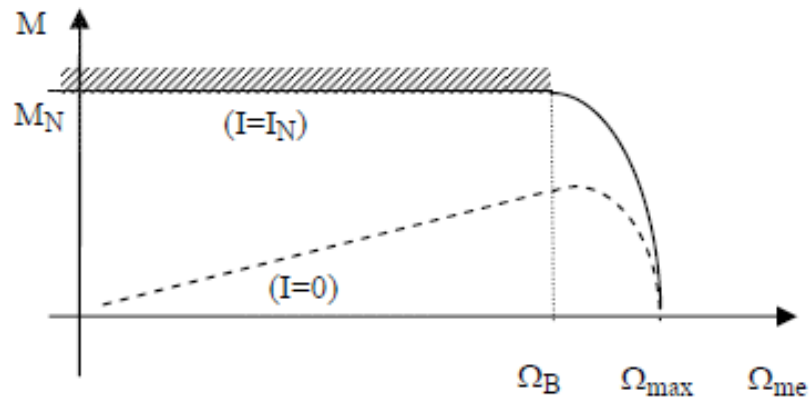


Fig. 4.7 - Caratteristica meccanica del motore brushless. La coppia nominale M_N è sempre disponibile al di sotto della velocità base Ω_B

È importante sottolineare come le velocità caratteristiche Ω_B e Ω_{max} , sopra calcolate, siano funzione della tensione lato DC poiché ad essa corrisponde una certa ampiezza massima del vettore spaziale di tensione. La caratteristica meccanica, quindi, non rimane sempre uguale durante il funzionamento del motore, poiché ad esempio la velocità base decresce al diminuire dello stato di carica dell'accumulatore che alimenta il DC bus. Occorre quindi misurare costantemente, tramite un opportuno sensore, la tensione lato DC.

L'azionamento, infine, lavora sempre nel secondo o nel terzo quadrante del piano $I_d - I_q$, a seconda che il funzionamento richiesto sia da motore o da freno.

5 Simulazione degli algoritmi di controllo

L'algoritmo di controllo per motore brushless anisotropo è stato implementato in ambiente Simulink. Sono stati creati due diversi modelli, "Control.slx" per simulare il controllo con inverter e motore idealizzati e "Power_stage.slx", che modella con più precisione lo stadio di conversione della potenza (IGBTs e motore). Come ci si può aspettare, questa ultima simulazione è computazionalmente molto più onerosa della prima. Essa è usata per collaudare l'algoritmo di Space Vector Modulation e per verificare il dimensionamento della capacità di DC-link, mentre la taratura dei regolatori viene verificata con "Control.slx". Entrambi i modelli richiedono di compilare il file Matlab "Initialization.m" prima di poter eseguire la simulazione.

5.1 L'algoritmo di controllo AMK

Il controllo adottato dalla AMK è un controllo di velocità con limitazione dinamica della coppia. Sono quindi presenti gli anelli di corrente ma anche un anello esterno di velocità. Lo schema prevede quattro regolatori del tipo proporzionale-integrale (PI). In ingresso l'utente fornisce il riferimento di velocità, che in accelerazione è posto pari alla massima velocità che si desidera il motore raggiunga, mentre in frenata o a riposo è pari a zero. Una volta calcolato l'errore tra riferimento e velocità misurata attraverso l'encoder, questo viene dato in ingresso al regolatore di velocità che lo elabora e calcola il riferimento di coppia. Quest'ultimo è soggetto a limitazione secondo le richieste del pilota, tramite acceleratore e freno. La coppia richiesta viene successivamente trasformata, tramite la costante di coppia K_t , in un riferimento di corrente di asse q che viene dato in ingresso ad un altro regolatore PI. Alle tensioni di asse q e d richieste dai rispettivi regolatori vengono successivamente sommati dei termini, probabilmente di disaccoppiamento e compensazione della forza contro elettromotrice in feedforward (vedi "Voltage precontrol" in figura). Infine, dai valori di tensione vengono ricavati i duty-cycle da applicare agli IGBTs.

Si noti la presenza di un ulteriore regolatore, detto "di tensione", in quanto controlla che le tensioni richieste all'invertitore non superino il limite fisico della tensione disponibile al DC bus. Se questo limite viene superato, esso genera un riferimento di corrente di asse diretto negativa per ridurre la forza contro elettromotrice del motore. Al di fuori di questa particolare situazione, il cui funzionamento verrà descritto nel paragrafo 5.2.3, il riferimento di i_d vale sempre zero.

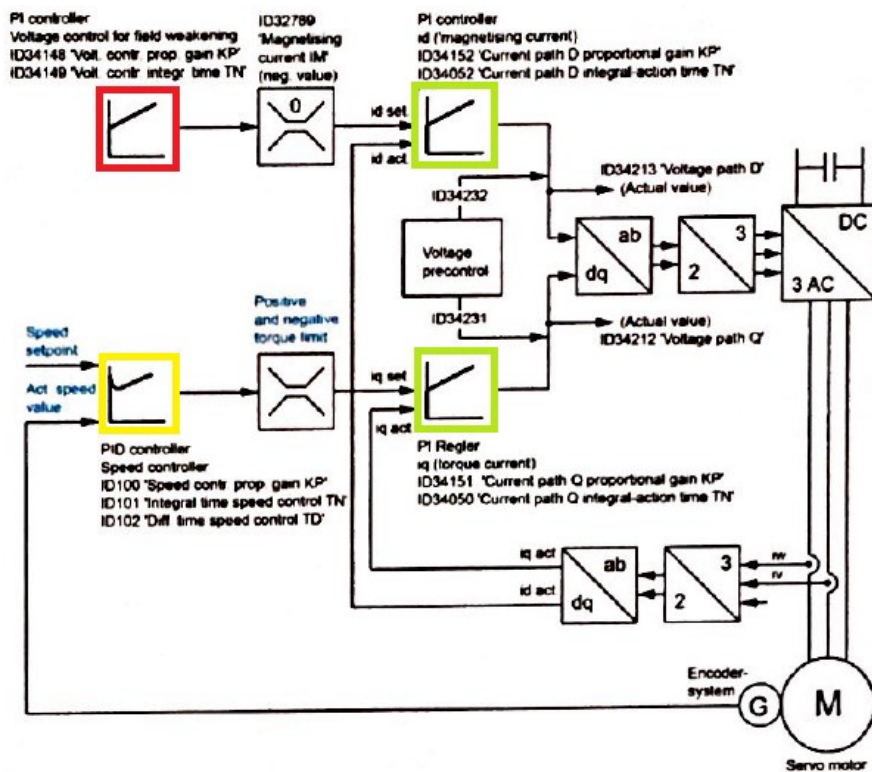


Fig. 5.1 – Schema a blocchi dell’algoritmo di controllo dell’azionamento AMK: in verde sono evidenziati i regolatori PI di corrente di asse D (sopra) e Q (sotto), in giallo il regolatore di velocità, in rosso il regolatore di tensione per il deflussaggio

Un approccio di questo tipo consente di limitare la massima velocità di rotazione del motore, indipendentemente dal carico applicato all’albero. Ad una velocità prossima a quella massima, infatti, il regolatore di velocità interviene richiedendo agli anelli di corrente una coppia inferiore da quella richiesta dal pilota tramite acceleratore e freno.

Un altro vantaggio è rappresentato dalla possibilità di variare dinamicamente il riferimento e quindi la velocità massima, in sinergia con un algoritmo di torque-vectoring.

La centralina della vettura formula SAE esegue le sue operazioni con una cadenza di 100 Hz. Ogni 10 ms, quindi, essa legge i dati provenienti dai potenziometri di acceleratore e freno e li converte in richieste di coppia e velocità da comunicare ai quattro inverters, via CAN. Le variabili che la centralina invia sono contenute nel pacchetto “AMK Setpoints 1”:

1. AMK_TargetVelocity: riferimento di velocità
2. AMK_TorqueLimitPositiv: limite positivo di coppia
3. AMK_TorqueLimitNegativ: limite negativo di coppia

Queste variabili, che di fatto costituiscono i riferimenti per l’azionamento, vengono gestite come riportato nel seguito.

Tabella 5.1

Accelerazione	AMK_TargetVelocity = velocità massima desiderata AMK_TorqueLimitPositiv = massima coppia positiva AMK_TorqueLimitNegativ = 0
Frenata	AMK_TargetVelocity = 0 AMK_TorqueLimitPositiv = 0 AMK_TorqueLimitNegativ = massima coppia frenante (negativa)

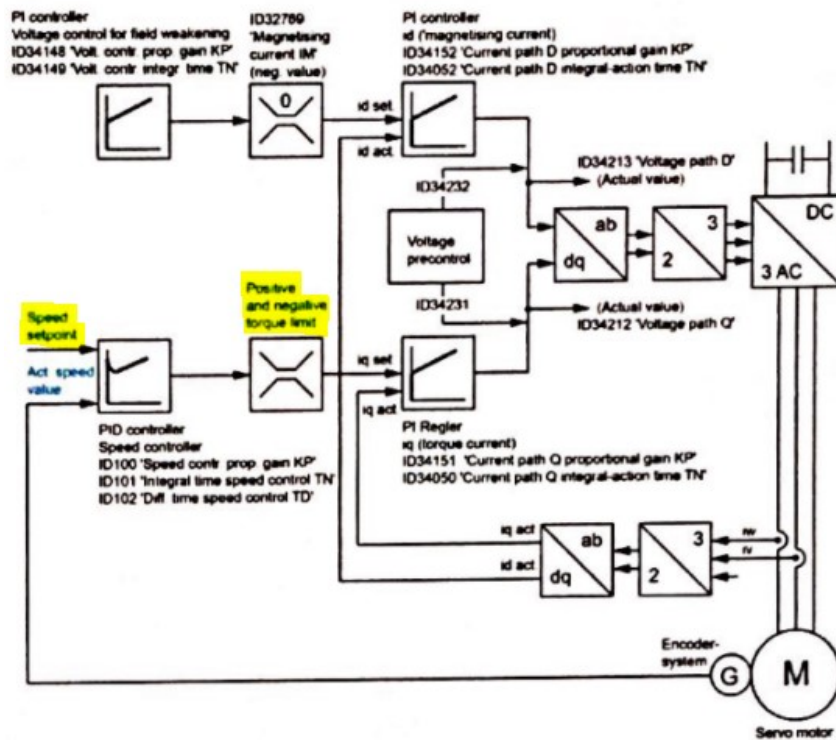


Fig. 5.2 – Sono evidenziati in giallo i punti dello schema a blocchi nei quali intervengono i riferimenti inviati all’inverter via CAN

Dall’immagine soprastante si nota come i motori, nonostante la loro anisotropia ($L_d \neq L_q$) vengano di fatto controllato come delle macchine isotrope, cioè si regola la coppia solo attraverso la corrente di asse q (I_q). Oltre a semplificare il controllo, questo approccio probabilmente è stato scelto poiché i motori in questione hanno delle induttanze L_d ed L_q molto piccole, conseguentemente la coppia di riluttanza è poco apprezzabile ai bassi carichi.

5.2 Modello Simulink del controllo

La prima simulazione serve soltanto per validare l’algoritmo di controllo, senza valutare l’effetto della tecnica di modulazione. Dato che poi l’algoritmo andrà implementato nel microcontrollore, quindi in un sistema a tempo discreto, la simulazione viene eseguita in modalità “Fixed-step” ovvero ad intervalli di tempo

fissati e pari al periodo di campionamento del controllo T_s . Si può verificare ciò accedendo a Model Settings > Model Settings > Solver.

L'algoritmo di controllo ha una struttura molto simile a quello adottato da AMK, ovvero si tratta di un controllo di velocità con limitazione dinamica della coppia. È tuttavia possibile escludere il regolatore di velocità (tramite uno switch nel modello simulink e cambiando il valore di una variabile nel codice in C) per poter passare ad un puro controllo di coppia. Al fine di sfruttare l'anisotropia del motore, dal riferimento di coppia (calcolato dal regolatore di velocità o direttamente imposto dalla centralina) si ricavano i riferimenti di corrente di entrambi gli assi. In altre parole, su scelta dell'utente, è possibile mantenere la corrente di asse d ad un valore diverso da zero anche al di fuori del funzionamento in deflussaggio, contrariamente a quanto fatto da AMK.

I riferimenti di coppia e velocità, periodicamente inviati dalla centralina al microcontrollore che gestisce l'inverter, vengono gestiti come indicato in tabella, ovvero in maniera identica a quanto fa AMK.

Tabella 5.2

Situazione di guida	Riferimento di velocità	Limite positivo di coppia	Limite negativo di coppia
Accelerazione	Velocità massima	Proporzionale all'acceleratore	0
Frenata	0	0	Proporzionale al freno
Motore in folle	Qualsiasi	0	0

Mantenendo in saturazione l'anello di velocità, il controllo che si ottiene è, di fatto, un controllo di coppia, dato che la coppia chiesta dal regolatore di velocità è sempre pari a quella massima, quando la velocità effettiva non è prossima al riferimento, ed è quindi facile modulare la coppia effettiva richiesta agli anelli di velocità tramite un limitatore in uscita dal regolatore. In accelerazione, man mano che il motore si avvicina al riferimento di velocità, a comandare sarà proprio l'output del regolatore che richiederà una coppia via via minore. Lo stesso accade, però in senso opposto, in frenata, per velocità prossime a zero.

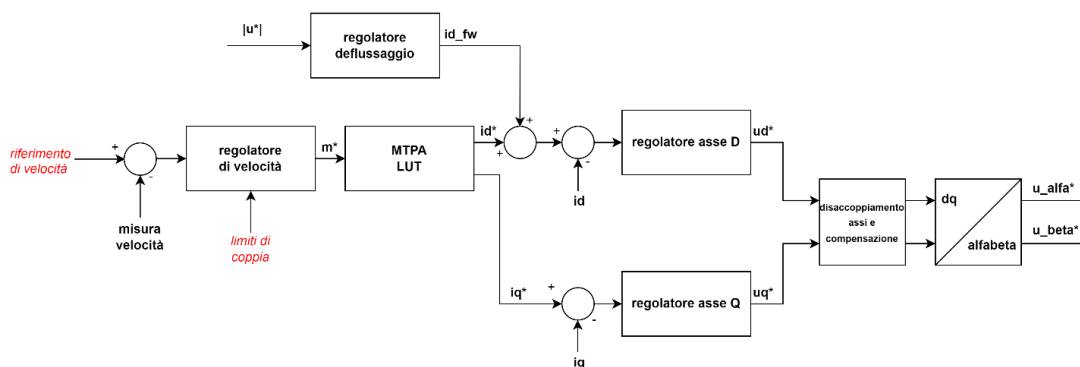


Fig. 5.3 - Schema a blocchi dell'algoritmo di controllo nel classico schema ad Orientamento di Campo (Field Oriented Control, FOC)

I riferimenti, nel modello Simulink, vengono generati da un apposito subsystem che simula la centralina presente in vettura. Questa lavora ad interrupts, generati ogni 10 ms da un timer al suo interno, per cui il subsystem lavora anch'esso a 100 Hz. Si simula una richiesta di coppia da parte del pilota e vengono generati degli appositi riferimenti secondo la tabella riportata sopra.

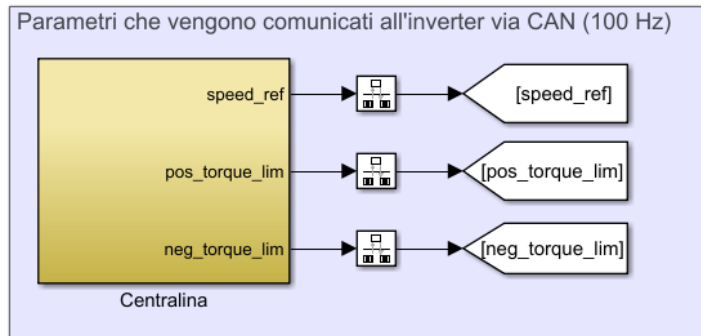


Fig. 5.4 - Subsystem che modella la centralina

5.2.1 I regolatori PI

Tutti i regolatori usati in questa tesi sono del tipo proporzionale – integrale (PI). Dapprima si considerano i regolatori a tempo continuo. Sia $x(t)$ l'ingresso all'istante t , allora l'uscita del regolatore $y(t)$ è data dalla somma di due termini, uno detto proporzionale $p(t)$ e l'altro detto integrale $i(t)$:

$$y(t) = K_p \varepsilon(t) + K_I \int \varepsilon(t) dt = p(t) + i(t) \quad (5.1)$$

dove K_p e K_I sono due costanti, chiamate rispettivamente guadagno proporzionale e guadagno integrale.

L'ingresso $x(t)$ del regolatore solitamente è una funzione errore $\varepsilon(t) = r(t) - f(t)$, dove $r(t)$ è il segnale di riferimento e $f(t)$ il segnale in retroazione, ottenuto tramite stime o misure sul sistema controllato tramite un sensore. L'azione integrale, se presente, garantisce che l'errore a regime ad un riferimento a gradino sia nullo. L'equazione di cui sopra, scritta nel dominio di Laplace, diventa:

$$Y(s) = \left(K_p + \frac{K_I}{s} \right) \varepsilon(s) = R(s) \varepsilon(s) \quad (5.2)$$

Solitamente, i regolatori sono implementati all'interno di sistemi a tempo discreto, quali sono i microcontrollori. Un algoritmo a tempo discreto viene eseguito ad intervalli regolari, pari al tempo di campionamento T_s . Occorre quindi descrivere la funzione di trasferimento del regolatore PI in un sistema a tempo discreto. Se $f(kT_s)$ è il campione k -esimo della generica funzione f , la parte proporzionale

$$p(t) = K_p \varepsilon(t) \quad (5.3)$$

diventa

$$p(kT_s) = K_p \varepsilon(kT_s) \quad (5.4)$$

L'integrale

$$i(t) = K_I \int \varepsilon(t) dt \quad (5.5)$$

invece diventa

$$i(kT_s) = K_I T_s \varepsilon(kT_s) + i[(k-1)T_s] \quad (5.6)$$

Questa tecnica è chiamata integrazione rettangolare con il metodo di Eulero all'indietro. L'integrale della funzione $\varepsilon(t)$, infatti, viene approssimato tramite una serie di rettangoli aventi base T_s ed altezza pari al valore di ε all'istante di campionamento. Accanto a questo metodo esistono anche le tecniche dell'integrazione bilineare e tramite Eulero in avanti.

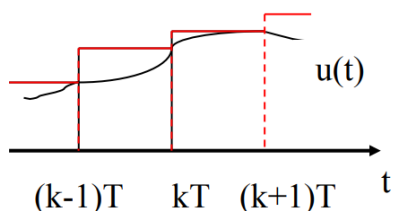


Fig. 5.5 - Rappresentazione grafica dell'integrazione a tempo discreto (con tempo di campionamento T) di un segnale continuo $u(t)$ tramite il metodo di Eulero all'indietro

Per indicare che un campione è stato prelevato all'istante $(k-1)T_s$ si può usare l'operatore z^{-1} . La funzione di trasferimento del regolatore PI, nella variabile z , diventa quindi

$$R(z) = K_p + \frac{K_I T_s}{1 - z^{-1}} \quad (5.7)$$

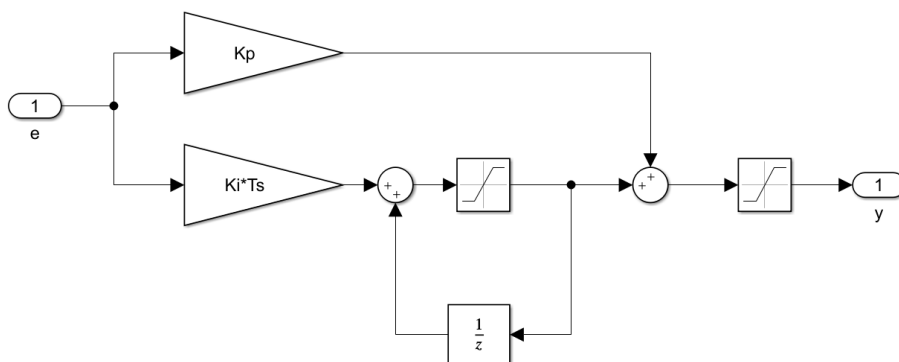


Fig. 5.6 - Regolatore PI implementato all'interno di un subsystem in ambiente Simulink

In figura si può notare la presenza di due limitatori. Quello immediatamente a monte dell'uscita serve ad evitare che il controllore richieda un valore della grandezza controllata, ad esempio la coppia, troppo elevato poiché il sistema non la può erogare. Il limitatore collocato lungo il percorso integrale assume la funzione di "anti-

windup statico”: onde evitare che l’integrale dell’errore assuma valori spropositatamente elevati, causando oscillazioni attorno al riferimento, si limita il valore che l’integrale può assumere.

In questo lavoro di tesi, i limiti dell’integratore e dell’uscita vengono posti uguali e costanti. Unicamente il regolatore di velocità assume come valori limite, per l’anti-windup e per il limitatore dell’uscita, i limiti di coppia richiesti dalla centralina, che variano nel tempo.

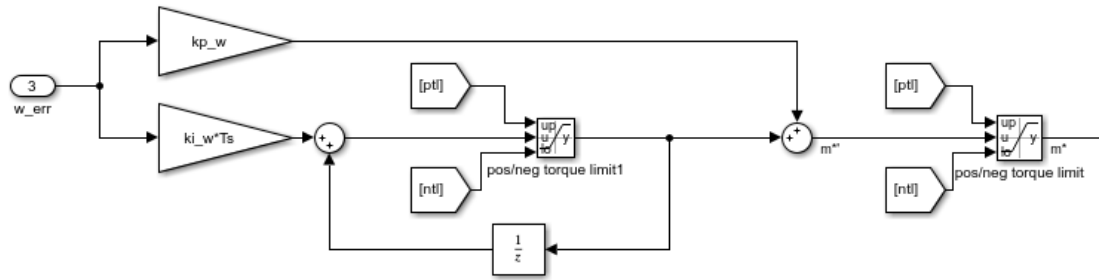


Fig. 5.7 - Dettaglio del regolatore di velocità

5.2.2 Funzionamento in MTPA

Come già visto, l’equazione della coppia elettromagnetica è:

$$m = \frac{3}{2}p\Lambda_{mg}I_q + \frac{3}{2}p(L_d - L_q)I_dI_q \quad (5.8)$$

Si nota come l’equazione sia composta da due termini: il primo è dovuto alla presenza dei magneti permanenti e prende il nome di “coppia di allineamento”, il secondo è dovuto all’anisotropia del circuito magnetico e prende il nome di “coppia di riluttanza”. Per sfruttare entrambe le componenti di coppia occorre iniettare negli avvolgimenti statorici entrambe le componenti I_q ed I_d di corrente. Il segno di I_q è lo stesso della coppia mentre I_d assume valori negativi compresi tra zero e $-I_{d,max}$, avendo quindi un effetto smagnetizzante.

Il controllo progettato da AMK prevede il regolatore di corrente asse d direttamente in cascata al regolatore di tensione. Se ne deduce che la corrente I_d non viene utilizzata per generare coppia ma soltanto per deflussare il motore, per velocità superiori a quella base. Al di sotto di essa, essendo mantenuta a zero la I_d , la coppia è quindi direttamente proporzionale alla corrente I_q :

$$m = \frac{3}{2}p\Lambda_{mg}I_q = K_tI_q \quad (5.9)$$

dove $K_t = 0.222 \text{ Nm/A}$ è la costante di coppia. Questo tipo di controllo risulta molto facile da gestire ma sfrutta soltanto la componente cilindrica della coppia.

Facendo uso dei valori di induttanza apparente indicati nel datasheet, è possibile calcolare, per ogni valore del modulo di corrente I , la combinazione di I_d ed I_q che

massimizza la coppia, sfruttando la componente di riluttanza. Le componenti del vettore spaziale di corrente I sono:

$$\begin{cases} I_d = I \cos(\alpha) \\ I_q = I \sin(\alpha) \end{cases} \quad (5.10)$$

dove α è l'angolo del vettore spaziale di corrente.

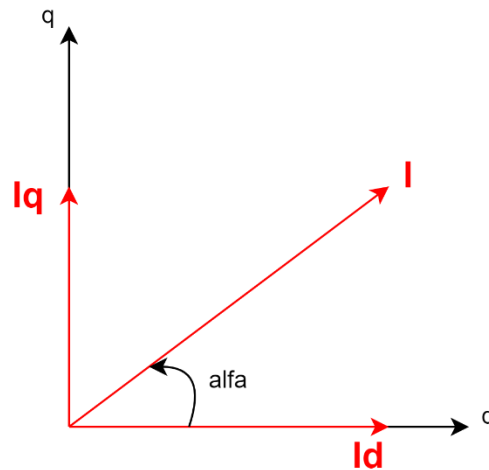


Fig. 5.8 - Vettore spaziale di corrente I nel piano dq

Sostituendo nell'equazione della coppia, si ottiene:

$$m = \frac{3}{2} p [\Lambda_{mg} I \sin(\alpha) + (L_d - L_q) I^2 \cos(\alpha) \sin(\alpha)] \quad (5.11)$$

Si ricerca α che massimizza la coppia, per una data I , per cui si deriva la funzione rispetto ad α e si pone la derivata pari a zero:

$$\frac{dm}{d\alpha} = 0 \Rightarrow \alpha = \cos^{-1} \left(\frac{-\Lambda_{mg} + \sqrt{\Lambda_{mg}^2 + 8(L_d - L_q)^2 I^2}}{4(L_d - L_q)I} \right) \quad (5.12)$$

Per ogni ampiezza I del vettore spaziale di corrente si può quindi trovare α che massimizza la coppia. Lo script Matlab "MTPA_full.m" provvede ad eseguire il calcolo per una serie di valori di corrente.

Con il metodo analitico, tuttavia, si trascura il fenomeno della saturazione dei circuiti magnetici del motore, il quale porta a non avere più una relazione lineare flusso concatenato - corrente. Queste relazioni si possono considerare lineari soltanto per bassi valori di corrente. A rigore, quindi, non è corretto considerare le induttanze L_d , L_q costanti.

Nel motore reale compaiono infatti le induttanze dette "differenziali", i cui valori possono essere ben diversi dalle induttanze apparenti L_d e L_q , in quanto calcolate, in ogni punto della curva $\Lambda - I$, come derivata della curva. Il luogo MTPA reale

pertanto, soprattutto per elevati valori di coppia e, quindi, di corrente, potrebbe discostarsi sensibilmente da quello analitico.

Per ottenere un controllo preciso del motore, occorre conoscerne il comportamento considerando anche la saturazione dei circuiti magnetici. Per ovviare a ciò, è possibile far uso dei parametri progettuali del motore, ad esempio tramite il calcolo agli elementi finiti, oppure si può caratterizzare la macchina tramite banco prova.

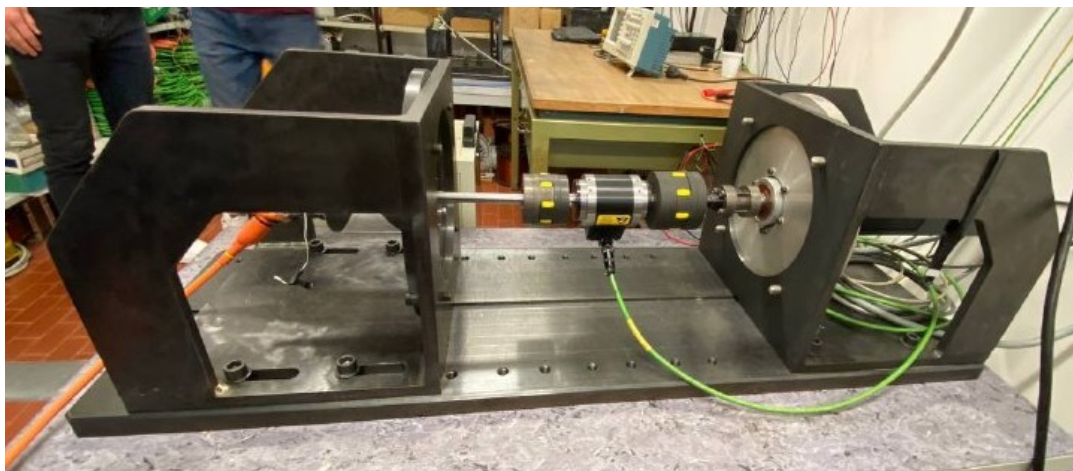


Fig. 5.9 - Banco prova "lento" del laboratorio di azionamenti (EdLab): a sinistra il motore AMK, a destra il Master Motor, in mezzo il giunto con torsionmetro

La prova a banco prevede di collegare l'albero del motore in prova (MUT, Motor Under Test) ad un altro motore (MM, Master Motor), controllato in velocità. Il MUT viene alimentato da un inverter in grado di erogare varie combinazioni di correnti I_d, I_q . Tramite un torsionmetro si misura la coppia erogata dal MUT.

La procedura si svolge nel seguente modo. Innanzitutto, l'operatore da software definisce l'ampiezza del vettore spaziale di corrente (e quindi della corrente di fase) con la quale eseguire la prova, dopodiché il MM porta in rotazione il MUT ad una velocità bassa e prefissata (0.159 rad/s). Nel frattempo, l'inverter che controlla il MUT inietta nelle fasi B e C una corrente continua di valore prefissato. Dato che il MUT viene trascinato, la coppia da esso erogata, opportunamente filtrata per togliere le oscillazioni dovute al controllo ed alla PWM e salvata nella memoria del computer, varia in maniera circa sinusoidale (a causa della sovrapposizione dei termini cilindrico e di riluttanza). Essa è nulla quando il vettore spaziale di corrente, associato alla corrente continua iniettata, è allineato con l'asse d e massima nel punto di MTPA; questo consente di individuare l'angolo meccanico e, pertanto, anche quello elettrico che consente di massimizzare la coppia. Terminato un giro meccanico, il computer che gestisce il banco calcola le coordinate del vettore spaziale che massimizza la coppia e, successivamente, inietta nel MUT una terna di correnti che realizza il suddetto vettore spaziale e misura la coppia erogata.

La procedura viene ripetuta per diverse ampiezze di corrente, a gradini di 5 A. Occorre prestare attenzione poiché l'ampiezza I_{DC} della corrente DC iniettata non corrisponde all'ampiezza del vettore spaziale. Se si suppone di alimentare le fasi B e C con una corrente pari a $I_b = I_{DC}$ e $I_c = -I_{DC}$, eseguendo la trasformazione $T_{abc \rightarrow \alpha\beta 0}$, si ricava che l'ampiezza del vettore spaziale corrispondente è $\frac{2}{\sqrt{3}} I_{DC}$.

Alimentare le fasi B e C con una corrente continua corrisponde ad iniettare nel motore un vettore spaziale di corrente costante nel sistema di riferimento $\alpha\beta$, avente componente lungo l'asse β soltanto. Nel sistema dq, invece, il vettore spaziale ruota con una pulsazione proporzionale alla velocità impressa al MUT dal MM.

Durante il test in laboratorio sono sorti alcuni problemi. Innanzitutto, si è reso necessario usare due diversi schemi di alimentazione, poiché il motore AMK richiede valori relativamente elevati di corrente per poter funzionare a coppia massima. Una prima modifica al setup originario è stata fatta sostituendo l'inverter del MUT con un alimentatore DC da 60 A.

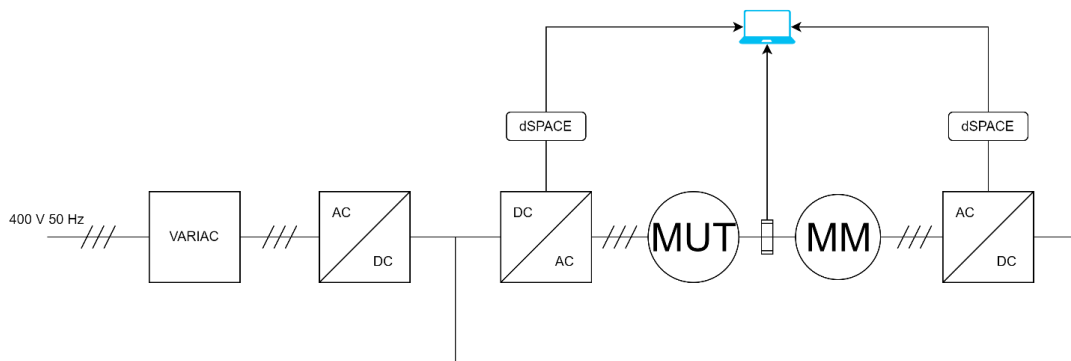


Fig. 5.10 - Setup del banco prova per correnti di fase fino a 30 A picco

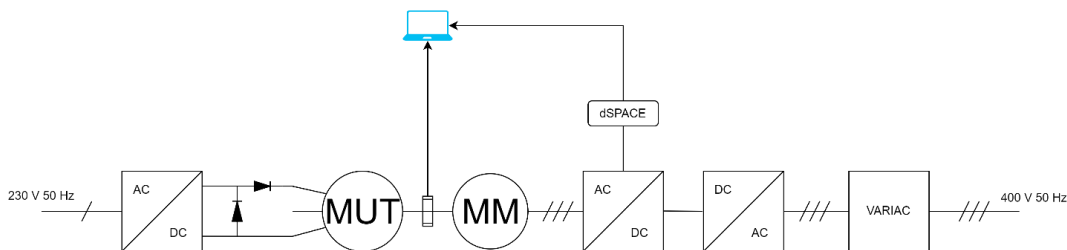


Fig. 5.11 - Setup del banco prova per correnti fino a 60 A DC. I due diodi in uscita dall'alimentatore DC servono per evitare la circolazione di corrente inverse dovute alla FCEM indotta dai magneti permanenti del MUT

Il secondo problema, ben più difficile da risolvere, consiste nel fatto che adottando la procedura sopra descritta, il vettore spaziale della corrente iniettata nel MUT ruota, allineandosi in maniera variabile col rotore. Il motore AMK è caratterizzato da una corrente massima di 105 A rms, mentre la massima corrente smagnetizzante dichiarata è di 35 A rms. Se si eseguisse la procedura di test con correnti superiori, in ampiezza, a $35\sqrt{2}$ A, si rischierebbe, per determinati orientamenti del vettore spaziale di corrente, l'immediata smagnetizzazione dei magneti permanenti situati nel rotore, con conseguenze catastrofiche. Si è scelto di eseguire quindi il test con una corrente I_{DC} massima di 40 A, la quale trasformata corrisponde a di 46.2 A, valore leggermente inferiore a $35\sqrt{2}$ A. In tal modo, anche quando il vettore spaziale è allineato con l'asse d e quindi tutta la corrente agisce lungo l'asse diretto, non si rischia di danneggiare irreversibilmente i magneti.

Per risolvere il problema dell'alimentazione che non è in grado di fornire una corrente sufficiente,

si può pensare di usare, una volta pronto, il prototipo di inverter, alimentandolo tramite l'accumulatore ad alta tensione, dato che esso può tranquillamente erogare i 148.5 A massimi richiesti dal motore. Il problema legato alla smagnetizzazione, invece, può essere affrontato nel seguente modo. Il motore AMK deve essere collegato ad un motore controllato in velocità, secondo l'ormai noto schema. Fissato un valore di corrente, si impone prima una $I_q = I$, poi si va a variare α tramite l'interfaccia di debug di CCS, facendo quindi ruotare il vettore spaziale di corrente. Occorre però assicurarsi sempre che la componente di asse diretto sia maggiore del valore massimo tollerato dal motore. Con un torsionometro si misura la coppia erogata dal MUT al variare di α , rispettando i limiti del motore. Infine, si va a selezionare, per una data I , l'angolo che massimizza la coppia. La procedura va ripetuta fino al raggiungimento della coppia massima.

Nonostante la breve durata delle prove, si è vista aumentare rapidamente la temperatura del MUT, monitorata attraverso il termistore presente al suo interno. Per questo motivo, si consiglia di raffreddare a liquido il motore durante il sopracitato test.

È possibile infine tracciare i luoghi MTPA ricavati analiticamente e tramite misure.

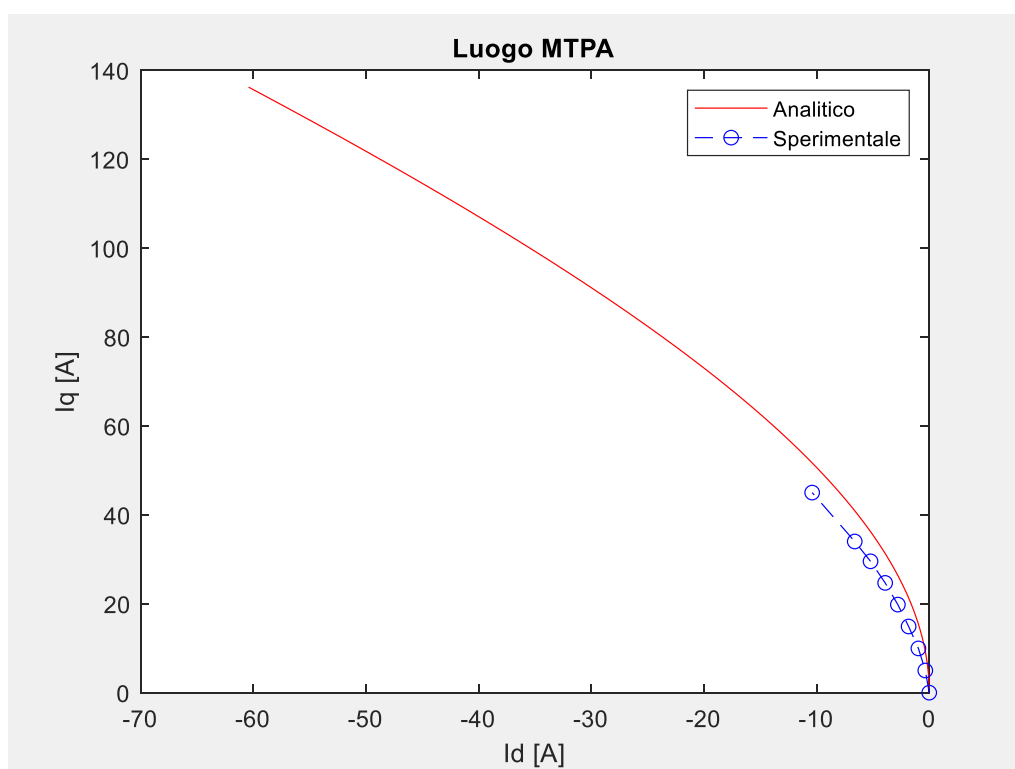


Fig. 5.12 - Confronto tra luogo MTPA calcolato in modo analitico tramite le induttanze apparenti e luogo ottenuto da misure a banco

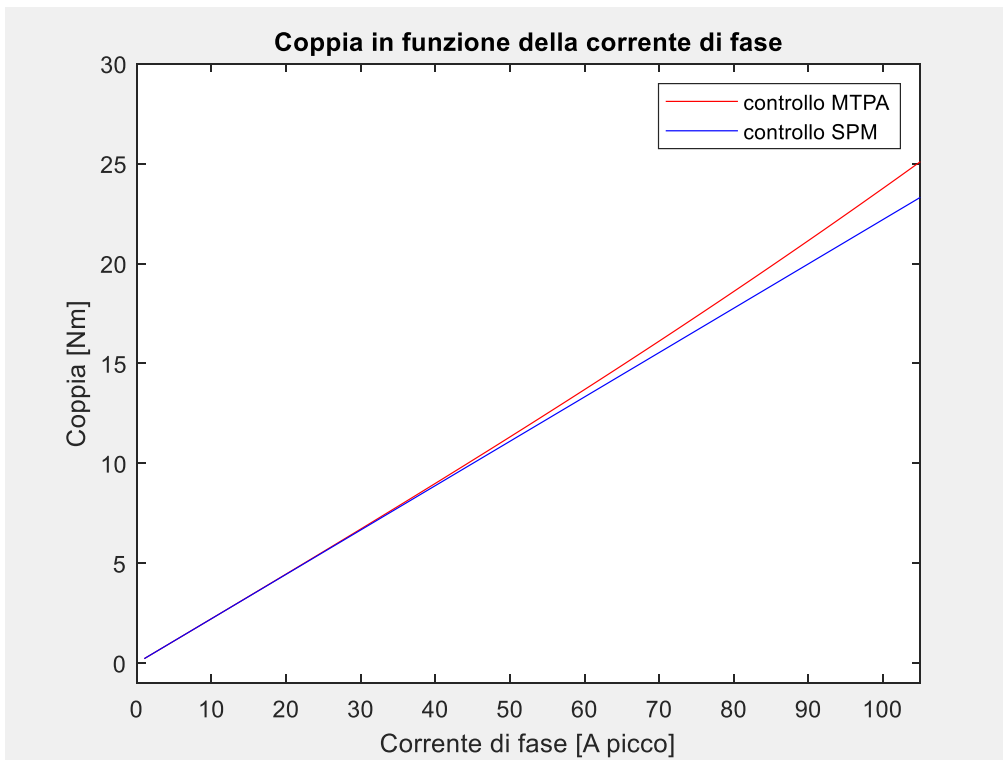


Fig. 5.13 - Relazione coppia elettromagnetica-corrente di fase con (MTPA) e senza (SPM) il controllo della corrente di asse diretto per la generazione della coppia. Il secondo tipo di controllo è tipicamente utilizzato con le macchine a magneti permanenti superficiali e pertanto qui viene indicato con la sigla SPM

Dalla figura sopra si nota che, tranne con elevato carico, la coppia “aggiuntiva” dovuta al controllo della corrente di asse d è minima. Potrebbe tuttavia essere possibile che test sotto carico evidenzino fenomeni di saturazione del ferro, qui trascurati, tali da rendere vantaggioso adottare la strategia di controllo di entrambi gli assi per produrre coppia.

Dalla traiettoria MTPA così ricavata si calcolano per interpolazione le I_d , I_q necessarie a produrre una certa coppia m . Una Matlab Function trasforma la richiesta di coppia nei corrispondenti riferimenti di corrente dei due assi. Al di sotto della velocità base, il punto di funzionamento del motore si trova sempre lungo la traiettoria MTPA per minimizzare la corrente assorbita, data una certa coppia erogata.

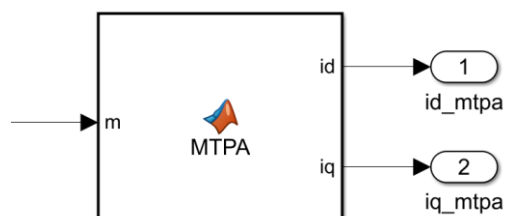


Fig. 5.14 - Generazione dei riferimenti di corrente a partire dalla richiesta di coppia m

5.2.3 Funzionamento in deflussaggio

Al di sopra della velocità base, il motore non può più lavorare in MTPA, in quanto una componente della corrente di statore deve essere deputata al deflussaggio dei magneti permanenti. La FCEM indotta dalla rotazione del rotore diventa così elevata che l'inverter non è più in grado di far scorrere correnti adeguate per il controllo del motore. Occorre quindi ridurre il flusso dei magneti permanenti, iniettando negli avvolgimenti di statore una corrente I_d smagnetizzante, cioè negativa.

Lo schema per il controllo in deflussaggio è preso ed adattato da [1]. Un regolatore di tipo I (integratore puro) monitora l'ampiezza del vettore spaziale di tensione \mathbf{u}^* che viene applicato ai morsetti dell'inverter. \mathbf{u}^* corrisponde al valore di picco della tensione di fase (u_a, u_b, u_c). Ricordando che il riferimento di tensione \mathbf{u}^* si ottiene dalla somma vettoriale di u_α^* ed u_β^* , si calcola:

$$u^* = |\mathbf{u}^*| = \sqrt{(u_\alpha^*)^2 + (u_\beta^*)^2} \quad (5.13)$$

La massima tensione che l'inverter può fisicamente applicare ai morsetti del motore, come ampiezza del vettore spaziale, è $\frac{V_{dc}}{\sqrt{3}}$, operando con la Space Vector Modulation. Se si ponesse tale valore come soglia da monitorare (U_{lim}), il regolatore interverrebbe quando la tensione è praticamente già saturata al valore massimo. Occorre quindi confrontare u^* con $U_{lim} < \frac{V_{dc}}{\sqrt{3}}$. Si è pertanto scelta $U_{lim} = 0.9 \frac{V_{dc}}{\sqrt{3}}$ cioè il controllo ha disposizione il 10% di $\frac{V_{dc}}{\sqrt{3}}$ per "accorgersi" che il motore sta superando la velocità base e conseguentemente innescare il meccanismo di deflussaggio.

Il regolatore prende in ingresso i riferimenti di tensione u_α^* ed u_β^* , calcolati dal controllo all'istante temporale k . Nella routine di controllo il microcontrollore però dispone dei riferimenti di tensione calcolati all'istante successivo ($k+1$), per cui nel Simulink sono stati inseriti i blocchetti "memoria" che ritardano di un periodo di campionamento i segnali relativi ad u_α^* e u_β^* .

Il sistema ne calcola poi il modulo $|\mathbf{u}^*|$ e lo confronta con U_{lim} . L'errore viene calcolato come $\varepsilon_u = u^* - U_{lim}$ in modo che, se la tensione richiesta è maggiore della soglia, venga generato un riferimento di corrente I_d negativo, cioè smagnetizzante.

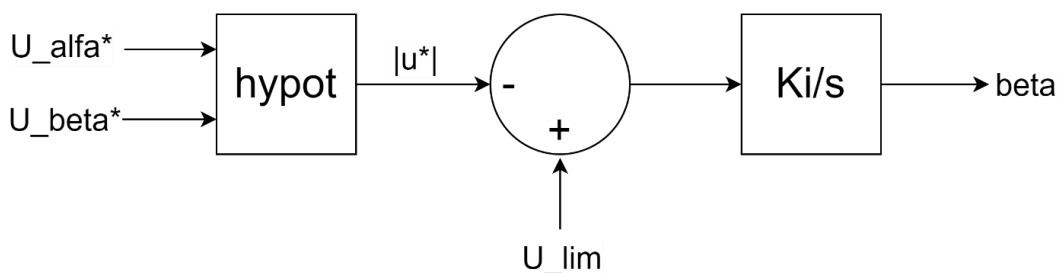


Fig. 5.15 – Regolatore di tensione

Il regolatore restituisce un parametro β variabile tra 1 e 0. Questo viene poi usato per calcolare il valore i_d^* richiesto al regolatore di corrente di asse d:

$$i_d^* = \beta i_d^{MTPA} + (1 - \beta) i_d^{fw} \quad (5.14)$$

i_d^{MTPA} è il valore di I_d calcolato dal blocco MTPA, mentre i_d^{fw} è posta pari alla massima corrente smagnetizzante tollerabile dal motore, indicata come $I_{d,max}$. Si riconosce quindi come, nel funzionamento in MTPA, il valore di β sia unitario, mentre β nullo corrisponde al massimo deflussaggio. Sia l'azione integrale che l'uscita del regolatore sono limitati tra $[0,1]$, tuttavia il valore iniziale dell'integrale va messo ad 1, onde evitare che all'accensione β valga 0 finché l'integrale non satura, con un conseguente picco di corrente negativa.

Si potrebbe pensare che la U_{lim} assume un valore costante, pari a $0.9 \frac{V_{dc}}{\sqrt{3}}$. Tuttavia, occorre tenere conto che la U_{lim} con accumulatore ad alta tensione a piena carica vale $U_{lim} = 0.9 \frac{600}{\sqrt{3}} = 311.77 \text{ V}$, mentre la tensione nominale concatenata del motore di 350 V rms corrisponde a $U_{lim,mot} = \frac{350\sqrt{2}}{\sqrt{3}} = 285.77 \text{ V}$ picco di fase.

Poiché la tensione nominale del motore è la tensione per la quale sono dimensionati gli avvolgimenti e l'isolamento, tale valore non può essere superato. Ne consegue che il sistema di controllo deve effettuare la seguente verifica:

```
function U_lim = voltage_limit_calculator(Vdc, U_lim_mot,
flux_mg, p)

U_disp = 0.9*Vdc/sqrt(3); %tensione di fase resa disponibile
dall'inverter

if U_disp < U_lim_mot
    U_lim = U_disp;
else
    U_lim = U_lim_mot;
end

end
```

Il codice di cui sopra è implementato sotto forma di funzione Matlab all'interno di un subsystem che aggiusta la U_{lim} data in ingresso al regolatore di tensione.

Dato che il deflussaggio non è mai stato testato nella pratica, per verificare la funzionalità dell'algoritmo occorrono dei test dedicati, particolarmente delicati poiché la corrente I_d agisce lungo l'asse d cioè lungo l'asse del magnete. Una corrente I_d negativa, in modulo superiore ad $I_{d,max}$, può smagnetizzare permanentemente i magneti situati sul rotore, con conseguenze catastrofiche. Nei primi test ad alte velocità si consiglia pertanto di escludere il regolatore di tensione e di limitare il riferimento di velocità alla velocità a vuoto Ω_0 [rad.el/s], che è la massima velocità raggiungibile dal motore senza deflussarlo. Come già visto, se $\frac{V_{dc}}{\sqrt{3}} > U_{lim,mot}$

allora $\Omega_0 = \frac{U_{lim,mot}}{\Lambda_{mg}}$, altrimenti $\Omega_0 = \frac{\frac{V_{dc}}{\sqrt{3}}}{\Lambda_{mg}}$.

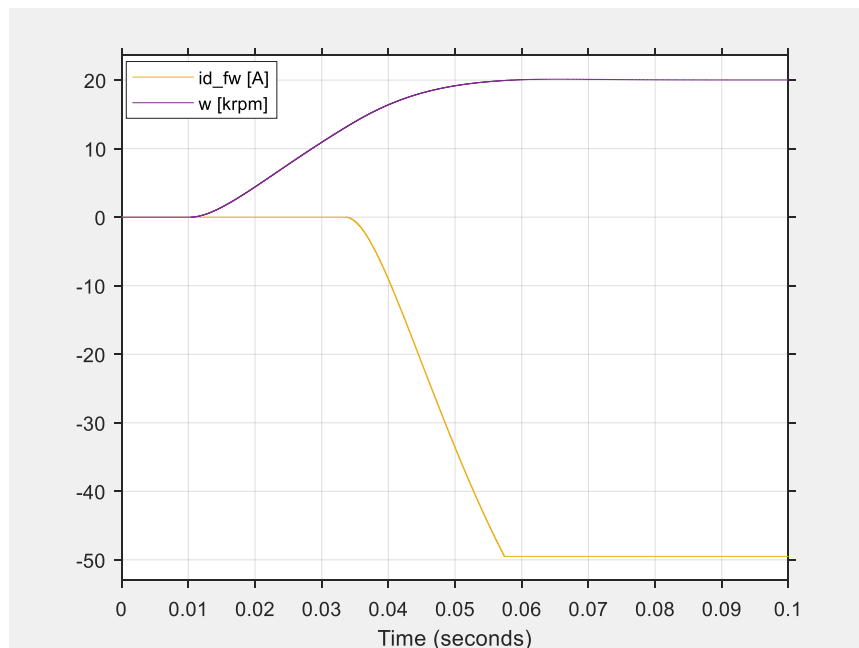


Fig. 5.16 – Riferimento di corrente di asse diretto (in giallo) al variare della velocità di rotazione del motore (linea viola), espressa in krpm. La tensione DC è impostata a 600 V e la corrente di asse d viene controllata soltanto per deflussare il motore

L'algoritmo di deflussaggio è stato testato, tramite simulazione, per i valori di tensione DC di normale funzionamento dell'accumulatore della vettura di Formula SAE, compresi tra 500 e 600 V.

Per tensioni DC inferiori si consiglia di eseguire simulazioni più approfondite e, se necessario, abbassare il riferimento di velocità, pari ad un valore inferiore, onde evitare che l'azionamento perda il controllo del motore.

5.2.4 Modellizzazione dell'inverter

Nella simulazione dell'algoritmo di controllo non è di interesse modellizzare in modo accurato il fenomeno della commutazione e, in generale, il comportamento degli interruttori di potenza. Pertanto, si assume che l'invertitore sia in grado di produrre ai morsetti del motore esattamente la tensione richiesta dal controllo, trascurando le non idealità quali la resistenza di conduzione degli interruttori e la presenza dei tempi morti.

Occorre tuttavia tenere presente che l'invertitore non riesce a produrre istantaneamente le tensioni necessarie a far scorrere le correnti desiderate. Sia T_s l'intervallo di tempo tra due chiamate successive dell'Interrupt Service Routine che contiene l'esecuzione dell'algoritmo di controllo. Il campionamento della corrente di fase avviene ad un certo istante kT_s , successivamente il microcontrollore calcola i nuovi duty-cycles e li applica al modulo PWM all'istante $(k + 1)T_s$, tuttavia l'effetto della PWM non è immediato, bensì lo si può considerare concentrato a metà del periodo di commutazione successivo, quindi a metà tra $(k + 1)T_s$ e $(k + 2)T_s$. Quanto detto è mostrato in figura.

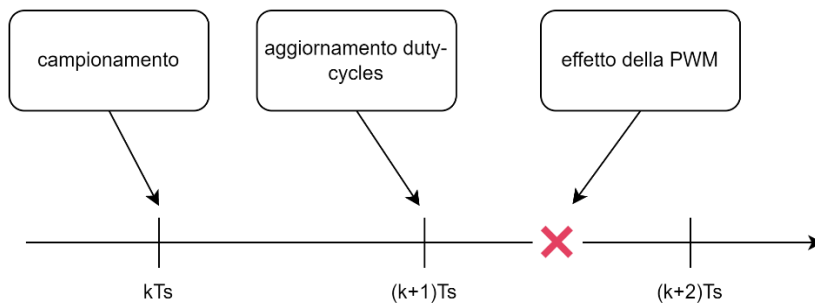


Fig. 5.17 - Rappresentazione grafica del ritardo dovuto all'invertitore

Si può quindi affermare che l'invertitore causa un ritardo pari a $1.5T_s$, in cui T_s solitamente coincide con il periodo di commutazione T_{sw} .

Nel dominio di Laplace, definita la variabile complessa s , un ritardo temporale pari ad un tempo τ_c lo si rappresenta con l'esponenziale $e^{-s\tau_c}$, il quale può essere approssimato nel seguente modo:

$$e^{-s\tau_c} \approx \frac{1}{1 + s\tau_c} \quad (5.15)$$

L'inverter si comporta quindi come un filtro passa-basso, con il polo alla frequenza angolare $\frac{1}{\tau_c} = \frac{1}{1.5T_s} \approx \frac{f_{sw}}{10}$ dove f_{sw} è la frequenza di commutazione degli interruttori di potenza. Per questo motivo, l'invertitore non riesce a produrre correttamente in uscita tensioni e correnti a frequenza superiore ad un decimo di quella di commutazione.

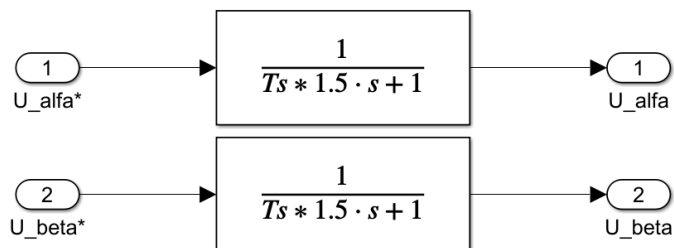


Fig. 5.18 - Nel modello Simulink l'invertitore viene rappresentato scomposto nei due assi α e β , ognuno dei quali è caratterizzato dalla funzione di trasferimento del primo ordine che rappresenta il ritardo di applicazione delle tensioni richieste dal controllo

Si è verificato tramite simulazione che il ritardo dell'invertitore, unitamente alla scelta di $f_{sw} = 20 \text{ kHz}$ quando la massima frequenza del motore è 1667 Hz, provoca problemi ai regolatori di corrente, i quali non riescono a seguire con precisione i riferimenti ed a garantire errore nullo a regime, come è lecito aspettarsi da dei controllori PI. La causa dei problemi (riferimenti di corrente seguiti in modo approssimativo e perdita di controllo alle alte velocità) è stata individuata essere nel ritardo dell'invertitore poiché annullando il ritardo o incrementando la frequenza di commutazione la situazione migliorava.

Per compensare il ritardo si è adottata la soluzione proposta in [2]. L'effetto del ritardo si traduce in un errore di modulo e fase del vettore spaziale di tensione

effettivamente applicato al motore, rispetto a quello richiesto dal controllo, pertanto la soluzione proposta consiste nel compensare l'angolo di ritardo, pari a $1.5T_s\omega_{me}$, con un aggiustamento del vettore del riferimento di tensione. Quest'ultimo, indicato come $\mathbf{u}^* = u_d^* + ju_q^*$, calcolato dall'algoritmo di controllo, viene pertanto corretto in ampiezza e fase nel sistema di riferimento sincrono, moltiplicandolo per il seguente termine:

$$\frac{2}{T_s\omega_{me}} \sin\left(\frac{T_s\omega_{me}}{2}\right) e^{j1.5T_s\omega_{me}} \quad (5.16)$$

Ricordando che

$$e^{j1.5T_s\omega_{me}} = \cos(1.5T_s\omega_{me}) + jsin(1.5T_s\omega_{me}) \quad (5.17)$$

si può calcolare il vettore spaziale di tensione corretto:

$$\mathbf{u}^{*'} = \mathbf{u}^* \frac{2}{T_s\omega_{me}} \sin\left(\frac{T_s\omega_{me}}{2}\right) e^{j1.5T_s\omega_{me}} = (u_d^* + ju_q^*) \frac{2}{T_s\omega_{me}} \sin\left(\frac{T_s\omega_{me}}{2}\right) [\cos(1.5T_s\omega_{me}) + jsin(1.5T_s\omega_{me})] = u_d^{*'} + ju_q^{*'} \quad (5.18)$$

Le tensioni richieste all'inverter, quindi, saranno $u_d^{*'}$ ed $u_q^{*'}$. La suddetta compensazione è stata implementata nella Matlab function "compensator", immediatamente a valle del disaccoppiamento degli anelli di corrente e compensazione della FCEM.

Si sottolinea come, anche in assenza di perdita del controllo del motore da parte del microcontrollore, operare controllando precisamente le correnti degli assi consente di seguire la traiettoria MTPA e, quindi, di operare con massima efficienza energetica, minimizzando il consumo di energia che viene fornita dall'accumulatore ad alta tensione.

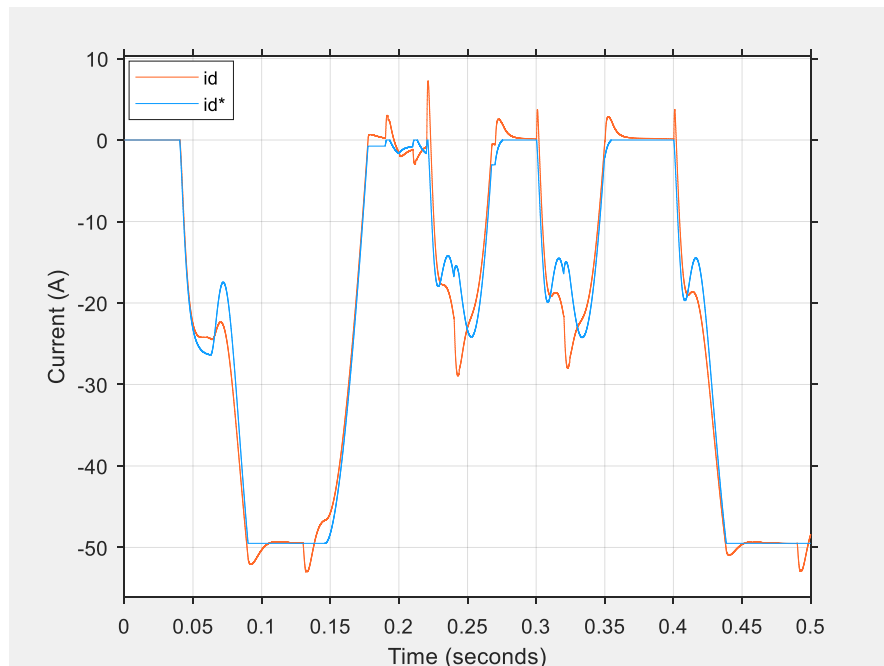


Fig. 5.19 - Corrente di asse diretto con compensazione del ritardo dell'invertitore

5.2.5 La taratura dei regolatori PI

Si riporta in figura lo schema a blocchi di un algoritmo di controllo basato su regolatori di corrente nel sistema di riferimento sincrono (R_{id} e R_{iq} rispettivamente per gli assi d e q) per motore brushless isotropo. La stessa trattazione vale anche per la macchina anisotropa, con la sola differenza che l'induttanza sincrona L deve essere sostituita con L_d ed L_q quando si considerano rispettivamente gli assi d e q. L'invertitore si modella come descritto nel paragrafo precedente ed è pertanto rappresentato scomposto nei due assi tramite le funzioni di trasferimento $G_c(s)$. Il motore, invece, viene modellizzato come descritto nel paragrafo "Equazioni del motore a magneti permanenti interni nel sistema sincrono dq".

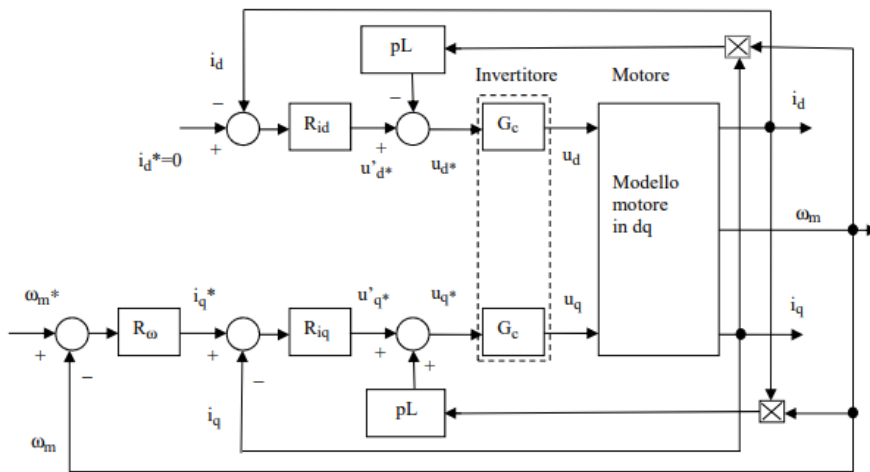


Fig. 5.20 - Schema a blocchi dell'algoritmo di controllo di un motore brushless con disaccoppiamento

Appare conveniente separare le dinamiche dei due assi, in modo da poter controllare separatamente le rispettive correnti. Si effettua perciò il cosiddetto "disaccoppiamento", sommando (per l'asse q) o sottraendo (per l'asse d) ai riferimenti di tensione generati dal controllo dei termini uguali in modulo e contrari in segno a quelli che vengono sommati o sottratti all'interno dello schema a blocchi, a causa delle dinamiche del motore. Se la costante di tempo τ_c dell'invertitore è piccola rispetto alle costanti di tempo elettriche $\frac{L_d}{R}$ e $\frac{L_q}{R}$, si ritiene che così facendo le dinamiche dei due assi siano perfettamente disaccoppiate.

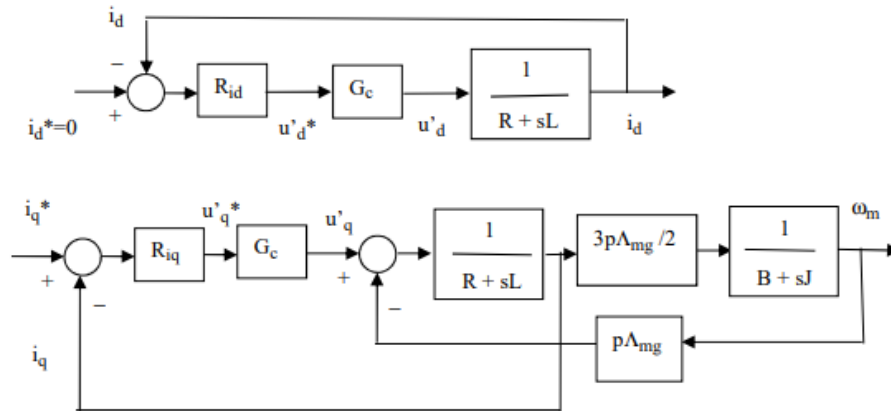


Fig. 5.21 - Schema a blocchi degli anelli di corrente di asse d e q (comprensivi del motore) a seguito del disaccoppiamento

Per migliorare la dinamica dell'azionamento, conviene inoltre compensare l'effetto della forza elettromotrice indotta (FCEM) pari a $\Omega_{me} \Lambda_{mg}$ che si manifesta nell'asse q . Si somma pertanto al riferimento u_q^* , in uscita dal regolatore, il termine $\Omega_{me} \Lambda_{mg}$.

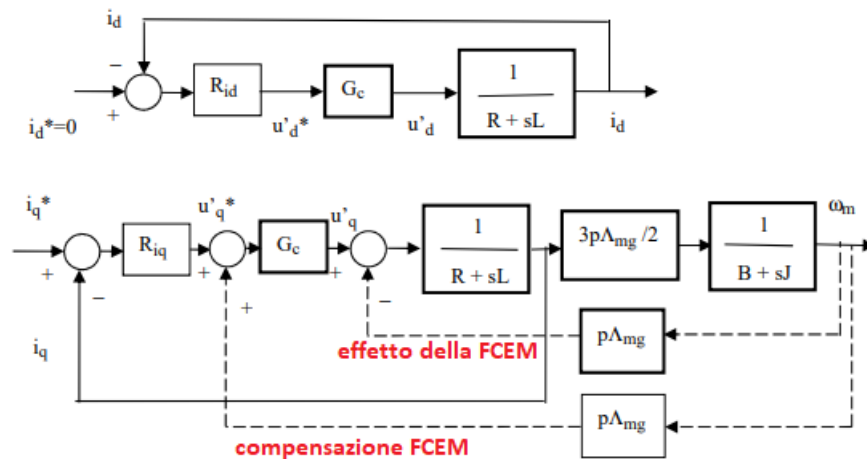


Fig. 5.22 - Schema a blocchi degli anelli di corrente di asse d e q , comprensivi di invertitore e motore, dopo disaccoppiamento e compensazione della forza contro-elettromotrice

La dinamica delle correnti risulta ora essere indipendente dai parametri del carico meccanico.

Per tarare i regolatori di corrente si è scelto di utilizzare la tecnica della cancellazione zero-polo: si usa lo zero del regolatore PI per cancellare il polo del motore, ovvero si pone la costante di tempo $\tau_R = \frac{K_P}{K_I}$ del PI pari a quella elettrica τ_e . Per l'asse q :

$$R_{iq}(s) = K_{Pq} \frac{1 + s\tau_{Riq}}{s\tau_{Riq}} = K_{Pq} \frac{1 + s\tau_e}{s\tau_e} \quad (5.19)$$

$$\text{con } \tau_e = \frac{L_q}{R} = \frac{240 \cdot 10^{-6}}{0.0675} = 3.56 \text{ ms.}$$

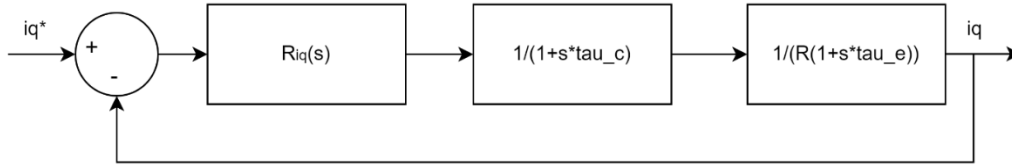


Fig. 5.23 - Schema del controllo della corrente nel dominio di Laplace, trascurando la dinamica meccanica

La funzione di trasferimento ad anello aperto, trascurando il carico meccanico, è:

$$GH(s) = K_{Pq} \frac{1 + s\tau_{Riq}}{s\tau_{Riq}} \frac{1}{1 + s\tau_c} \frac{1}{1 + s\tau_e} \frac{1}{R} = \frac{K_{Pq}}{R} \frac{1}{s\tau_e} \frac{1}{1 + s\tau_c} \quad (5.20)$$

Si impone ora il margine di fase m_φ . Esso deve essere positivo per la stabilità del controllo; di solito si scelgono valori nell'intervallo $45^\circ \div 90^\circ$. Scelti arbitrariamente 70° e posto $s = j\nu_{aiq}$, dove ν_{aiq} è la banda passante dell'anello di corrente:

$$m_\varphi = 70^\circ = 180^\circ + \arg[GH(j\nu_{aiq})] = 180^\circ - 90^\circ - \text{atan}(\nu_{aiq}\tau_c)$$

Riorganizzando e ipotizzando τ_c pari a $75 \mu\text{s}$ (questo verrà motivato nel paragrafo relativo alla scelta della frequenza del controllo), si ottiene una banda passante pari a:

$$\nu_{aiq} = \frac{\tan(20^\circ)}{\tau_c} = \frac{\tan(20^\circ)}{75 * 10^{-6}} = 4852.94 \text{ rad/s}$$

Noto il valore della banda passante, si può stimare il valore del guadagno proporzionale $K_{Pq} \approx \nu_{aiq}L_q = 1.1647$.

Per un calcolo più preciso di K_{Pq} si impone che il modulo di $GH(j\nu)$ valga 1 in corrispondenza della pulsazione di attraversamento:

$$|GH(j\nu_{aiq})| = \frac{K_{Pq}}{R\tau_e} \frac{1}{\nu_{aiq}\sqrt{1 + (\nu_{aiq}\tau_c)^2}} = 1 \quad (5.21)$$

che risolta in K_{Pq} porge $K_{Pq} = 1.2395 \text{ V/A}$. Ricordando che $\tau_R = \frac{K_P}{K_I}$ si ricava il valore del guadagno integrale $K_{Iq} = 348.6 \text{ V/A}$.

Ripetendo il procedimento per l'asse d, considerando però $\tau_e = \frac{L_d}{R} = \frac{120 * 10^{-6}}{0.0675} = 1.78 \text{ ms}$, si ricavano i guadagni $K_{Pd} = 0.61973 \text{ V/A}$ e $K_{Id} = 348.6 \text{ V/A}$.

Per quanto riguarda gli anelli di velocità e di tensione, l'approccio adottato è stato diverso. Per entrambi si sono scelti i valori dei guadagni tramite simulazioni in ambiente Simulink. Per l'anello di velocità, si è trovato che i valori ottimali sono $K_{P\omega} = 0.01 \frac{\text{Nm}}{\text{rad.el/s}}$ e $K_{I\omega} = 5$.

Per tarare il regolatore PI che gestisce il deflussaggio si sono effettuate simulazioni con diversi valori dei guadagni K_{Pu} e K_{Iu} , partendo dai valori indicati nel datasheet dei motori AMK: $K_{Pu} = 0.08 \text{ A/V}$ e $K_{Iu} = 13.33 \text{ A/V}$.

Per consentire al motore di accelerare uniformemente fino a raggiungere il riferimento di velocità, è necessaria una rapida azione di deflussaggio che può essere facilmente ottenuta introducendo un guadagno proporzionale diverso da zero. Si è notato tuttavia che con sola azione integrale ($K_{Pu} = 0$) l'azione deflussante è molto più regolare, come si può vedere dall'immagine sottostante.

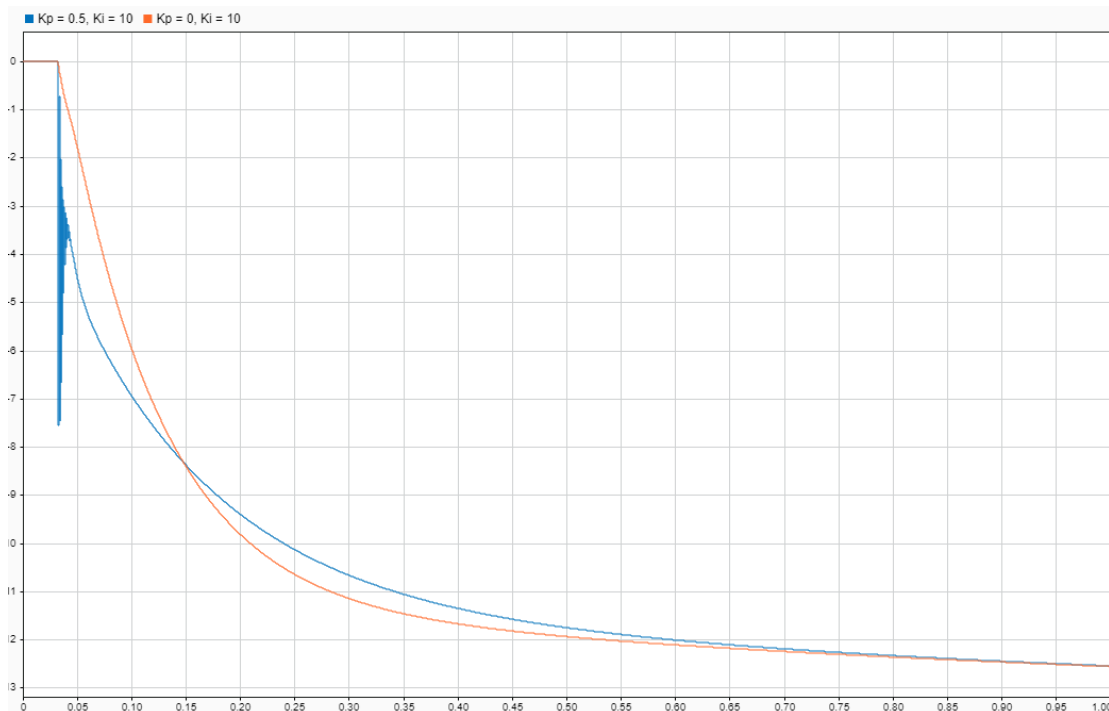


Fig. 5.24 - Riferimento di corrente I_d in uscita dal regolatore di tensione, in risposta ad un riferimento di velocità $\omega^* = 20000$ rpm, limiti di coppia ± 21 Nm. In arancione la risposta con sola azione integrale, in blu quella con azione sia integrale che proporzionale

Occorre poi tenere presente che la tensione DC varia da 600 V a 500 V ed il controllo deve essere in grado di portare il motore alla massima velocità raggiungibile, rispettando la massima corrente smagnetizzante che il motore può assorbire, ovvero $49.5 A_{pk}$.

Procedendo per tentativi si ricavano i valori $K_{Pu} = 0$ e $K_{Iu} = 1$, i quali garantiscono una efficace azione deflussante quando la tensione DC sta nell'intervallo $500 \div 600$ V e consentono di evitare pericolose sovraelongazioni della corrente smagnetizzante. Il regolatore di tensione è quindi un integratore puro.

Si può agevolmente verificare che è fisicamente possibile raggiungere 20000 rpm su tutto l'intervallo di tensione DC, infatti alla tensione minima del pacco batteria, la massima velocità raggiungibile in deflussaggio si calcola tramite la (4.20):

$$\Omega_{max} = \frac{\frac{500}{\sqrt{3}}}{A_{mg} - L_d I_{d,max}} = 12201 \text{ rad. el/s}$$

corrispondenti a 23300 rpm. Dai calcoli sopra eseguiti si ricava che la banda passante degli anelli di corrente è pari a 772 Hz, pertanto il controllo di corrente è molto reattivo. In fase di simulazione si è notato un comportamento "nervoso" quando i

limiti di coppia variano bruscamente, soprattutto nel passaggio da accelerazione a frenata (da coppia positiva a negativa) e viceversa. La prontezza degli anelli di corrente si è rivelata essere pericolosa soprattutto per la sovraelongazione nella risposta della corrente di asse d, la quale, se non controllata opportunamente, può smagnetizzare irreversibilmente i magneti permanenti. Per tale motivo si è scelto di implementare un filtro passa-basso del primo ordine che attenua le variazioni di coppia richieste dal regolatore di velocità agli anelli di corrente. Il filtro è caratterizzato dalla seguente funzione di trasferimento:

$$G(s) = \frac{1}{1 + s\tau} \quad (5.22)$$

La frequenza di taglio è stata scelta a circa un ventesimo della banda passante dei regolatori e posta pari a 40 Hz:

$$f = 40 \text{ Hz} \Rightarrow \tau = \frac{1}{2\pi f} = 3.98 \text{ ms}$$

Il filtro deve essere implementato in un sistema a tempo discreto. Sia x il valore da filtrare ed y il valore filtrato, allora y all'istante kT_s si può calcolare tramite i valori x ed y nel precedente istante di campionamento $(k - 1)T_s$:

$$y(kT_s) = A y((k - 1)T_s) + B x(kT_s) + C x((k - 1)T_s) \quad (5.23)$$

Tramite il Matlab Live Script "Discretizzazione.mlx" [3] è possibile ricavare i coefficienti A , B , C per implementare il filtro in ambiente Simulink e nel microcontrollore. Scelta la frequenza di taglio ed il tempo di campionamento, lo Script ricava la funzione di trasferimento a tempo discreto da quella a tempo continuo, tramite l'approssimazione di Tustin:

$$G(s) = \frac{1}{1 + s\tau} \Rightarrow G(z) = \frac{Bz + C}{z - A}$$

Con il filtro, la risposta del motore ad un gradino di velocità è leggermente più lenta, ma ciò è stato giudicato influente sulle prestazioni richieste all'azionamento.

5.3 Modello Simulink dello stadio di potenza

Il modello "Power_stage.slx" consente di simulare l'algoritmo di controllo, sviluppato in "Control.slx", assieme allo stadio di potenza, composto da accumulatore ad alta tensione, inverter ad IGBTs e motore a magneti permanenti interni. Gli anelli di controllo sono a tempo discreto, in quanto vengono simulati ad intervalli regolari pari al periodo del controllo T_s , mentre il modello fisico dell'azionamento è modellizzato a tempo continuo. La presenza dei modelli fisici di convertitore e motore aumenta l'onere computazionale della simulazione, la quale pertanto risulta essere notevolmente più lenta di quella relativa all'algoritmo di controllo.

Lo scopo di questo modello Simulink è validare l'algoritmo di Space Vector Modulation (nelle sue varianti simmetrica ed asimmetrica), tramite una

visualizzazione grafica delle correnti di fase e della coppia erogata dal motore. Può anche essere utilizzato per dimensionare la capacità di DC-link. Di questo si parlerà nei seguenti paragrafi.

5.3.1 Algoritmo di Space Vector Modulation

La Space Vector Modulation (SVM) è una procedura matematica che consente di calcolare i duty-cycles degli interruttori superiori di ogni fase, in un inverter a due livelli, a partire dai riferimenti di tensione in $\alpha\beta$ calcolati dalla routine di controllo. Essa consente quindi di tradurre il vettore spaziale di tensione in comandi fisici ai gates degli interruttori di potenza (passando attraverso i gate drivers).

Lo stato istantaneo di inverter a due livelli può essere descritto mediante un numero a tre cifre. Ogni cifra corrisponde ad una gamba e perciò ad una fase. Se tale cifra è 1 allora sta conducendo lo switch superiore, se è 0 allora sta conducendo quello inferiore. Essendo le fasi pari a 3, i possibili stati sono $2^3 = 8$. Ogni stato corrisponde ad un determinato vettore spaziale di tensione applicato ai morsetti di fase. Gli stati 000 e 111 corrispondono, nel piano $\alpha\beta$, al vettore di stato nullo (\mathbf{u}_0 o \mathbf{u}_7), poiché tutti gli interruttori, rispettivamente inferiori e superiori, conducono, mettendo di fatto in cortocircuito i terminali del carico. I restanti sei stati (100 ... 101) sono rappresentati da vettori (\mathbf{u}_1 \mathbf{u}_6) centrati nell'origine e che puntano ai vertici di un esagono, inscritto in una circonferenza di raggio $\frac{2}{3}V_{dc}$, dividendo quest'ultimo in sei settori (o sestanti).

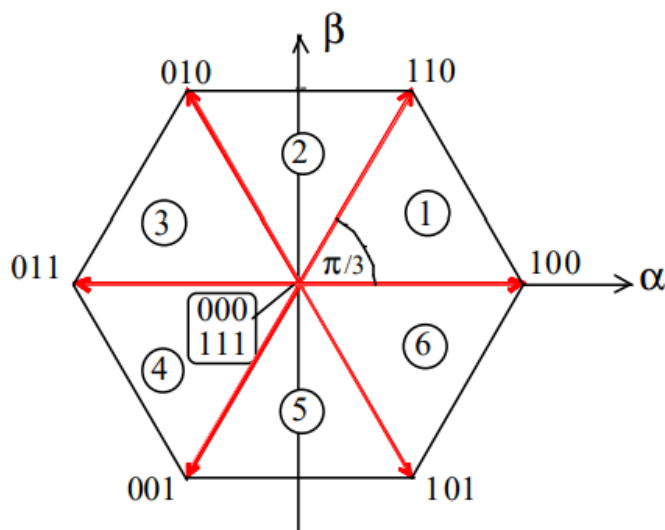


Fig. 5.25 – Esagono formato, nel piano $\alpha\beta$, unendo le punte dei sei vettori di stato non nulli dell'inverter a due livelli (indicati in rosso)

Sfruttando gli otto vettori di stato, è possibile sintetizzare un qualsiasi vettore spaziale di tensione. Si prenda come esempio la figura sottostante. Il vettore di riferimento è indicato con \mathbf{u}_s . Sfruttando la tecnica della PWM, si può suddividere il periodo di commutazione T_s in due subintervalli. Si immagini di applicare nel primo subintervallo, denominato t_1 , il vettore di stato \mathbf{u}_1 (100) e nel secondo, denominato t_2 , il vettore \mathbf{u}_2 (110). Se T_s è molto minore della costante di tempo del carico, ovvero

se la frequenza di commutazione è sufficientemente elevata, la tensione vista dal carico è quella media, pari alla combinazione lineare $\alpha_1 \mathbf{u}_1 + \alpha_2 \mathbf{u}_2$, dove $\alpha_1 = \frac{t_1}{T_s}$ e $\alpha_2 = \frac{t_2}{T_s}$. Deve risultare $\alpha_1 + \alpha_2 \leq 1$. L'ampiezza del vettore spaziale così sintetizzato può essere controllata mediante l'inserzione di un tempo t_0 , durante il quale si applicano i due stati 000 o 111. Deve risultare pertanto $t_0 + t_1 + t_2 = T_s$.

La Space Vector Modulation si presta bene ad essere implementata nei microcontrollori, poiché questi dispositivi eseguono agevolmente calcoli matematici e la tendenza è quella di una sempre maggiore potenza di calcolo, accompagnata ad un costo decrescente. La SVM, inoltre, consente di ottenere una tensione di fase massima sul carico pari a $\frac{V_{dc}}{\sqrt{3}}$, in valore di picco, mentre la tradizionale tecnica di PWM sinusoidale è limitata a $\frac{V_{dc}}{2}$, ovvero l'ampiezza massima con SVM è maggiore del 15.47%. Ciò si ottiene grazie all'iniezione, nella tensione di fase, di una terza armonica che poi scompare dalle tensioni concatenate.

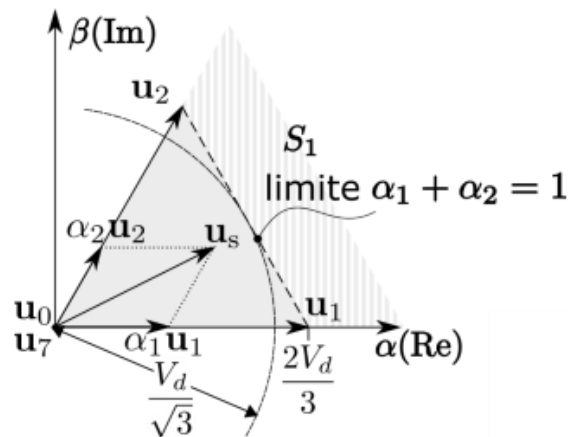


Fig. 5.26 – Sintesi di un generico vettore \mathbf{u}_s nel primo settore mediante la combinazione lineare dei vettori di stato adiacenti

L'algoritmo implementato in ambiente Simulink si divide in due parti.

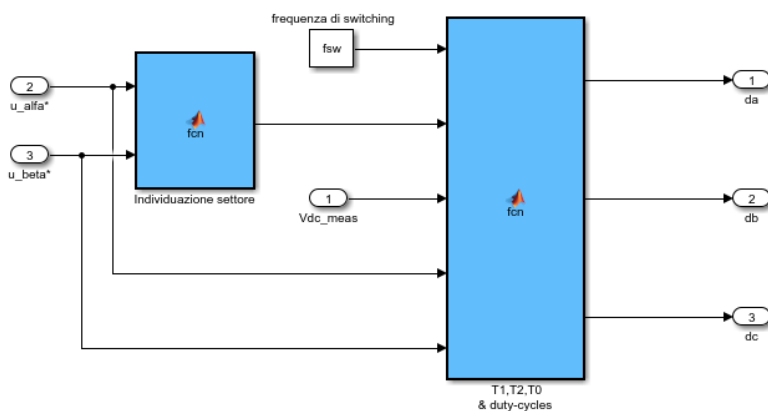


Fig. 5.27 – Functions che implementano la Space Vector Modulation

Occorre innanzitutto identificare in quale dei sei settori sia collocato il vettore spaziale $\mathbf{u}^* = u_\alpha^* + ju_\beta^*$. La routine di riconoscimento è implementata in una apposita funzione Matlab, la quale volutamente non fa uso della funzione arcotangente per ridurre il carico computazionale che grava sul microcontrollore. La funzione prende in ingresso le coordinate u_α^* ed u_β^* e restituisce il settore di appartenenza di \mathbf{u}^* .

```
Function set = fcn(u_alfa,u_beta)

if u_beta > 0                                %set = 1,2 o 3
    if abs(u_alfa) > abs(u_beta)/sqrt(3)     %set = 1 o 3
        if u_alfa > 0
            set=1;
        else
            set = 3;
        end
    else
        set = 2;
    end
else
    if abs(u_alfa) > abs(u_beta)/sqrt(3)     %set = 4, 5 o 6
        if u_alfa > 0                         %set = 4 o 6
            set = 6;
        else
            set = 4;
        end
    else
        set = 5;
    end
end
```

Il settore viene poi dato in ingresso, assieme a frequenza di commutazione, tensione misurata del DC-link e riferimenti u_α^* ed u_β^* , ad un'altra funzione che calcola i tempi di applicazione t_1 e t_2 dei vettori di stato adiacenti al settore di appartenenza del riferimento, proiettando \mathbf{u}^* su di essi. Viene anche calcolato l'eventuale tempo di applicazione del vettore di stato nullo t_0 .

```
Function [da,db,dc] = fcn(fsw,set,Vdc,u_alfa,u_beta)

% tempi di applicazione dei vettori spaziali precedente, successivo
e nullo
T1 = sqrt(3)*(u_alfa*sin(set*pi/3)-
u_beta*(cos(set*pi/3)))/(fsw*Vdc);
T2 = sqrt(3)*(u_beta*cos((set-1)*pi/3)-u_alfa*(sin((set-
1)*pi/3)))/(fsw*Vdc);

if ((T1+T2)>(1/fsw)) %in sovramodulazione mantengo la fase del
vettore spaziale
    T1 = T1/(fsw*(T1+T2));
    T2 = T2/(fsw*(T1+T2));
end

T0= (1/fsw)-T1-T2; %vettore di stato nullo

% calcolo duty-cycles degli switchwes high-side delle fasi a,b,c
```

```

da = 0; %duty-cycle fase a (switch S1)
db = 0; %duty-cycle fase b (switch S3)
dc = 0; %duty-cycle fase c (switch S5)

switch set
    case 1
        da = (T1+T2+T0/2)*fsw;
        db = (T2+T0/2)*fsw;
        dc = (T0/2)*fsw;
    case 2
        da = (T1+T0/2)*fsw;
        db = (T1+T2+T0/2)*fsw;
        dc = (T0/2)*fsw;
    case 3
        da = (T0/2)*fsw;
        db = (T1+T2+T0/2)*fsw;
        dc = (T2+T0/2)*fsw;
    case 4
        da = (T0/2)*fsw;
        db = (T1+T0/2)*fsw;
        dc = (T1+T2+T0/2)*fsw;
    case 5
        da = (T2+T0/2)*fsw;
        db = (T0/2)*fsw;
        dc = (T1+T2+T0/2)*fsw;
    case 6
        da = (T1+T2+T0/2)*fsw;
        db = (T0/2)*fsw;
        dc = (T1+T0/2)*fsw;
end
end

```

La sovramodulazione non è di interesse in questa tesi, dato che una funzione apposita limita la tensione richiesta all'inverter ad un valore U_{lim} sempre minore di $\frac{V_{dc}}{\sqrt{3}}$ (vedi paragrafo 5.2.3).

5.3.2 Scelta della tecnica di modulazione vettoriale

Esistono alcune varianti della Space Vector Modulation. Le principali sono le seguenti [6]:

- a singolo fronte
- simmetrica
- asimmetrica
- con modulazione a due gambe (o ad aggancio di bus, "bus-clamped modulation")

La SVM a singolo fronte è la più intuitiva: il periodo di commutazione comincia con gli interruttori superiori in conduzione (a patto che il loro duty-cycle sia maggiore di zero).

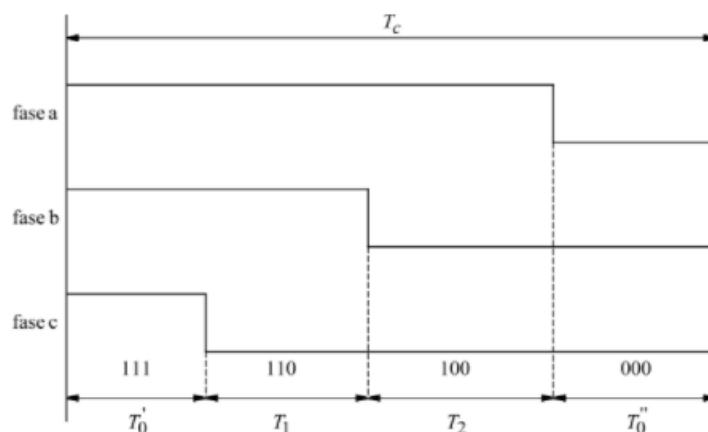


Fig. 5.28 - Modulazione a singolo fronte nel primo settore. In figura sono riportati i segnali PWM che controllano gli interruttori superiori, in un periodo di commutazione T_c

La modulazione simmetrica produce un contenuto armonico molto più favorevole. Essa consiste nel centrare il periodo di conduzione di ogni interruttore nella mezziera del periodo di commutazione, il quale comincia e termina sempre con un vettore di stato nullo. Ciò ovviamente non vale se il corrispondente duty-cycle è pari a 1.

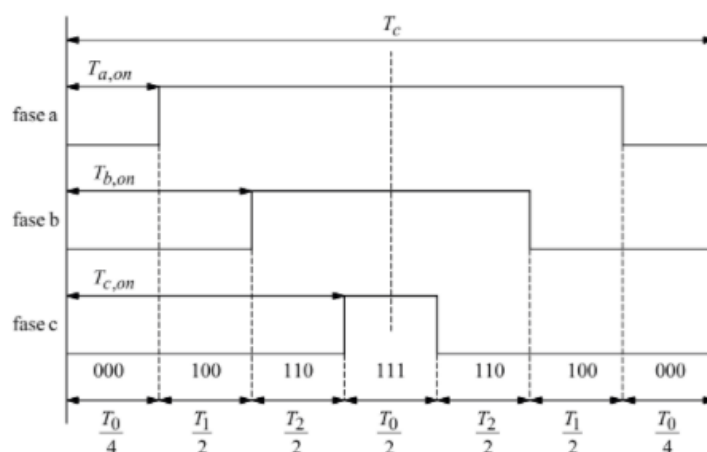


Fig. 5.29 - Modulazione simmetrica nel primo settore. In figura sono riportati i segnali PWM che controllano gli interruttori superiori, in un periodo di commutazione T_c

Al fine di ridurre le perdite di commutazione, direttamente proporzionali alla frequenza di commutazione f_{sw} , sono state sviluppate le versioni asimmetrica e quella ad aggancio di bus.

La prima consiste nell'aggiornare due volte i duty-cycles all'interno del periodo T_c . Questo richiede che l'algoritmo di controllo venga eseguito ad una frequenza doppia rispetto a quella della PWM.

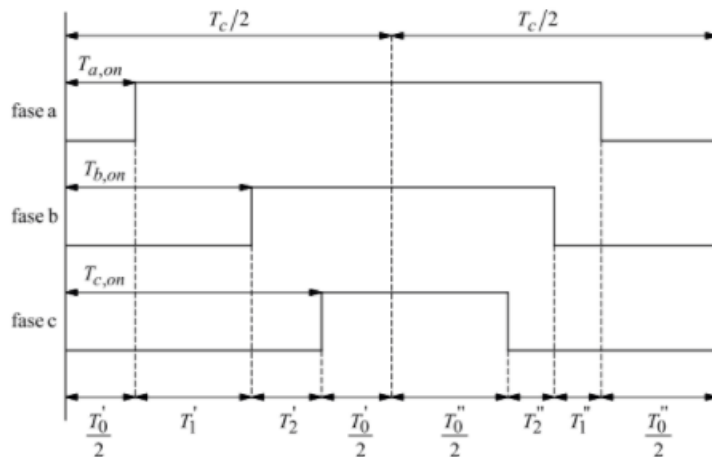


Fig. 5.30 - Modulazione asimmetrica nel primo settore. In figura sono riportati i segnali PWM che controllano gli interruttori superiori, in un periodo di commutazione T_c

La modulazione ad aggancio di bus, invece, consiste nel far commutare, in ogni periodo di commutazione, soltanto gli interruttori di due gambe, mantenendo sempre in conduzione o in interdizione quelli della terza gamba.

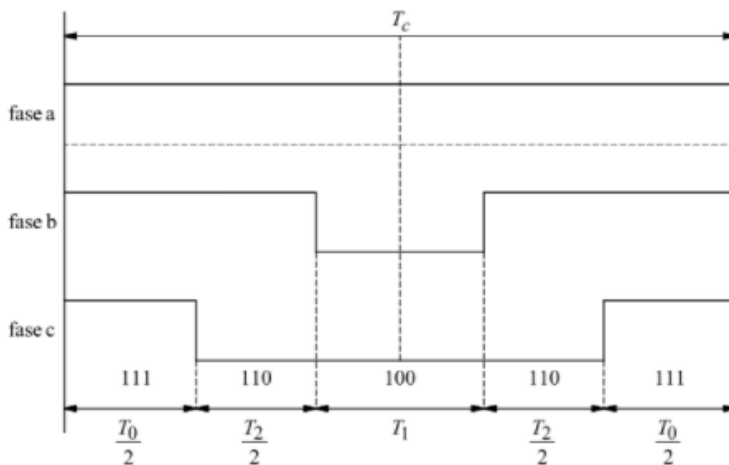


Fig. 5.31 - Modulazione ad aggancio di bus nel primo settore. Si noti come l'interruttore superiore della fase A resti in conduzione, senza commutare, "agganciando" la fase A al polo positivo del DC bus

Solitamente, nei convertitori che fanno uso di SVM simmetrica o a singolo fronte, la routine di controllo del motore viene eseguita in maniera sincrona con la PWM, quindi avviata all'inizio o a metà del periodo di commutazione. Si parla pertanto di "single-update" PWM poiché l'aggiornamento dei duty-cycles degli interruttori avviene una sola volta per ogni periodo.

La SVM asimmetrica, invece, è del tipo "double-update" PWM, in quanto prevede il calcolo e l'aggiornamento dei duty-cycles due volte in ogni periodo. Per questo essa consente di ottenere una frequenza del controllo pari al doppio di quella di commutazione. Si può utilizzare questa possibilità per migliorare ulteriormente il controllo del motore, fissata f_{sw} , oppure si può scegliere di mantenere la frequenza del controllo ma dimezzare la frequenza di commutazione, dimezzando quindi le

perdite di commutazione. Solo in questo ultimo caso, abbassando f_{sw} , le perdite sono più basse rispetto alla modulazione simmetrica.

A titolo di esempio, si consideri di usare la SVM simmetrica con $f_{sw} = 20$ kHz. L'algoritmo di controllo e la PWM sono sincroni, quindi la frequenza del controllo f_c è uguale a quella di commutazione f_{sw} .

Con la SVM asimmetrica, sempre con $f_{sw} = 20$ kHz, si riesce ad ottenere $f_c = 40$ kHz. È però possibile anche fissare $f_c = 20$ kHz e dimezzare f_{sw} , ponendola pari a 10 kHz. Ciò che si ottiene è un sistema con prestazioni leggermente peggiori del caso simmetrico, ma con perdite di commutazione dimezzate. In generale, più la frequenza f_c è elevata, più elevata può essere la frequenza massima del motore che si intende controllare.

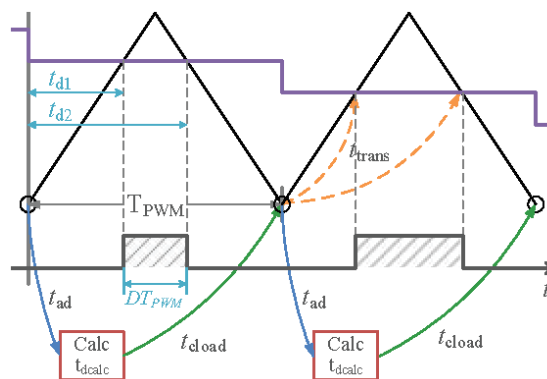


Fig. 5.32 - Schema di single-update PWM. L'onda triangolare (carrier) ha una frequenza pari a quella di commutazione, quindi un suo periodo coincide con quello di commutazione. La routine di controllo viene eseguita quando il carrier giunge al valore minimo (cerchiato in nero). I nuovi duty-cycles calcolati vengono caricati nei registri della PWM soltanto all'inizio del successivo periodo di commutazione, in corrispondenza del minimo successivo.

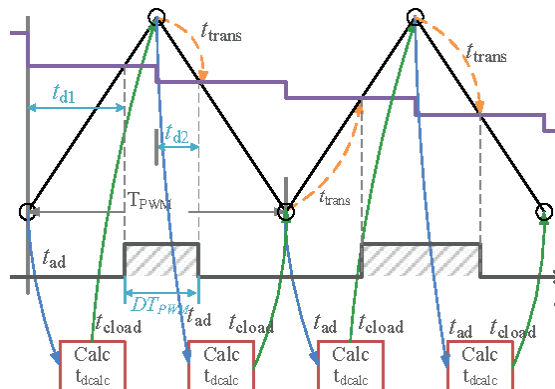


Fig. 5.33 - Schema di double-update PWM. Il campionamento delle correnti di fase, l'algoritmo di controllo e l'aggiornamento dei duty-cycles vengono effettuati due volte all'interno dello stesso periodo di commutazione.

La modulazione ad aggancio di bus, invece, prevede di far commutare soltanto due delle tre gambe in un periodo, pertanto la potenza dissipata in commutazione si riduce di $\frac{1}{3}$ rispetto al caso simmetrico o a singolo fronte, a parità di frequenza di commutazione. Questo si traduce in un peggior contenuto armonico delle correnti di fase e, conseguentemente, in un aumento delle perdite nel motore.

Anche con SVM asimmetrica peggiora il contenuto armonico, rispetto al caso di SVM simmetrica, tuttavia le simulazioni hanno mostrato che la asimmetrica è più adatta della bus-clamped, sia per la distorsione armonica che per il controllo del motore.

Originariamente, per il prototipo si era prevista una frequenza di commutazione di 20 kHz, usando la SVM simmetrica. Questa scelta è dettata dal fatto che i motori AMK hanno 10 poli ed una velocità massima di 20000 rpm. Ne consegue che la frequenza elettrica massima è $f_1 = \frac{20000 \cdot 5}{60} \approx 1667$ Hz. Per ottenere prestazioni di controllo adeguate, in letteratura si consiglia di adottare un rapporto di modulazione della frequenza $m_f = \frac{f_{sw}}{f_1}$ almeno pari a 10. Questo è giustificato dal fatto che, come spiegato nel paragrafo 5.2.4, se si rappresenta l'invertitore con una funzione di trasferimento del primo ordine, il polo si trova a circa un decimo della frequenza di commutazione. Scegliere 20 kHz sembra quindi la scelta più ovvia.

Sono state tuttavia compiute delle simulazioni termiche tramite il software Iposim, messo a disposizione da Infineon Technologies. Queste simulazioni hanno evidenziato come una frequenza di commutazione di 20 kHz porterebbe ad avere, nel funzionamento a pieno carico del motore (corrente di fase 105 Arms, 20000 rpm, 600 V lato continua), delle perdite negli IGBTs doppie rispetto agli inverters di AMK. Si parla di 2141 W/inverter a fronte dei 1080 W/inverter. I risultati delle simulazioni sono riportate in tabella.

Tabella 5.3

f_sw [Hz]	P. commutazione [W]	P. conduzione [W]	Perdite totali [W]
8000	707.04	405.36	1112.4
10000	853.56	383.04	1236.6
12000	1028.64	368.28	1396.92
14000	1162.44	380.22	1542.66
16000	1402.5	379.26	1781.76
18000	1555.74	370.98	1926.72
20000	1775.34	365.88	2141.22

Si nota che il contributo prevalente, nelle perdite totali, è costituito dalla potenza dissipata a causa delle commutazioni.

Nella simulazione probabilmente non si tiene conto delle perdite nei motori che sono causate dalla distorsione della corrente di fase, dovuta alla sintetizzazione di tensioni tramite PWM, evidentemente più elevate commutando a 8 kHz invece che a 20 kHz. La domanda che sorge spontanea è: come è possibile che gli inverters AMK controllino, commutando a 8 kHz, dei motori la cui fondamentale vale, alla velocità massima, ben 1667 Hz? Una ipotesi plausibile è che AMK abbia adottato proprio la SVM asimmetrica, ottenendo una frequenza della routine di controllo doppia rispetto alla frequenza di commutazione e pari a 16 kHz.

I vantaggi di questo approccio sono evidenti: a fronte della possibilità di controllare correttamente i motori anche ad elevate velocità, senza perderne il controllo, si riducono le perdite di commutazione poiché esse sono proporzionali alla frequenza di commutazione. Nella pratica, è possibile rispettare il limite indicato nel paragrafo 5.2.4 ponendo $f_c = 2f_{sw}$ e scegliendo $f_c > 10 f_1$.

Al fine di contenere le dissipazioni nel prototipo di inverter, le simulazioni si sono concentrate sulle tecniche a basse perdite di commutazione. La modulazione a due gambe è stata rapidamente scartata, in quanto la distorsione armonica delle correnti di fase era molto marcata e rendeva scarse le prestazioni dell'algoritmo di controllo. Le soluzioni proposte, quindi, restano due. Si può adottare la SVM simmetrica, mantenendo la sincronia controllo-PWM e si adotta una frequenza di commutazione nel range 10-20 kHz, accettando delle perdite maggiori che negli inverters AMK ma sapendo che la forma d'onda della corrente di fase sarà meno distorta. L'altra possibilità è di lavorare con SVM asimmetrica, scegliendo ad esempio $f_{sw} = 10$ kHz e $f_{sw} = 20$ kHz.

Dato che la riduzione maggiore delle perdite si ha tramite SVM asimmetrica, si è scelto di approfondire questa soluzione. Nel file Matlab "Initialization.m", che contiene i parametri necessari per il modello Simulink, si possono facilmente impostare f_{sw} e f_c semplicemente agendo sulle variabili f_{sw} ed f_{cont} presenti in "Initialization.m", per impostare rispettivamente le frequenze di commutazione e del controllo.

Da prove in ambiente Simulink sono state ricavate le seguenti immagini, ottenute imponendo al sistema di controllo un gradino di velocità da 0 a 20000 rpm, limite di coppia pari alla coppia massima di 21 Nm e 600 V lato continua.

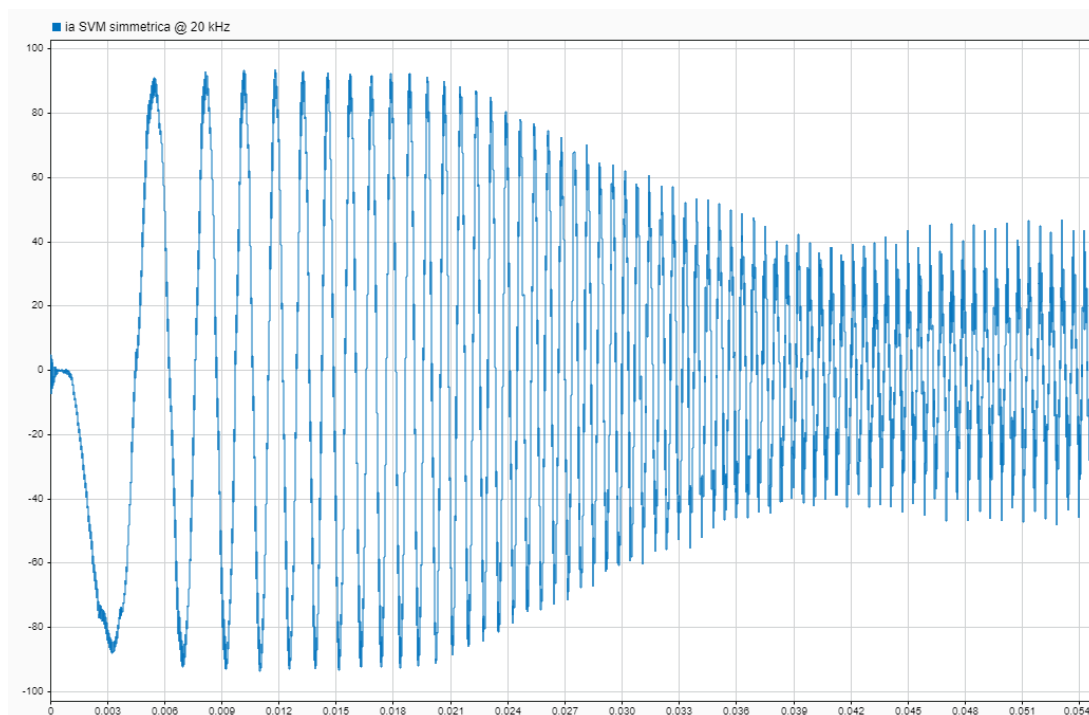


Fig. 5.34 - Corrente della fase A, SVM simmetrica, $f_{sw} = 20$ kHz

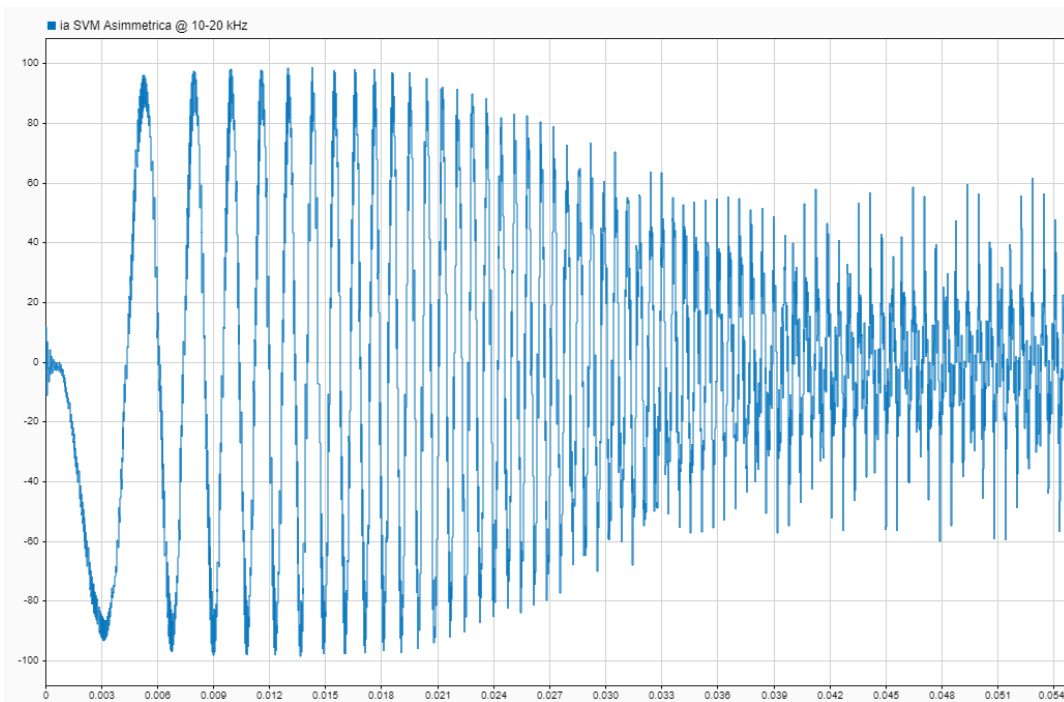


Fig. 5.35 - Corrente della fase A, SVM asimmetrica, $f_{sw} = 10 \text{ kHz}$, $f_c = 20 \text{ kHz}$

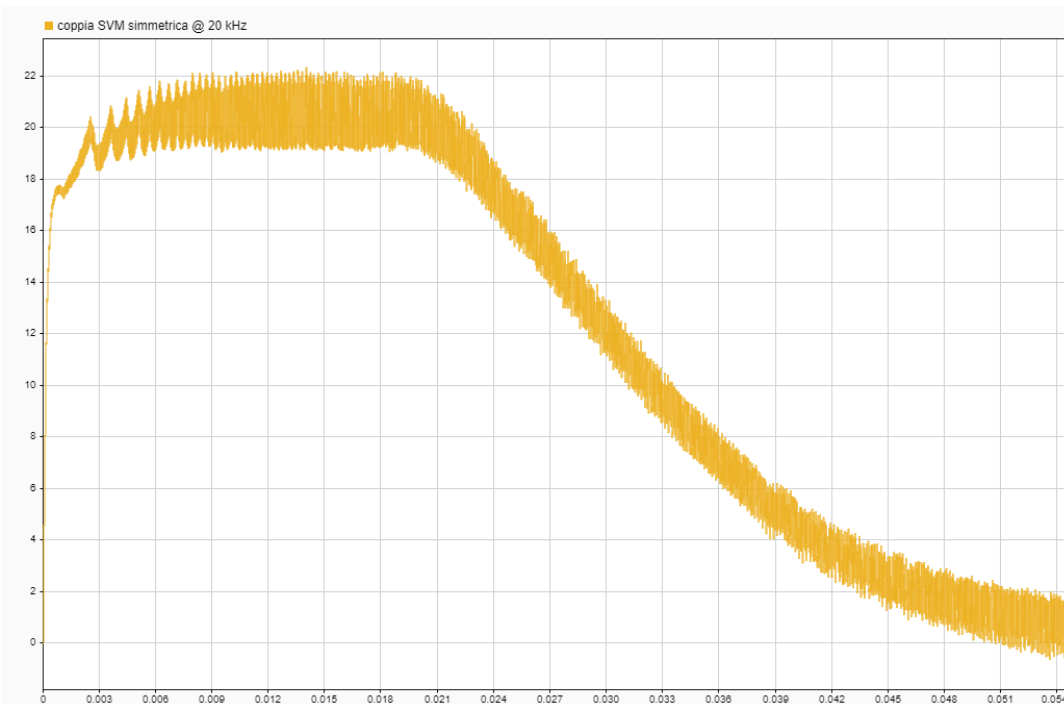


Fig. 5.36 - Coppia erogata, SVM simmetrica, $f_{sw} = 20 \text{ kHz}$

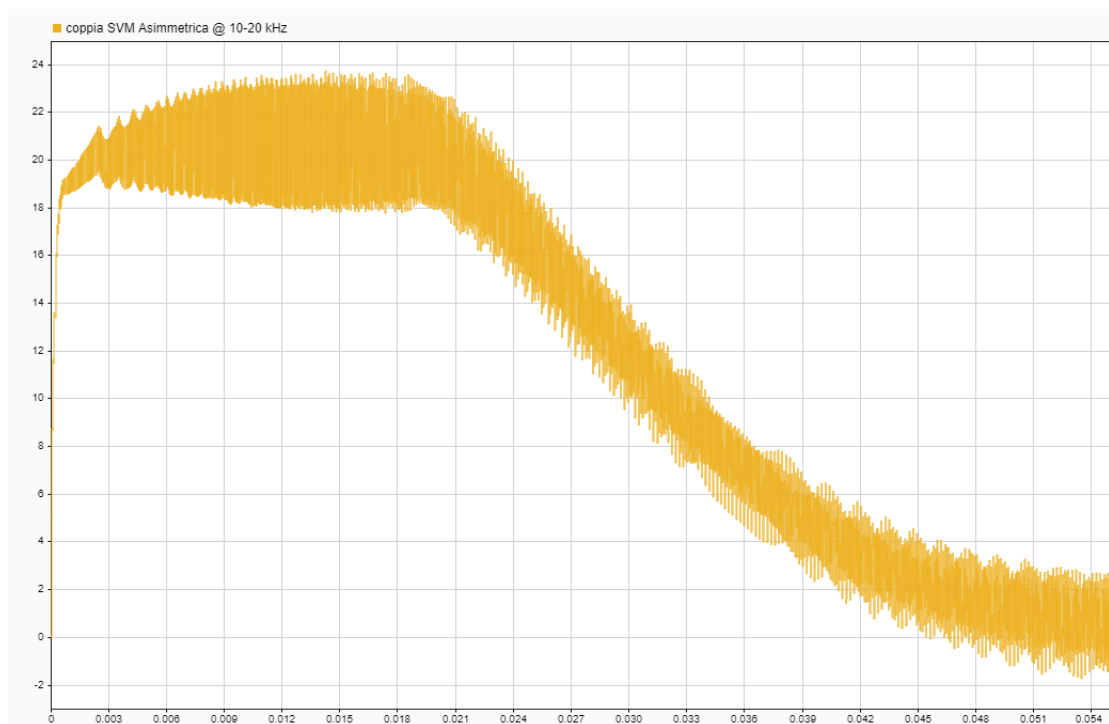


Fig. 5.37 - Coppia erogata. SVM asimmetrica, $f_{sw} = 10 \text{ kHz}$, $f_c = 20 \text{ kHz}$

Si può notare che, con Space Vector Modulation asimmetrica, il ripple della corrente di fase è maggiore, soprattutto ad elevata velocità, quando l'indice di modulazione della frequenza assume il valore minimo. Questo si ripercuote nella coppia erogata dal motore: le oscillazioni, nella fase di accelerazione a coppia massima, sono di 5 Nm circa, a fronte dei 3 Nm con modulazione simmetrica. I risultati delle simulazioni sono ritenuti promettenti. Di fatto, la SVM asimmetrica consente di sintetizzare ben due vettori spaziali di tensione all'interno dello stesso periodo di commutazione, per cui rappresenta un buon compromesso tra prestazioni dinamiche del sistema di controllo del motore e perdite negli interruttori dell'inverter.

Si opta quindi per un controllo eseguito a 20 kHz, mentre la frequenza di commutazione viene fissata a 10 kHz.

Dato che i criteri di calcolo dei duty-cycles sono gli stessi, sia con SVM simmetrica che asimmetrica, si può agevolmente passare da una tecnica di modulazione all'altra, impostando le frequenze di lavoro e senza bisogno di ulteriori modifiche.

5.3.3 Dimensionamento della capacità di DC-link

Per collegare un sistema in corrente continua, come un accumulatore elettrochimico, con uno in corrente alternata, quale un motore trifase, non è sufficiente interporre tra i due un convertitore a commutazione, poiché il funzionamento ottimale dei due sistemi si ottiene se tra di essi è presente un elemento in grado di accumulare energia, ovvero una induttanza oppure, nel caso di convertitori "a tensione impressa" come quello trattato in questa tesi, un condensatore di capacità appropriata.

Il condensatore svolge una duplice funzione: innanzitutto filtra le armoniche di corrente che si hanno lato continua, dovute alla commutazione degli interruttori nell'inverter, offrendo loro un percorso di richiusura a bassa impedenza. La reattanza capacitiva che un condensatore di capacità C offre all'armonica h -esima, di frequenza f_h , è pari a $\frac{1}{2\pi f_h C}$. Più aumenta l'ordine dell'armonica e, quindi, la sua frequenza, più la reattanza esibita a tale armonica sarà bassa. Se la reattanza dello stadio DC è maggiore di quella del condensatore, le correnti ad alta frequenza si richiudono attraverso il condensatore.

Il condensatore, inoltre, limita le oscillazioni della tensione DC dovute alla commutazione ed alle induttanze parassite. Più la capacità è elevata, tanto minore sarà l'oscillazione.

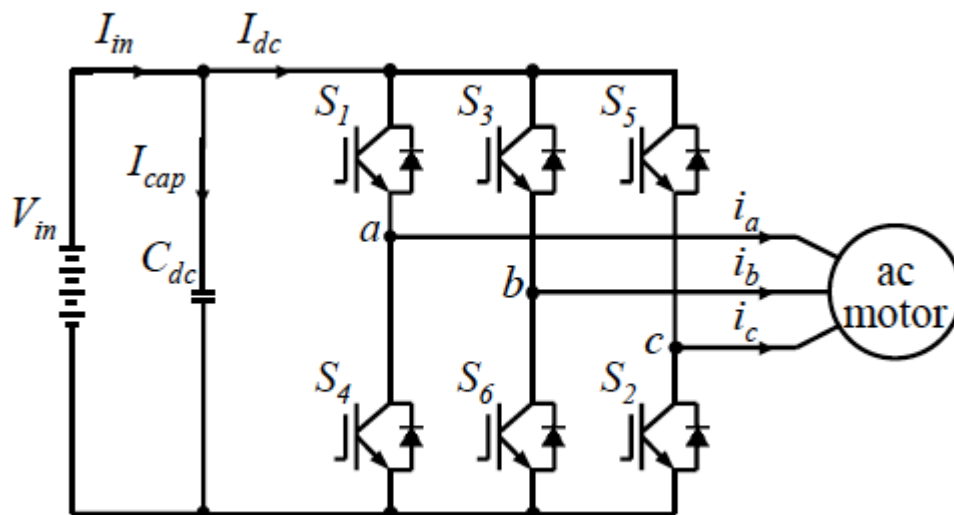


Fig. 5.38 - Schema semplificato di inverter a tensione impressa. C_{dc} è la capacità di DC-link

La capacità di un condensatore è $C = \frac{\epsilon S}{d}$, dove ϵ è la costante dielettrica assoluta del materiale presente tra le armature, aventi superficie S e distanti d . A parità di ϵ , per aumentare la capacità occorre avvicinare le armature o aumentare la superficie delle stesse, comportando quindi un aumento delle dimensioni (e della massa) del condensatore. Per questo motivo la capacità scelta dovrebbe avere il minimo valore necessario che garantisce il corretto funzionamento del sistema.

Un altro motivo per non eccedere nel valore di capacità sta nel fatto che il DC-link deve venire precaricato ad ogni accensione del Traction System e scaricato ad ogni spegnimento. Un banco di condensatori di capacità C non può essere collegato ad una sorgente di alimentazione in continua semplicemente chiudendo degli interruttori, poiché la corrente i_c assorbita da C (e quindi erogata dalla sorgente) sarebbe proporzionale alla rapidità con cui sale la tensione:

$$i_c = C \frac{dv}{dt} \quad (5.24)$$

Chiudendo istantaneamente i poli dell'accumulatore (Accumulator Isolation Relays, AIRs), la corrente erogata da quest'ultima sarebbe molto elevata, con effetti distruttivi per le celle che la costituiscono. Per evitare ciò, alla chiusura dei poli si collega in serie un resistore che limita i_C . Come da regolamento Formula SAE, raggiunto il 95% della tensione a vuoto dell'accumulatore sui condensatori, la resistenza di precarica viene esclusa dal circuito di alta tensione.

Quando si scollega il circuito di alta tensione, ad esempio quando la vettura viene spenta o in caso di eventi anomali che possono minare la sicurezza del pilota, è necessario dissipare l'energia che rimane accumulata nel DC-link. Ciò si ottiene collegando tra i due poli del DC-bus, ad accumulatore scollegato, un altro resistore detto di scarica.

Si comprende quindi come non sia desiderabile avere una capacità di DC-link più grande del necessario, poiché i tempi di precarica e scarica, governati dalla costante di tempo RC , sarebbero molto elevati.

D'altra parte, più la capacità lato DC è elevata, più energia è disponibile per alimentare il motore durante i transitori e minore è la componente armonica di corrente che attraversa il pacco batteria.

I criteri di scelta del condensatore di DC-link per l'invertitore sono:

- 1) il ripple di tensione lato continua deve essere ridotto ad un valore accettabile
- 2) le tensioni nominali e massima del condensatore devono essere tali da garantire la sicurezza di funzionamento anche in occasione di guasti, senza che il componente venga distrutto
- 3) il condensatore deve essere in grado di erogare o assorbire continuamente un certo valore efficace di corrente senza scaldarsi eccessivamente

Dato che gli invertitori della AMK, attualmente presenti in vettura, commutano ad una frequenza prossima a quella scelta per il prototipo, e dato che il carico (il motore) è lo stesso, si prendono come riferimento i dati della sua capacità di DC-link. All'interno di ogni invertitore, il DC-link è costituito dal parallelo di cinque condensatori WIMA, modello DC LINK MKP4 da 15 uF, che sopportano 10.5 Arms (a 10 kHz) e 900 V. La capacità totale quindi ammonta a 75 uF per ogni inverter.

Il ripple di tensione lato DC, in valore picco-picco, si può stimare come in [4]:

$$\Delta V_{pp} \approx \frac{I_0}{4f_{sw}C} \quad (5.25)$$

dove I_0 è l'ampiezza della corrente di fase. La suddetta stima vale nell'ipotesi di carico puramente reattivo e rappresenta pertanto la situazione relativa al caso peggiore. Si riportano in tabella i valori del ripple di tensione DC, rapportato alla tensione nominale del pacco batteria, al variare della frequenza di commutazione e della corrente (valori nominale e massimo).

Tabella 5.4

Capacità [μF]	f_{sw} [Hz]	Corrente [A picco]	Ripple [% V_{nom}]
75	8000	$41\sqrt{2}$	4.5
75	8000	$105\sqrt{2}$	11.6
75	10000	$41\sqrt{2}$	3.6
75	10000	$105\sqrt{2}$	9.3

Dai calcoli sopra eseguiti è difficile dire quando il ripple di tensione è accettabile o meno. Trattandosi di un prototipo, tuttavia, non c'è ancora la necessità di ottimizzare gli ingombri e le masse, per cui nel seguito si investigherà la possibilità di installare un valore di capacità maggiore a quella attualmente presente.

Il condensatore di DC-link si può modellare come un circuito RLC serie. R ed L rappresentano rispettivamente la resistenza (Equivalent Series Resistance, ESR) e l'induttanza (Equivalent Series Inductance, ESL) parassite. L' ESR è particolarmente importante poiché la potenza dissipata su di essa contribuisce a scaldare il condensatore e vale:

$$P = ESR \cdot I_{rms}^2 \quad (5.26)$$

dove I_{rms} è il valore efficace di corrente del ripple, assorbito dall'inverter. La ESL , invece, va valutata onde evitare che alle frequenze di lavoro si verifichi una risonanza serie tra di essa e la capacità.

Il motore controllato dal prototipo è sempre quello di AMK, quindi è ragionevole supporre che il valore efficace di corrente che scorrerà nel condensatore da dimensionare sarà simile a quello dei condensatori WIMA. Per ricavare la corrente efficace che scorre nel condensatore sono state eseguite simulazioni tramite il modello "Power_stage.slx".

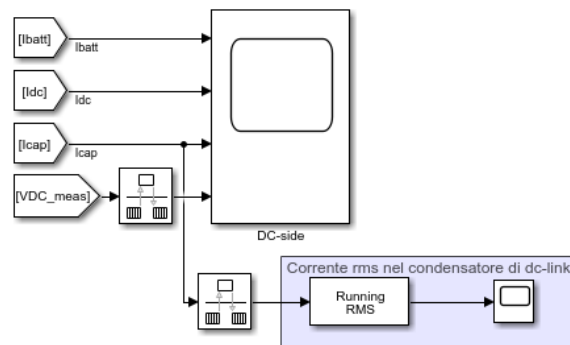


Fig. 5.39 - Calcolo della corrente efficace nel condensatore

Da simulazione, risulta che il valore massimo di corrente efficace nel condensatore è 34.5 Arms, compatibile con il rating dei condensatori scelti da AMK.

La tensione nominale, infine, dovrebbe essere almeno una volta e mezza quella massima di lavoro del circuito di alta tensione (600 V), dato che la commutazione degli interruttori è di tipo hard.

I requisiti che devono essere soddisfatti dal condensatore sono quindi riportati in tabella.

Tabella 5.5

Capacità	$\geq 75 \mu\text{F}$
Tensione nominale	$\geq 900 \text{ V}$
Corrente	$\geq 40 \text{ Arms}$

Sarebbe auspicabile, inoltre, avere dei bassi valori di ESR ed ESL . Occorre infine verificare che la frequenza di risonanza tra C ed ESL sia molto più elevata della frequenza di commutazione.

Il modello scelto è il 944U101K102ABM della Cornell Dubilier, avente tensione nominale 1 kV, capacità 100 μF e ripple di corrente ammissibile pari a 68 Arms, a 55°C. Il condensatore è in grado di reggere una tensione pari al 150% della nominale per un massimo di 10 secondi; questo dovrebbe consentire di sopportare eventuali sovratensioni transitorie. I valori del ripple di tensione percentuale sono riassunti nel seguito.

Tabella 5.6

Capacità [μF]	f_{sw} [Hz]	Corrente [A picco]	Ripple [% V_{nom}]
100	10000	$41\sqrt{2}$	2.7
100	10000	$105\sqrt{2}$	7.0

La frequenza di risonanza del condensatore si calcola come:

$$f_0 = \frac{1}{2\pi\sqrt{LC}} = \frac{1}{2\pi\sqrt{ESL * C}} \quad (5.27)$$

e risulta essere pari a 91.9 kHz, ed è sufficientemente maggiore di quella di commutazione f_{sw} , consentendo di evitare pericolose risonanze.

6 Implementazione degli algoritmi nel firmware di un microcontrollore

Il codice per il microcontrollore è stato sviluppato nell'ambiente Code Composer Studio, a partire dai modelli Simulink dell'algoritmo di controllo. La traduzione in codice è stata eseguita a mano, nonostante la Mathworks renda disponibile un generatore automatico di codice. Il motivo di questa scelta risiede nel fatto che, mentre un programma ottenuto tramite generatore automatico è di difficile interpretazione, la scrittura a mano consente una organizzazione più razionale del codice e rende più interattive le fasi di debugging e collaudo dello stesso.

Nella figura sottostante è rappresentata la struttura dell'algoritmo di controllo, eseguito dal microcontrollore, che si colloca all'interno del riquadro "controllo motore" nel diagramma riportato nel paragrafo 3.3.

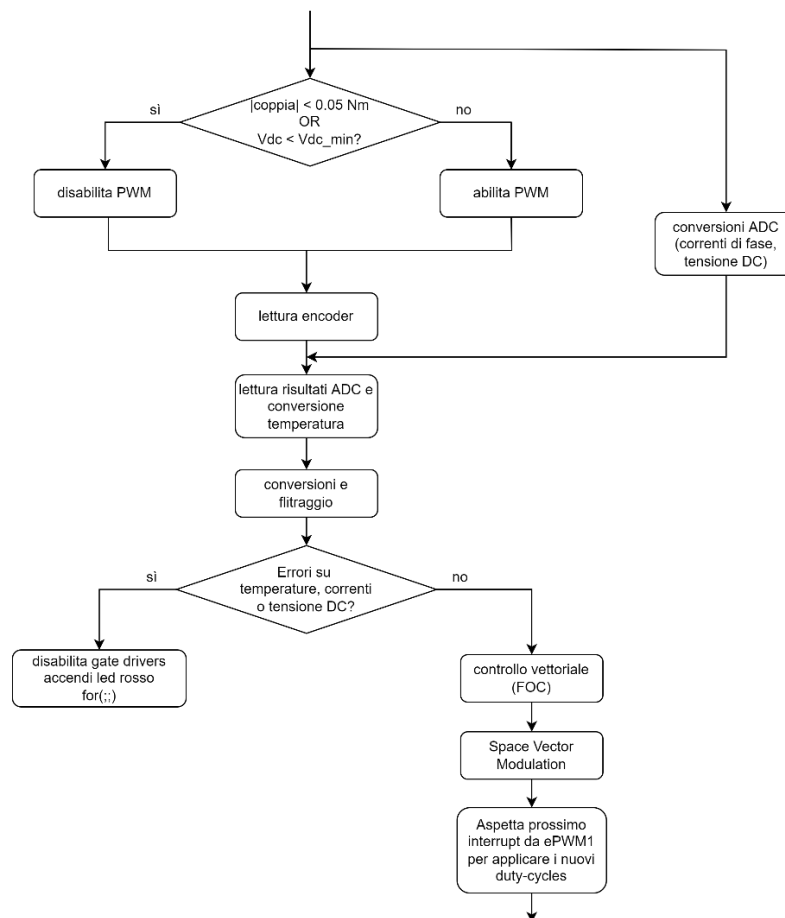


Fig. 6.1 - Struttura del firmware. Vista in dettaglio del contenuto del blocchetto "controllo motore" presentato nello schema del paragrafo 3.3

L'algoritmo viene eseguito ogniqualvolta il contatore del modulo ePWM1 genera un interrupt, ovvero all'inizio di ogni periodo di commutazione, in caso di SVM simmetrica, o sia all'inizio che a metà dello stesso, in caso di modulazione asimmetrica. Questo interrupt avvia sia le conversioni ADC di correnti e tensione, sia la Interrupt Service Routine (ISR) che regola il controllo del motore. Per prime vengono eseguite le tre conversioni ADC (questo è stato verificato sperimentalmente), che vengono avviate nello stesso istante ma non sono contemporanee, in quanto il convertitore analogico - digitale è uno solo. Successivamente il microcontrollore esegue le istruzioni contenute nella ISR.

Le uscite PWM della LAUNCHXL-F28379D vengono abilitate soltanto se la richiesta di coppia all'invertitore è maggiore, in modulo, di 0.05 Nm e se la tensione del DC-bus è superiore alla soglia minima. Con questa verifica si evita che l'invertitore inietti negli avvolgimenti di statore correnti troppo piccole per essere controllate adeguatamente (tenuto conto della lettura imperfetta dei sensori di corrente) ma anche che il motore ruoti finché i condensatori lato continua non sono stati precaricati. Se le uscite PWM non sono abilitate, esse assumono tutto lo stato logico 0, con l'effetto di mantenere il motore "in folle" poiché tutti i transistor sono interdetti. Eseguita questa verifica, indipendentemente dal suo esito, in successione viene letto l'encoder, vengono calcolate la posizione elettromeccanica e la velocità, vengono letti i risultati delle conversioni ADC e calcolati i valori assoluti delle grandezze. Infine, tutti i valori vengono opportunamente filtrati.

Arrivato a questo punto, il microcontrollore controlla che tutte le grandezze misurate siano all'interno dell'intervallo consentito. Se così non fosse, verrebbero disabilitate le uscite PWM ed il segnale di Enable ai gate drivers, verrebbe acceso il LED rosso della LAUNCHXL-F28379D ed il microcontrollore entrerebbe in un ciclo for infinito.

Una volta verificata l'assenza di errori, si esegue l'algoritmo di controllo motore vero e proprio, costituito dagli anelli di velocità, corrente e tensione. Infine, tramite la procedura di Space Vector Modulation, si calcolano i nuovi valori dei duty-cycles per produrre sui morsetti del motore le tensioni richieste. Questi valori verranno però scritti nei registri dei moduli PWM soltanto al verificarsi del successivo interrupt.

Nel seguito verranno descritte nel dettaglio le periferiche utilizzate dal microcontrollore, ad esclusione di quella SPI, già presentata nei precedenti capitoli.

6.1 La periferica ePWM

La scheda LAUNCHXL-F28379D è dotata di otto moduli ePWM ("enhanced PWM") che consentono di generare segnali ad onda quadra, di ampiezza 3.3 V, a duty-cycle variabile. Tramite questi segnali è possibile controllare gli interruttori dell'invertitore applicando una tecnica chiamata "modulazione a larghezza d'impulso" (Pulse Width Modulation, PWM). Essa consente di alimentare un carico con una tensione variabile, pur disponendo solo di un numero finito di livelli di tensione continua.

L'esempio più semplice di regolazione della tensione mediante l'applicazione della PWM è rappresentato nella figura sottostante. Disponendo di due soli livelli di tensione (0 e V_d), controllando i tempi di apertura e chiusura dell'interruttore allo

stato solido T , è possibile applicare al carico R una tensione media V_o che può assumere un valore compreso tra 0 e V_d :

$$V_o = \frac{1}{T_s} \int_0^{T_s} v_o(t) dt = V_o \frac{t_{ON}}{T_s} = V_o D \quad (6.1)$$

dove T_s è il periodo di commutazione e t_{ON} la durata della conduzione. Il duty-cycle D dell'interruttore T è definito come il rapporto tra il tempo durante il quale l'interruttore è chiuso (cioè conduce) ed il periodo di commutazione.

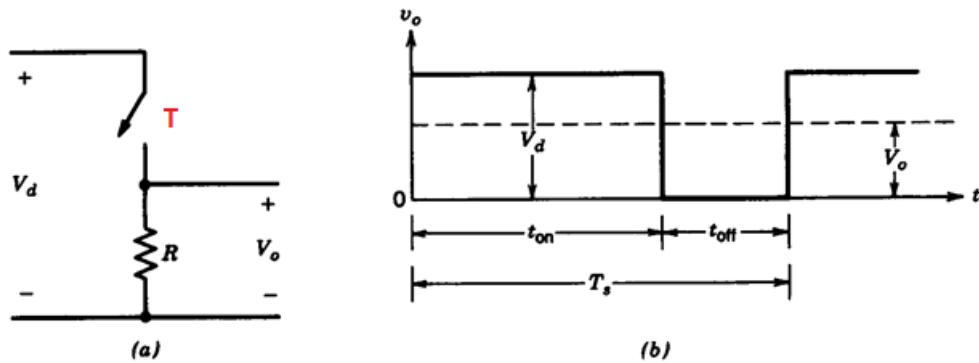


Fig. 6.2 – Regolazione della tensione media sul carico mediante PWM

Un inverter a tensione impressa, a due livelli, funziona secondo lo stesso principio, tuttavia gli interruttori da controllare sono sei.

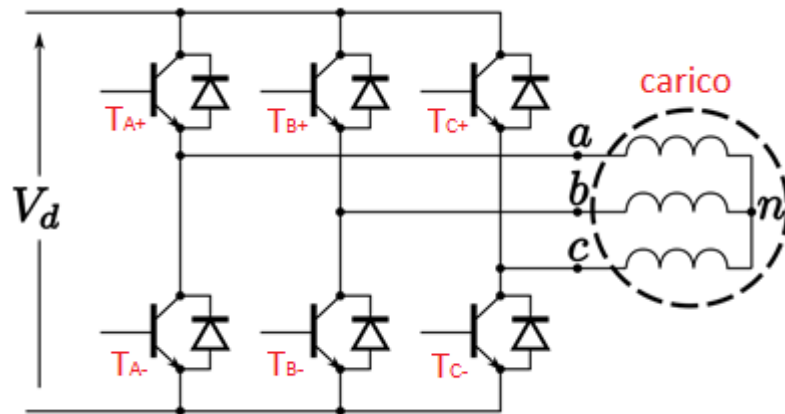


Fig. 6.3 – Schema di principio di inverter a due livelli

Sono quindi necessari sei diversi segnali PWM, tuttavia su una stessa gamba i due transistor (ad esempio T_{A+} e T_{A-} per la fase A) sono pilotati in maniera alternativa: quello inferiore conduce quando quello superiore è in interdizione e viceversa. L'algoritmo di controllo calcola, per ogni gamba dell'invertitore (e quindi per ogni fase) il duty-cycle D_+ dell'interruttore superiore, mentre la periferica ePWM ricava poi automaticamente il duty-cycle D_- di quello inferiore come $D_- = 1 - D_+$.

Siccome gli interruttori allo stato solido reali non passano istantaneamente dallo stato di interdizione a quello di conduzione e viceversa, occorre inserire, tra lo

spegnimento di un interruttore e l'accensione dell'altro dispositivo sulla stessa gamba, il cosiddetto "tempo morto". Si tratta di un intervallo durante il quale entrambi gli interruttori di una gamba non conducono, onde evitare cortocircuiti distruttivi ("shoot-through").

Ogni periferica ePWM è dotata di due canali, chiamati A e B. Essi possono essere gestiti indipendentemente o congiuntamente. Ogni periferica riceve in ingresso un valore numerico e produce in uscita, sul canale A, un segnale in onda quadra avente duty-cycle pari a D . Sull'altro canale (B), la periferica è in grado di produrre, se l'utente lo desidera, un segnale ad onda quadra, avente duty-cycle pari a $1 - D$ e complementare al canale A. La periferica si occupa anche dell'inserzione del tempo morto. I canali A e B sono collegati direttamente alla gate driver board, dove i gate drivers si occupano di trasformare i segnali logici in segnali di comando per i gates degli IGBTs.

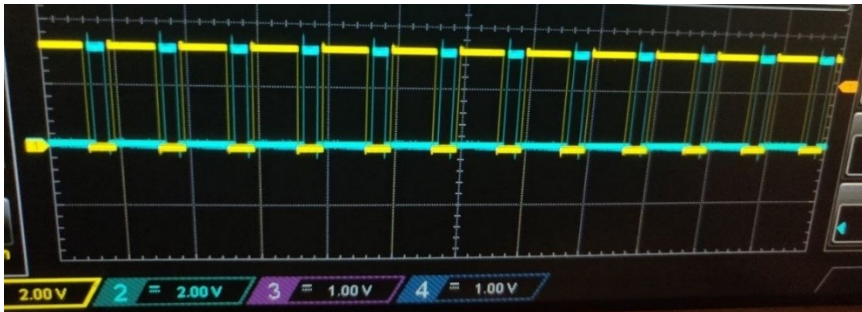


Fig. 6.4 – Rilievo tramite oscilloscopio di forme d'onda di segnale PWM, canale A (giallo) e B (blu). Si noti la presenza del tempo morto tra il fronte di discesa di un canale ed il successivo fronte di salita dell'altro canale

Le periferiche (o moduli) ePWM vengono configurate via software. Il codice relativo si trova in `epwm_management.c` e nell'omonimo header. Nel seguito verrà spiegato sommariamente il funzionamento dei moduli.

Per produrre un'onda quadra, la periferica si serve di un contatore interno, il cui valore istantaneo è leggibile nel registro TBCTR. Ad ogni periodo del clock TBCLK (riscaldato da EPWMCLK, il clock della periferica ePWM), il contatore incrementa di uno, conseguentemente anche TBCTR viene aumentato di uno. Si è scelto di usare la modalità "Up-Down Count Mode": il contatore parte da zero, incrementa fino al valore TBPRD, dopodiché decrementa fino a raggiungere il valore zero ed il ciclo riparte da capo. Con tale modalità è possibile realizzare la Space Vector Modulation centrata attorno alla mezzieria del periodo (sia simmetrica che asimmetrica).

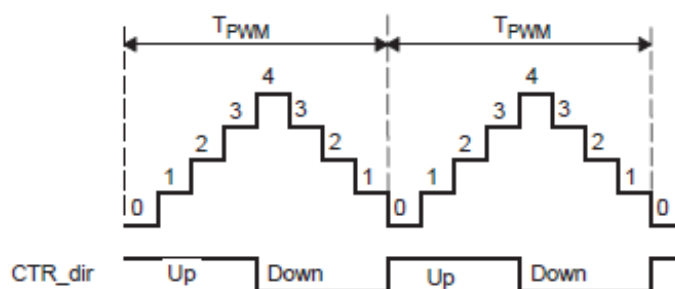


Fig. 6.5 – Schema illustrativo dell'Up-Down Count Mode. T_{PWM} è il periodo di commutazione, in cicli di TBCLK, mentre `CTR_dir` indica la direzione del contatore (ascendente Up o discendente Down)

Il duty-cycle viene impostato scrivendo nei registri CMPA e CMPB, relativi rispettivamente ai canali A e B. Quando il contatore TBCTR, incrementato o decrementando, assume un valore pari a CMPA o CMPB, l'uscita che produce il segnale PWM cambia stato logico, passando da 0 ad 1 o viceversa.

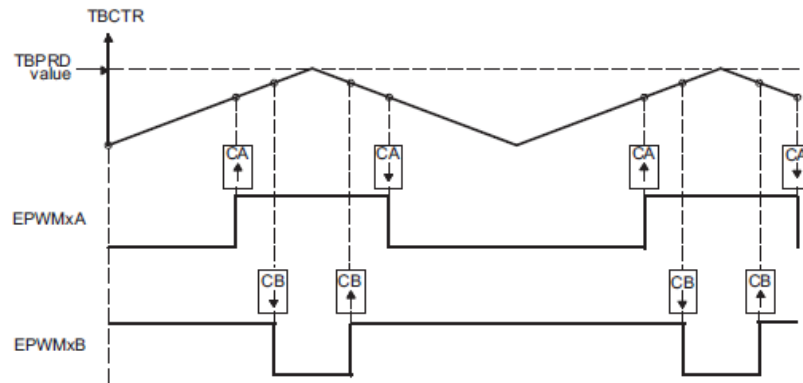


Fig. 6.6 – Il segnale EPWMx, canale A (EPWMxA), è a 1 finché TBCTR >= CMPA (indicato come CA), mentre il canale B (EPWMxB), viene controllato indipendentemente dall'A ed è a 1 quando TBCTR <= CMPB (CB)

Volendo controllare i due interruttori della stessa gamba in maniera alternativa, i canali PWM A e B di ogni modulo sono stati impostati in modo da lavorare alternativamente. La commutazione avviene al passaggio di TBCTR per CMPA, per il canale A, e per CMPB, per il canale B. Ponendo sempre $CMPA = CMPB$ si ottiene la commutazione sincrona dei due canali, ovvero ci si assicura che quando un canale passa allo stato logico "alto", l'altro passa a quello "basso".

L'algoritmo di controllo calcola i duty-cycles e poi li converte in opportuni valori di CMPA e CMPB, mediante la seguente formula:

$$CMPA = CMPB = TBPRD(1 - D) \quad (6.2)$$

dove D è compreso tra 0 e 1. Si nota come il tempo di ON del canale A (quando il segnale è a 1), è centrato attorno alla mezzeria del periodo (proprio come il segnale EPWMxA nell'immagine soprastante).

Il tempo morto viene gestito dal microcontrollore, in particolare dal Dead-Band (DB) Generator Submodule. Questo consente di ritardare il fronte di salita, inserendo un Rising Edge Delay (RED), oppure il fronte di discesa, inserendo un Falling Edge Delay (FED).

Siccome il controllo calcola il duty-cycle D di un solo segnale (quello del canale A), quello del canale B si ricava per complementarità. Tale modo di funzionamento è chiamato Active High Complementary (AHC) dalla Texas Instruments. Onde evitare cortocircuiti sulla stessa gamba, occorre ritardare i fronti di saliti di entrambi i segnali (canali A e B). Tutto ciò viene impostato tramite dalla funzione **setupEPWMActiveHighComplementary**.

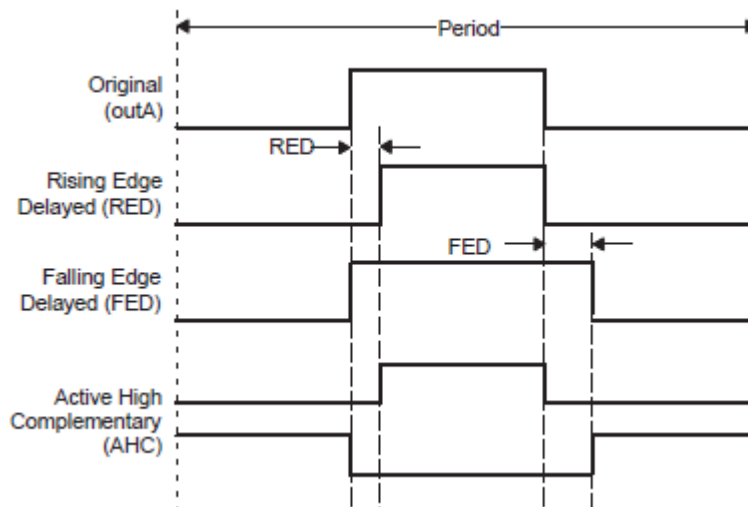


Fig. 6.7 – Inserzione del tempo morto. Il primo segnale in alto è quello inalterato, ovvero un’onda quadra con il duty-cycle richiesto dal controllo. Il secondo ed il terzo segnale risultano rispettivamente dall’inserimento di un ritardo sul fronte di salita e su quello di discesa. Gli ultimi due segnali rappresentano le uscite effettive dei canali A e B di un modulo ePWM impostato in modalità AHC.

Assegnando un valore alla macro `DEADTIME`, presente in `user.h`, si regola la durata del tempo morto t_{Δ} , in microsecondi. La macro è espressa in centesimi di microsecondo, per cui si calcola (quando $TBCLK = EPWMCLK$):

$$DEADTIME = t_{\Delta} \cdot 100 \quad (6.3)$$

Con le informazioni fornite dal produttore degli IGBTs in [9], [12] e [13] si ricava la durata minima del tempo morto che è pari a $0.66 \mu s$, tuttavia i gate drivers sono dotati di una logica di controllo che verifica che il tempo morto sia effettivamente compreso tra 0.72 e $1.2 \mu s$. In via cautelativa, quindi, si sceglie $1 \mu s$.

La frequenza di commutazione f_{sw} viene impostata sempre in `user.h`, alla riga 6, modificando la macro `EPWM_TIMER_TBPRD`. La formula per calcolare `EPWM_TIMER_TBPRD` a partire dalla frequenza di commutazione f_{sw} desiderata è la seguente:

$$EPWM_TIMER_TBPRD = \frac{1}{2 \cdot f_{sw} \cdot TBCLK} \quad (6.4)$$

dove `TBCLK` è il “Time-base clock”, una versione scalata del clock della PWM (`EPWMCLK`). `TBCLK` vale 10^{-8} e determina la velocità con la quale il contatore che gestisce le uscite PWM incrementa o decrementa. `FSW`, espressa in hertz, va aggiornata di conseguenza, ma serve soltanto all’algoritmo di controllo, cioè non modifica le impostazioni della periferica ePWM.

Quest’ultima può essere usata per la generazione interrupts. Questa funzione è particolarmente utile per sincronizzare l’esecuzione dell’algoritmo di controllo e le conversioni analogico-digitali con l’inizio, ed eventualmente anche con la metà, del periodo di commutazione degli interruttori.

In particolare, sincronizzare la lettura delle correnti di fase con la PWM con può risultare molto vantaggioso per il seguente motivo. Se si fa coincidere l’istante di campionamento con l’inizio o la metà del periodo di commutazione, infatti, si

campiona quando lo stato dell'inverter è 000 o 111, cioè conducono contemporaneamente tutti gli switch inferiori o superiori rispettivamente. Affinché ciò valga è necessario che i duty-cycles delle tre gambe siano maggiori di un valore minimo ma minori di quello massimo, garantendo la presenza di un tempo durante il quale è applicato uno dei due vettori di stato nulli. Il valore letto tramite i sensori di corrente risulta essere pertanto il valore medio della corrente di fase, cioè l'armonica di frequenza fondamentale che è di interesse controllare. Si parla pertanto di "campionamento sincrono".

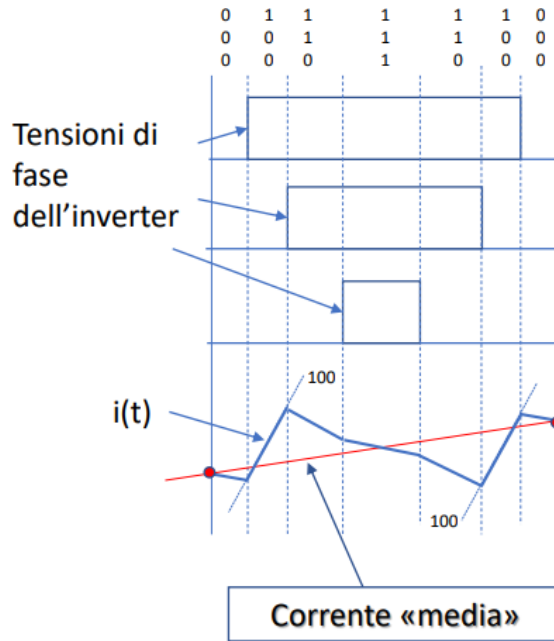


Fig. 6.8 – Campionando la corrente di fase all’inizio oppure nella mezzeria del periodo di commutazione, si misura automaticamente il valore medio, ovvero l’armonica fondamentale di corrente

Tale sincronizzazione viene meno quando uno dei duty-cycle D , degli interruttori inferiori o superiori risulti essere inferiore a $\frac{t_{\Delta}}{T_s}$. In tal caso, il periodo di accensione t_{ON} dello switch sarebbe inferiore alla durata del tempo morto:

$$t_{ON} < t_{\Delta}$$

per cui la periferica ePWM non genererebbe il segnale che va a comandare il gate driver.

È necessario pertanto potersi sincronizzare precisamente con i segnali PWM prodotti dal microcontrollore. Ovviamente i tre moduli ePWM usati per controllare le tre gambe dell’invertitore devono essere tra loro sincronizzati. Tramite oscilloscopio si è verificato che la suddetta condizione è verificata anche senza la scrittura di apposite istruzioni in codice.

Si sottolinea, infine, la necessità di sincronizzare l’aggiornamento dei duty-cycles con i segnali PWM. Inizialmente, infatti, si pensava di gestire la routine di controllo tramite uno dei tre timers resi disponibili dal microcontrollore. Questi timers, tuttavia, non sono precisi come il contatore dei moduli ePWM e, soprattutto, non possono essere sincronizzati con questi. Alimentando l’inverter di prova con tensioni

superiori ai 30 V il motore produceva rumori sospetti, dovuti a cicliche perdite del controllo dello stesso da parte dell'azionamento. Il problema è stato risolto impostando il modulo ePWM1 in modo che esso avvii la routine di controllo all'inizio (oppure all'inizio ed a metà) di ogni periodo di commutazione, in sincronia con l'aggiornamento dei duty-cycles con i valori calcolati tramite Space Vector Modulation nel precedente ciclo. La funzione di Shadowing, resa disponibile dal DB Submodule, consente infatti di "congelare" i valori dei duty-cycles calcolati, per poi caricarli nei registri dei moduli ePWM solo quando il contatore TBCTR passa per determinati valori.

Tabella 6.1

Modulo ePWM	Canale	Pin	IGBT controllato
1	A	40	T_{B+}
	B	39	T_{B-}
2	A	38	T_{A+}
	B	37	T_{A-}
6	A	76	T_{C+}
	B	75	T_{C-}
4	-	-	Generazione clock per lettura encoder

Una modifica che migliorerebbe leggermente le prestazioni del microcontrollore, a discapito di una leggera complicazione del codice, è la seguente. Il modulo ePWM1 avvia solo le conversioni ADC, senza avviare alcuna Interrupt Service Routine. Una volta terminate le prime, il convertitore analogico - digitale genera un interrupt e richiama la ISR che contiene il controllo del motore. Così facendo, la durata temporale delle conversioni non andrebbe ad occupare una porzione della ISR che contiene il controllo, il quale può essere eseguito solo in un intervallo di tempo limitato, prima della generazione del prossimo interrupt. Come verrà spiegato nel paragrafo 6.5, attualmente non ci sono problemi di durata di esecuzione del codice per cui questa soluzione non è stata implementata.

6.2 La periferica ADC

L'algoritmo di controllo motore proposto è a catena chiusa, ovvero necessita di misure sul motore derivanti da appositi sensori. In particolare, per controllare la coppia erogata dal motore è necessaria una retroazione sulle correnti di fase assorbite dalla macchina. Per un corretto calcolo dei duty-cycles occorre inoltre disporre della misura della tensione del DC-link. Occorre, infine, misurare le temperature dei semiconduttori di potenza e quella interna del motore, al fine di verificare che esse non raggiungano valori critici.

Tutte le grandezze elencate possono essere misurate tramite sensori di tipo analogico. Il microcontrollore, tuttavia, elabora informazioni digitali, motivo per cui è necessaria la conversione dei segnali analogici. La periferica che si occupa della conversione è detta Convertitore Analogico-Digitale (ADC).

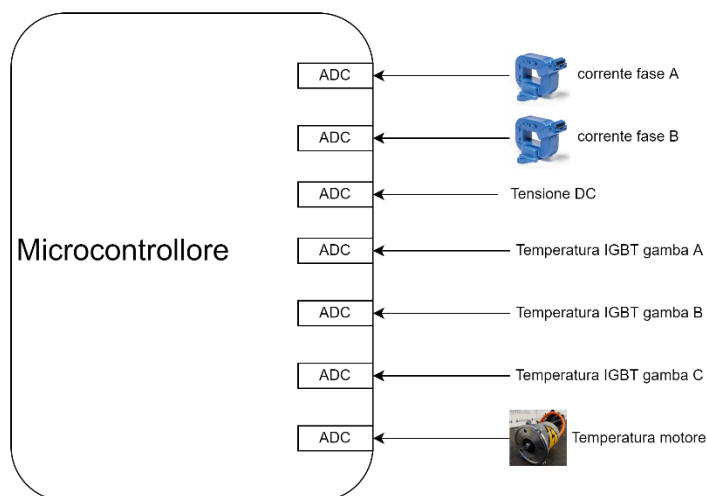


Fig. 6.9 – Schema dei sensori letti tramite periferica ADC

Il microcontrollore TMS320F28379D è provvisto di una periferica ADC ad approssimazioni successive (SAR), cioè in grado di calcolare la migliore approssimazione digitale del segnale analogico campionato. Le tensioni date in ingresso ai convertitori devono essere comprese tra 0 e 3 V, mentre la risoluzione della conversione può essere scelta tra 12 o 16 bit.

Scegliendo la risoluzione a 12 bit, i valori restituiti dal convertitore variano tra 0 e $2^{12} - 1$. Tale modalità di funzionamento prende il nome di “Single-ended mode”, in quanto è richiesto un solo pin per la lettura del polo positivo della sorgente (in questo caso del generico sensore). Il polo negativo della stessa viene collegato al Ground del microcontrollore.

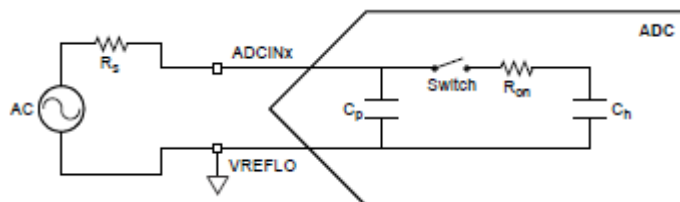


Fig. 6.10 – Modello dello stadio di ingresso in modalità “Single-ended”. AC è la sorgente di tensione della quale si vuole misurare il valore

Scegliendo la risoluzione a 16 bit, i valori restituiti variano tra 0 e $2^{16} - 1$. Tale modalità prende il nome di “Differential input mode”, in quanto il microcontrollore legge la tensione come differenza di due segnali e non come valore di un polo rispetto a Ground.

Questa modalità è vantaggiosa poiché l’eventuale rumore presente su entrambi gli input viene largamente cancellato, tuttavia sono necessari due pin del microcontrollore per leggere il segnale analogico [16].

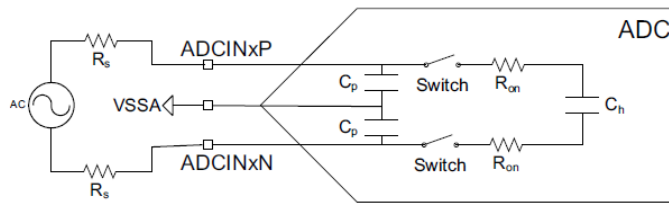


Fig. 6.11 – Modello dello stadio di ingresso in modalità “Differential input”

Le correnti di fase possono essere misurate tramite resistenze di shunt oppure tramite sensori ad effetto Hall.

Le prime non sono altro che resistenze che vengono attraversate dalla corrente che si intende misurare. Si manifesta così una caduta di tensione ai loro capi, proporzionale alla corrente che vi scorre, che può essere misurata previa l’interposizione di un dispositivo che realizza l’isolamento galvanico, dato che altrimenti il circuito di misura sarebbe circa allo stesso potenziale del conduttore.

I sensori ad effetto Hall, invece, realizzano intrinsecamente l’isolamento galvanico, in quanto sfruttano il principio del trasformatore. La corrente da misurare I_p scorre nel conduttore ed induce del flusso magnetico nel nucleo ferromagnetico posizionato attorno ad esso, che costituisce il sensore. Il valore di induzione B nel nucleo, nella regione lineare del ciclo d’isteresi, è proporzionale alla corrente I_p . Nel nucleo è interposto un volume di materiale che sfrutta l’effetto Hall: ai capi del materiale, attraversato anch’esso dal flusso, si manifesta una tensione proporzionale all’induzione al suo interno. Questa tensione viene amplificata da un amplificatore a bassa frequenza, il quale rende disponibile in uscita quindi una tensione proporzionale alla corrente I_p . La banda limitata dell’amplificatore agisce da filtro per limitare il rumore a banda larga. Con queste sonde è possibile misurare anche correnti continue, senza incorrere nella saturazione del nucleo, grazie alla compensazione attiva.

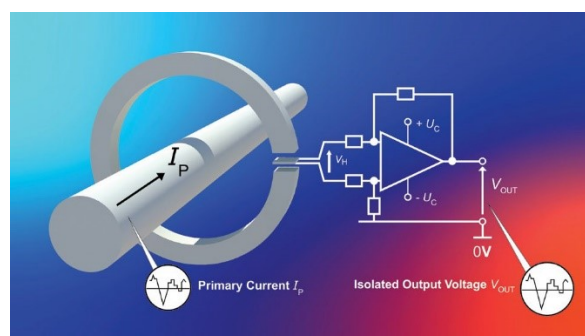


Fig. 6.12 – Schema di principio di una sonda ad effetto Hall

Dato che la targa del motore riporta una corrente massima di circa 150 A picco, si richiede che i sensori scelti siano in grado di misurare un valore sufficientemente più grande di corrente, onde evitare che sovracorrenti di fase non siano rilevabili a causa della saturazione del nucleo. La scelta ricade sul modello HO 150-S-0100 della LEM International SA, in grado di misurare correnti comprese tra +375 A e -375 A picco. Questo sensore viene alimentato a 5 V, valore già utilizzato nelle schede di controllo.

Il sensore restituisce ai suoi capi una tensione di 5.333 mV per ogni Ampere che scorre al primario. La tensione d'uscita è centrata attorno a 2.5 V, in quanto misurata tra i pin U_{out} ed U_{ref} del sensore, quindi correnti positive produrranno tensioni superiori a 2.5 V, correnti negative tensioni inferiori. Convenzionalmente, la corrente è positiva quando scorre dall'inverter al motore e negativa in caso contrario. Gli ADC del microcontrollore, tuttavia, accettano in ingresso una tensione massima di 3 V, corrispondenti a:

$$\frac{(3 - 2.5)1000}{5.333} = 93.8 \text{ A}$$

Fortunatamente, è possibile fornire in ingresso al sensore una tensione di riferimento al pin U_{ref} , corrispondente alla tensione restituita in uscita con corrente nulla. Fornendo 1.5 V, il sensore fornisce una tensione proporzionale alla corrente che scorre al primario, centrata proprio intorno ad 1.5 V. Tenendo in considerazione la tensione limite degli ADC, si ricava che il massimo valore di corrente misurabile diventa:

$$\frac{(3 - 1.5)1000}{5.333} = 281 \text{ A}$$

Ovviamente, lo stesso vale per correnti di segno opposto, alle quali corrisponderanno tensioni comprese tra 0 e 1.5 V.

La banda passante dichiarata dal costruttore è pari a 100 kHz, pertanto una corretta misurazione della corrente su tutto l'intervallo di frequenze di funzionamento del motore è garantita.



Fig. 6.13 – Sensore ad effetto Hall HO 150-S-0100 usato per la misura delle correnti di fase

Per la legge di Kirchhoff delle correnti, essendo il motore trifase senza centro stella, è sufficiente installare due sensori per misurare due correnti e la terza si ricava come la somma algebrica delle prime due, cambiata di segno:

$$i_a + i_b + i_c = 0 \Rightarrow i_c = -(i_a + i_b) \quad (6.5)$$

Ovviamente, questa relazione è valida istantaneamente. Ciò comporta la necessità di campionare le due correnti di fase nello stesso istante. Le conversioni delle misure di corrente sono pertanto avviate automaticamente all'inizio del periodo di commutazione dal contatore del modulo ePWM1 (ed anche a metà periodo, in caso di SVM asimmetrica).

Prima di essere portato in ingresso all'ADC, il segnale proveniente dal sensore di corrente attraversa un amplificatore differenziale a guadagno unitario ed un filtro passa-basso del primo ordine del tipo RC. Per una descrizione dettagliata dello stadio di condizionamento dei segnali provenienti dai sensori di corrente si rimanda a [8]. Si riporta il codice della funzione che converte la lettura dell'ADC in valore di corrente di fase. Il segno meno è dovuto ad un aggiustamento temporaneo dello stadio di condizionamento del segnale, in uscita dall'amplificatore differenziale. Tramite la variabile offset è possibile compensare, in Ampere, eventuali letture di corrente non nulle quando la corrente che effettivamente scorre verso il motore è nulla. Tali errori di misura sono dovuti allo stadio d'ingresso degli ADC.

```
float getCurrent(float raw_data, float offset){
    return -((VREFHI*1000.0f*(2.0f*raw_data/65536.0f -
1.0f)/LEM_mV_TO_AMP) - offset);
}
```

La tensione DC viene misurata tramite un partitore di tensione ed un amplificatore isolato a guadagno unitario.

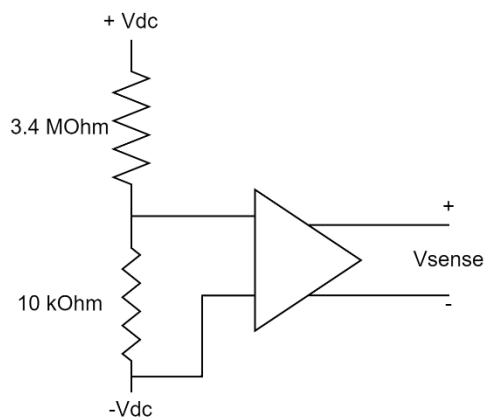


Fig. 6.14 – Schema semplificato dello stadio di lettura della tensione lato continua

```
float getDCvoltage(float raw_data){ //raw_data: tensione in uscita
dall'amplificatore
    return raw_data*0.249816849f; //Vdc = Vin*341 = lettura/4095*3*341 [V]
}
```

Al fine di consentire il monitoraggio delle temperature interne, sia il modulo IGBT che il motore sono provvisti di termistori. La variazione di resistenza di questi sensori è proporzionale alla temperatura dei componenti, quindi tramite ingressi ADC il microcontrollore può periodicamente misurare la caduta di tensione sui termistori e, pertanto, ricavare la temperatura rilevata dagli stessi. Lo schema di alimentazione dei quattro termistori è lo stesso: è necessario un partitore resistivo per alimentare i sensori di temperatura, un amplificatore differenziale (isolato per il modulo IGBT,

non isolato per il motore) rende poi disponibile ai suoi morsetti la tensione corrispondente alla resistenza misurata.

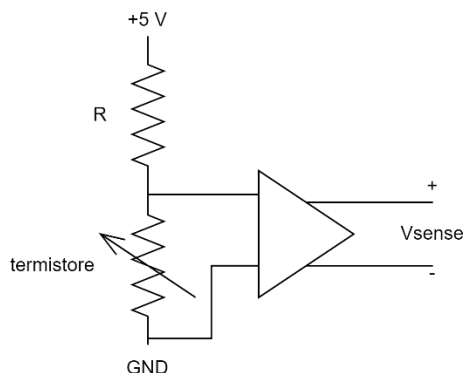


Fig. 6.15 – Schema semplificato dello stadio di lettura dei termistori

Il modulo IGBT FS380R12A6T4LB è dotato di tre termistori NTC (Negative Temperature Coefficient), quindi con resistenza decrescente all’aumentare della temperatura, uno per ogni semiponte. La formula che restituisce la temperatura, in gradi Celsius, ricavata dalla resistenza rilevata è stata ricavata da [11] e [13].

```
float getIGBTtemp(float raw_data){
    raw_data *= 0.000732421f; //converto la tensione da 0-4095 a 0-3 V
    float R;
    R = 3300.0f*raw_data/(5.0f - raw_data); //resistenza del termistore

    return 1.0f/((logf(R/5000.0f)/3433.0f)+1/298.15f)-273.15f; //pag. 30
    "Hybridpack application note"
}

```

Il termistore del motore, invece, è del tipo KTY84. La curva resistenza-temperatura può essere approssimata con due spezzate, come fatto da AMK. Tramite il software proprietario Aipex Pro è possibile leggere i dati salvati nella memoria degli invertitori e ricostruire le spezzate. In particolare, all’indirizzo ID34166 vengono forniti i dati utili all’interpolazione, che portano alla seguente funzione, che restituisce sempre la temperatura, in gradi.

```
float getMotorTemp(float raw_data){
    raw_data *= 0.000732421f; //converto la tensione da 0-4095 a 0-3 V
    float R;
    R = 560.0f*raw_data/(5.0f - raw_data); //resistenza del termistore

    //termistore KTY84, resistenza [Ohm]: 603.2 @25°C, 853.6 @75°C, 1160
    @125°C
    if(R <= 853.6f) return 0.1997f*R - 95.459f;
    else return 0.1632f*R - 64.312f;
}

```

Le temperature vengono lette alla frequenza di 1 Hz, sfasate di 0.25 secondi l'una dall'altra. I transistori termici, infatti, sono notoriamente più lenti di quelli elettrici e pertanto non richiedono elevate frequenze di campionamento.

Le misure più critiche per il funzionamento corretto dell'algoritmo di controllo sono quelle di corrente, pertanto si è scelto di adottare come risoluzione 16 bit. Le altre misure vengono eseguite con risoluzione pari a 12 bit, necessitano pertanto di un solo pin ciascuna e sono più rapide di quelle a 16 bit.

Si riportano in tabella le impostazioni adottate per le conversioni ADC.

Tabella 6.2

Grandezza	Modulo ADC	Canali	Pin	Risoluzione	Trigger
Corrente fase A	A	2-3	26,29	16 bit	ePWM1 SoCA
Corrente fase B	B	4-5	65,68	16 bit	ePWM1 SoCA
Tensione DC	C	3	24	12 bit	ePWM1 SoCA
Temperatura IGBT gamba A	C	2	27	12 bit	software
Temperatura IGBT gamba B	C	15	63	12 bit	software
Temperatura IGBT gamba C	C	5	64	12 bit	software
Temperatura motore	C	4	67	12 bit	software

Tutti i risultati grezzi delle conversioni vengono salvati nell'array `analogValues`, prima di essere elaborati da `getCurrent`, `getDCvoltage`, `getIGBTtemp` e `getMotorTemp`.

6.3 Interfaccia per la lettura encoder

I motori forniti da AMK sono dotati di encoder rotativi induttivi di tipo assoluto, modello ECI 1118 della Heidenhain. Essi comunicano in modo digitale, tramite il protocollo EnDat, la posizione angolare del rotore rispetto allo statore. Si ricorda che la conoscenza della posizione elettromeccanica di rotore θ_{me} è un requisito fondamentale per poter eseguire un controllo ad orientamento di campo (Field Oriented Control, FOC) come descritto nei capitoli precedenti.

La comunicazione seriale con protocollo EnDat necessita di sei linee: +5 V e GND per alimentare l'encoder, CLOCK+ e CLOCK- per fornire il segnale di clock e DATA+ e DATA- per la trasmissione dei dati. Sia il clock che i dati sono trasmessi in modo differenziale.

Il modulo B della SPI sulla scheda LAUNCHXL-F28379D è parte integrante della comunicazione tra scheda ed encoder, in quanto utilizzato per emulare l'interfaccia EnDat, pertanto non può essere utilizzato per comunicare con i gate drivers.

La risoluzione dell'encoder è pari a 18 bit, pertanto il valore restituito dal sensore è sempre compreso tra 0 e $2^{18} - 1$. I motori sono a $2p = 10$ poli, pertanto la posizione

elettrica relativa tra statore e rotore varia tra 0 e $p \cdot 2\pi$. Occorre quindi associare i valori tra $[0, 2^{18} - 1]$ ai loro corrispondenti nell'intervallo $[0, p \cdot 2\pi]$ ovvero $[0, 10\pi]$. Si può semplicemente pensare di ricavare l'angolo elettrico θ_{me} come:

$$\theta_{me} = \frac{\text{lettura_encoder}}{\text{risoluzione_encoder}} 10\pi \quad [\text{radianti elettrici}] \quad (6.6)$$

Per fare ciò, tuttavia, occorre conoscere la posizione iniziale tra il sistema di riferimento dq sincrono col rotore e quello $\alpha\beta$, solidale con lo statore. In altre parole, occorre capire quale è la posizione del rotore quando l'encoder restituisce un valore nullo.

Una tecnica molto diffusa consiste nell'applicazione di una piccola corrente positiva costante lungo l'asse α del sistema di riferimento. Così facendo, il rotore tenderà ad allineare il proprio asse d, situato in corrispondenza della mezzeria di un polo nord di rotore, con l'asse α . Il valore letto dall'encoder a transitorio esaurito può essere assunto come offset per compensare le misure provenienti dal sensore.

Questo principio di allineamento presenta un piccolo svantaggio, legato agli attriti di natura meccanica, i quali fanno sì che l'allineamento tra gli assi d ed α non sia mai perfetto.

Fortunatamente, AMK provvede già ad orientare l'encoder in modo tale che quando l'encoder legge zero si ha l'allineamento tra polo nord di rotore ed asse α di statore. Si possono comunque eseguire delle verifiche con la tecnica sopra descritta, eseguendo varie letture dell'encoder. I valori ottenuti risultano essere tutti in prossimità di 0 o $n \frac{2^{18}}{5} \simeq n \cdot 52429$, con n numero intero compreso tra 1 e 5. Avendo il motore 5 coppie polari, è lecito aspettarsi cinque diversi valori di allineamento, a seconda di quale dei cinque poli Nord di rotore si allinei con quale degli altrettanti assi α di statore [6].

Tramite due letture consecutive della posizione, rispettivamente agli istanti kT_s e $(k+1)T_s$, è possibile calcolare la velocità di rotazione media del motore come derivata media della posizione sul periodo di campionamento T_s :

$$\Omega_{me} = \frac{\theta_{me,k+1} - \theta_{me,k}}{T_s} \quad [\text{radianti elettrici/s}] \quad (6.7)$$

T_s è fisso e pari all'inverso della frequenza del controllo f_c . Per ottenere una indicazione precisa della velocità, occorre naturalmente eseguire le letture dell'encoder con cadenza regolare. Si pensi ad esempio se la posizione venisse letta continuamente dopo una porzione di codice di durata variabile: nella formula per ottenere Ω_{me} compare al denominatore un valore fisso, mentre l'effettivo tempo trascorso tra le due letture successive potrebbe essere ben diverso e non costante.

Uno svantaggio di questo metodo è costituito dal fatto che la derivata amplifica i rumori presenti nel sistema, con un effetto negativo sull'algoritmo di controllo che potrebbe calcolare una velocità oscillante quando magari il motore sta ruotando ad una velocità costante.

La velocità Ω_{me} viene utilizzata nel calcolo dei termini di disaccoppiamento degli assi dq e di compensazione della forza controlettromotrice. Disturbi nel calcolo di Ω_{me} si ripercuotono pertanto sia sull'azione del regolatore di velocità, ma anche nel

calcolo dei riferimenti di tensione per l'invertitore. Per eliminare disturbi ad alta frequenza introdotti nell'anello di controllo dalla derivata digitale si è implementato un filtro a media mobile a 5 o 10 elementi, a valle di **readEncoder**.

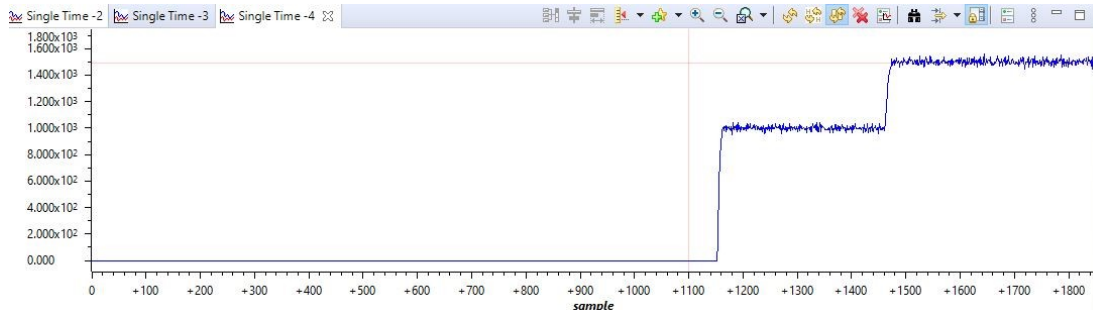


Fig. 6.16 - Velocità calcolata con motore in rotazione, rappresentata graficamente con l'interfaccia di debug di Code Composer Studio. Sulle ordinate si legge la velocità in giri al minuto. I campioni, in debug, sono presi dal microcontrollore ed inviati al pc con una frequenza massima di 10 Hz.

Occorre poi tenere presente che, se tra due letture successive di posizione dovesse avvenire un passaggio per lo zero, si avrebbe improvvisamente una lettura di velocità di segno opposto rispetto alle precedenti e di elevato modulo, con conseguente intervento del regolatore di velocità. Ciò va evitato in quanto porterebbe a brusche variazioni di coppia richiesta dal controllo. Nel codice viene pertanto eseguito una verifica sui valori di velocità calcolati: se risulta una differenza $\Delta\theta_{me} = \theta_{me,k+1} - \theta_{me,k}$ in modulo maggiore di 1 radiante elettrico, allora è avvenuto un passaggio per lo zero. Si noti come un $\Delta\theta_{me}$ pari a 1 significherebbe che, rilevato con un tempo di campionamento di $50 \mu s$, il motore starebbe girando a 38197 rpm, una velocità quasi doppia a quella massima consentita e quindi non plausibile.

È stata inserita, infine, una riga di codice che serve ad evitare il calcolo di valori di velocità diversi da zero quando il motore è fermo. Questo si rende necessario poiché l'encoder possiede una elevata risoluzione e spesso due letture successive differiscono di poche unità, anche quando il rotore è immobile.

Si riporta sotto la funzione **readEncoder**, definita in `epwm_management.h`, che gestisce la lettura della posizione ed il calcolo della velocità. Le variabili da essa manipolate sono la posizione elettromeccanica (θ_{me}) "theta_me", campionata all'istante kT_s e "prev_theta", campionata all'istante $(k-1)T_s$. La velocità Ω_{me} calcolata è salvata in "omega_me". "prev_pos" corrisponde al valore non elaborato, restituito dall'encoder nella lettura precedente all'istante $(k-1)T_s$.

```
void readEncoder(uint32_t *prev_pos, float *prev_theta, float *theta_me,
float *omega_me)
{
    uint16_t crc5_result; //variable for calculated crc result checking
    uint16_t retval1;
    uint32_t pos; //pos = position_lo
    //ENDAT_DATA_STRUCT endat22Data;
    retval1 = PM_endat22_setupCommand (ENCODER_SEND_POSITION_VALUES, 0, 0,
0);
    PM_endat22_startOperation();
    //while (endat22Data.dataReady != 1) {}
    retval1 = PM_endat22_receiveData(ENCODER_SEND_POSITION_VALUES, 0);
}
```

```

    // CRC check for Position Data
    crc5_result = PM_endat22_getCrcPos(endat22Data.position_clocks,
ENDAT21, endat22Data.position_lo, endat22Data.position_hi,
endat22Data.error1, endat22Data.error2, endat22CRCTable);

    if (!CheckCRC(crc5_result, endat22Data.data_crc))
    {
        ESTOP0;
    }
else
{
    pos = endat22Data.position_lo;
}

//calcolo della posizione
*theta_me = (pos*2.0f*M_PI*p)/ENCODER_RESOLUTION; //angolo elettrico
[rad.el]

//calcolo della velocità
if (abs((pos-(*prev_pos))) < 6) { //isteresi
    *omega_me = 0.0f;
}
else //motore effettivamente in movimento
{
    *omega_me = SDR*((*theta_me)-*prev_theta)/TS; //[rad_el/s]
}

//se tra due letture consecutive di posizione passo per lo zero
if ((*theta_me - *prev_theta) < -1.0f) { //velocità positiva
    *omega_me = SDR>(*theta_me - *prev_theta + 2.0f*M_PI*p)/TS;
}

if ((*theta_me-*prev_theta) > 1.0f) { //velocità negativa
    *omega_me = SDR>(*theta_me - *prev_theta - 2.0f*M_PI*p)/TS;
}

*prev_pos = pos;
*prev_theta = *theta_me;
}

```

La Texas Instruments fornisce esempi funzionanti, come quello usato in questa tesi, che fanno uso della libreria PM_endat22, sviluppata specificamente per gestire la comunicazione attraverso il protocollo EnDat. La funzione **readEncoder** è stata scritta partendo dall'esempio PM_endat22_BasicPosAcc_DelComp messo a disposizione da Texas Instruments, presente all'indirizzo C:\ti\controlSUITE\development_kits\BOOSTXL_POSMGR.

L'integrazione dell'esempio nel progetto che contiene la routine di controllo si è rivelata molto difficoltosa, per motivi di compatibilità software, a tal punto che è riuscita solo con l'intervento diretto della stessa Texas Instruments. Per completezza si riporta in bibliografia il link al thread sul forum e2e.ti.com, il quale riporta i passaggi per la soluzione del problema [7].

Per quanto concerne l'interfaccia fisica tra encoder e microcontrollore, la Texas Instruments commercializza la scheda TIDM-1008 o "Position Manager BoosterPack". Essa consente di interfacciare la LAUNCHXL-F28379D con una moltitudine di sensori di posizione, dai resolver agli encoder assoluti con interfacce EnDat 2.2, EnDat 2.1, BiSS-C e T-format.

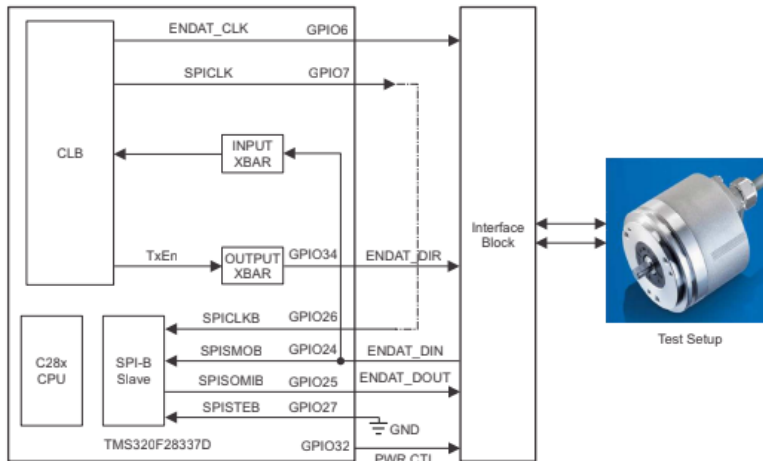


Fig. 6.17 - Schematico dei collegamenti tra TIDM-1008 e scheda F28379D

Nel datasheet dell'inverter AMK è specificato che la versione del protocollo EnDat utilizzata per comunicare con l'ECI 1118 è la 2.2. Per far funzionare la comunicazione tra microcontrollore ed encoder, tuttavia, è stato necessario modificare la macro `ENCODER_TYPE` che si trova alla riga 29 del file `endat.h`, assegnandole il valore di 21 invece che 22, corrispondente alla versione 2.1 del protocollo EnDat. Risultando la comunicazione funzionante con la versione 2.1 non si è ulteriormente indagato sul perché la 2.2 non funzioni.

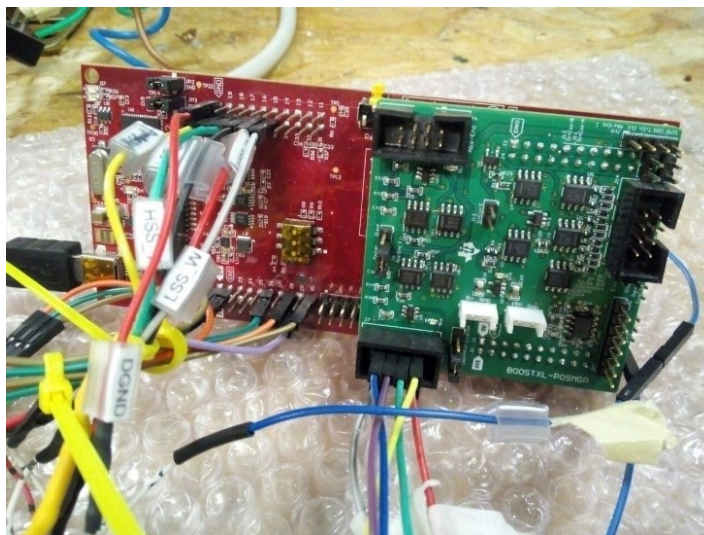


Fig. 6.18 - Scheda TIDM-1008 (in verde) montata sulla LAUNCHXL-F28379D. Il connettore usato per comunicare con l'encoder è il J7

Come si può notare dall'immagine soprastante, il Position Manager occupa circa metà della superficie superiore della LAUNCHXL-F28379D, coprendo tutti i 40 pin dei

raggruppamenti J5, J6, J7, J8. In questa ottica, potrebbe essere conveniente riprogettare l'interfaccia per renderla meno ingombrante, dal momento che sulla soluzione commerciale sono presenti più connettori e componenti di quelli strettamente necessari per comunicare con un encoder. Per il primo prototipo di invertitore, tuttavia, tale soluzione è stata giudicata sufficiente.

6.4 Funzionalità di protezione

Nel firmware del microcontrollore sono implementate le seguenti funzioni di protezione:

- verifica di presenza e corretto funzionamento dell'encoder
- monitoraggio delle correnti di fase
- monitoraggio della tensione del DC bus
- monitoraggio della velocità di rotazione del motore
- monitoraggio delle temperature interne del motore e delle tre gambe del modulo IGBT

Per quanto riguarda l'encoder, le verifiche sul suo corretto funzionamento sono implementate all'interno del file `endat.c`, presente negli esempi messi a disposizione dalla Texas Instruments. Al verificarsi di un'anomalia nella comunicazione viene eseguita la funzione **encoder_error_management** che prevede la disattivazione del segnale di enable dei gate drivers e l'accensione dei LEDs rosso e blu (D9 e D10 rispettivamente) della LAUNCHXL-F28379D. La direttiva ESTOPO provvede poi a bloccare l'attività del microcontrollore, similmente a quanto farebbe un ciclo for di durata infinita.

Attualmente, il microcontrollore non è in grado di verificare che entrambi i sensori di corrente siano collegati alla rispettiva scheda e funzionanti. In assenza del sensore, infatti, il corrispondente ADC leggerebbe una tensione pari a 0 V, in quanto entrambi i pin sarebbero a 1.5 V. Si può pensare di implementare, nella versione successiva della scheda condizionamento segnali, dei resistori di pull-down che trascinino a massa uno dei due pin del sensore di corrente, in modo che quando un sensore non fosse collegato, la rispettiva periferica ADC leggerebbe una tensione data dalla differenza tra il riferimento (1.5 V) e massa (GND). Tale tensione corrisponde, come già visto, a circa 281 A di corrente di fase e pertanto farebbe scattare la protezione da sovracorrente, bloccando l'invertitore prima che possano verificarsi incidenti dovuti a sensori di corrente scollegati.

Gli altri errori sono invece gestiti dalla funzione **general_errors_management**. Essa verifica che le correnti di fase, la tensione DC, la velocità del motore, le temperature di motore ed IGBTs non superino determinati limiti, impostabili tramite delle macro in `user.h`. Se tali soglie vengono superate, la PWM viene disabilitata, si disabilitano i gate drivers e viene acceso il LED rosso. Infine, il microcontrollore entra in un ciclo for infinito. La sopracitata funzione viene eseguita ad ogni Interrupt Service Routine, immediatamente a valle delle conversioni ADC.

Una caso particolare si verifica quando la tensione del DC bus scende sotto alla soglia minima. Questa situazione si può verificare sia in caso di guasto ma anche nel normale funzionamento dell'invertitore, prima che i condensatori di DC-link vengano precaricati. Questa ultima operazione deve essere eseguita ad ogni accensione del circuito ad alta tensione, cioè ogniqualvolta viene accesa la vettura. Per tale motivo è impensabile far bloccare il microcontrollore; la soluzione consiste nel tenere la PWM spenta finché la tensione lato DC è minore della soglia impostata con la macro `DC_VOLTAGE_MIN`.

6.5 Accorgimenti per velocizzare l'esecuzione del codice

La routine di controllo viene eseguita ad intervalli di tempo regolari e ben definiti, come descritto nei precedenti capitoli. Ne consegue che tutte le operazioni relative al controllo del motore devono essere eseguite tra due interrupt successivi generati dal modulo ePWM1. Occorre quindi prestare attenzione alla complessità computazionale del codice, onde evitare che la sua esecuzione necessiti di un tempo maggiore del periodo del controllo T_s .

In questo lavoro di tesi sono stati presi alcuni accorgimenti per rendere più rapida l'esecuzione della routine di controllo.

Innanzitutto, il codice è stato sviluppato a partire da un esempio disponibile nel pacchetto C2000WARE, messo a disposizione dalla Texas Instruments. Tutti i programmi ivi contenuti sono progettati per funzionare su una scheda chiamata ControlCard, che ha un quartz clock di 20 MHz, mentre quello del LaunchPad che è di 10 MHz. Per adattare i programmi vanno quindi eseguiti i seguenti passaggi, nell'ambiente Code Composer Studio: prima cliccare col tasto destro del mouse sul progetto nel Project Explorer, quindi Properties > Build > C2000 Compiler > Predefined Symbols. Nella box "Pre-define NAME" aggiungere `_LAUNCHXL_F28379D`, quindi cliccare "Apply and Close". Lo stesso va fatto anche per per "Configuration: CPU1_FLASH". Questo semplice accorgimento ha permesso di raddoppiare la velocità di esecuzione del codice [5]. Questa modifica dei Predefined Symbols consente di ottenere la frequenza di clock massima e pari a 200 MHz.

Si è poi scelto di precalcolare i valori di alcuni numeri frazionari e costanti matematiche, salvandone il valore in apposite macro definite in user.h. In tal modo il microcontrollore non deve calcolarne il valore ogni volta che viene eseguita la routine di controllo.

```
#define SQRT_3          1.73205080756887729353 //radice di tre
#define DUE_TERZI     0.66666666666666666666 // 2/3
#define PI_TERZI      1.04719755119659774615 //pi greco terzi
#define DUE_TERZI_PI  2.09439510239319549231 //due terzi pigreco
#define QUATTRO_TERZI_PI 4.18879020478639098462 //quattro terzi pigreco
```

Molte funzioni matematiche presenti nella libreria math.h sono adatte a manipolare variabili di tipo "double", a doppia precisione ovvero a 64 bit. Nel codice, tuttavia, si è scelto di usare solo variabili floating-point (brevemente, "float") a 32 bit. Ogniqualvolta una funzione scritta per lavorare con variabili del tipo "double" ne

prende in ingresso del tipo "float", il microcontrollore esegue il cosiddetto typecasting della variabile trasformandola [17]. Una volta eseguita la funzione, se il risultato va salvato in una variabile "float", trasforma nuovamente. Al fine di evitare queste inutili operazioni che richiedono del tempo per essere eseguite, si è scelto di definire tutte le variabili floating-point nel seguente formato: X.Yf, dove X è la parte intera ed Y quella decimale, seguite da una "f" che sta ad indicare esplicitamente il tipo della variabile. Le funzioni **sin**, **cos**, **atan**, **fabs** che calcolano rispettivamente seno, coseno, arcotangente e modulo dell'input, sono state sostituite a favore di **sinf**, **cosf**, **atanf**, **fabsf** adatte a lavorare con variabili floating-point a precisione singola.

Laddove non necessario, si è evitato l'uso di funzioni computazionalmente onerose come l'arcotangente. Un esempio è costituito dalla routine di riconoscimento del settore di appartenenza del vettore spaziale del riferimento di tensione, all'interno della funzione che implementa l'algoritmo di Space Vector Modulation.

Infine, quasi tutte le variabili locali sono state definite statiche. Una variabile locale, cioè definita all'interno di una funzione, è visibile soltanto dalla funzione stessa ed esiste finché questa viene eseguita dal microcontrollore. Ogni volta che la funzione viene richiamata, la variabile in questione viene creata e successivamente cancellata. Questo si può evitare antepoendo al tipo della variabile il termine **static**, indicando che essa deve essere mantenuta nella memoria (con il proprio valore), pur continuando ad essere visibile solo localmente.

Il tempo di esecuzione della routine di controllo è stato misurato ed ammonta a 13.34 μ s, di cui 9.31 μ s sono occupati dalla lettura dell'encoder. Per le misure temporali si è monitorato, tramite oscilloscopio, lo stato logico di una GPIO, impostandone il valore a 1 all'inizio dell'intervallo da misurare ed a 0 alla fine dello stesso.

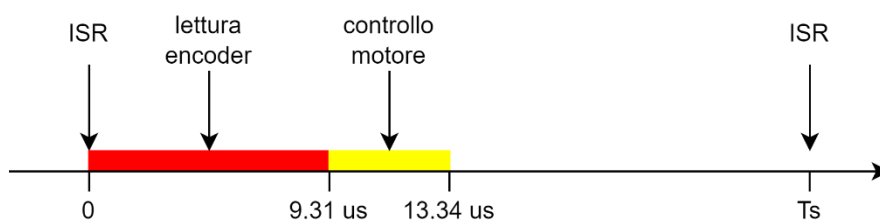


Fig. 6.19 - Durata temporale delle macrooperazioni eseguite dal microcontrollore, quando la sua frequenza di clock è pari a 200 MHz

6.6 Collaudo

L'algoritmo di controllo è stato sviluppato prima della costruzione del prototipo di convertitore, pertanto per testarne la funzionalità si è fatto uso di un invertitore gentilmente messo a disposizione da un'azienda sponsor del team di Formula Student (alpitronic SRL). Non sono note tutte le caratteristiche del dispositivo, in quanto dati confidenziali dell'azienda, tuttavia esso è stato fornito sulla base delle specifiche dei motori AMK, per cui non sorgono problemi di compatibilità. L'unico

dato noto riguarda il condensatore di DC-link, avente capacità nominale di 574 μF e tensione massima 437 V.

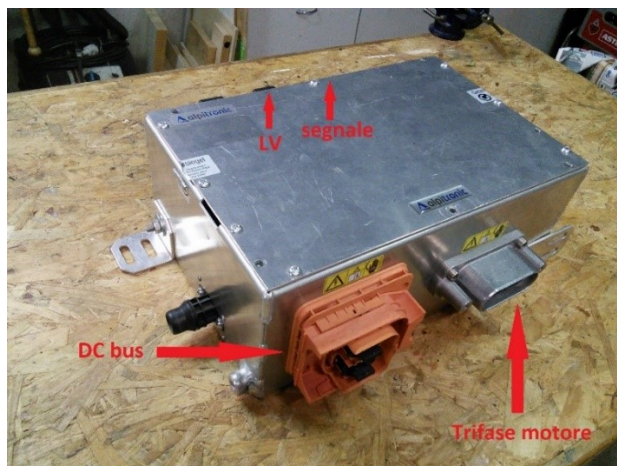


Fig. 6.20 - Inverter fornito dall'azienda sponsor

La parte in bassa tensione dell'inverter (LV) viene alimentata a 12 V, tramite un apposito cavo.

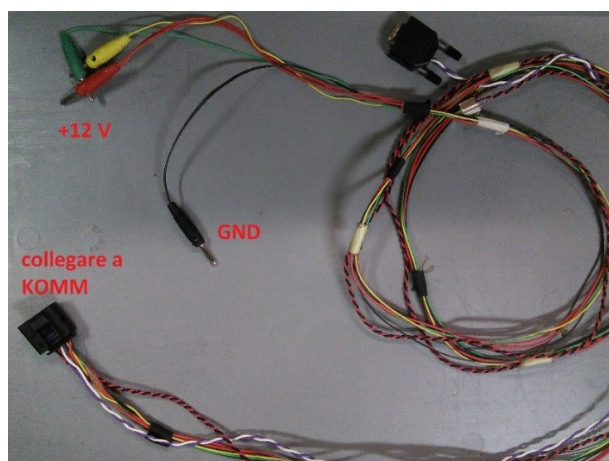


Fig. 6.21 - Cavo per alimentazione LV

I collegamenti di segnale sono molto delicati, dal momento che sono realizzati tramite un connettore da 50 pin appositamente realizzato dagli studenti del team. Il connettore va collegato all'invertitore, mentre l'altro lato del cavo è dotato di terminazioni di tipo "jumper", le quali possono essere collegate direttamente ai pin della scheda Texas. Ogni terminazione è dotata di etichetta per facilitare i collegamenti.

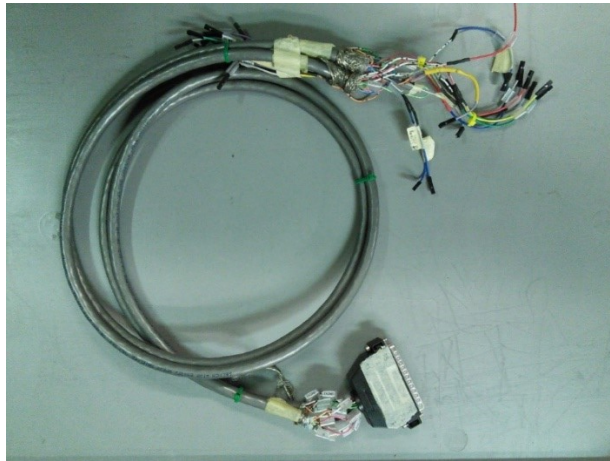


Fig. 6.22 - Cavo di segnale

Una eccezione è rappresentata dai segnali provenienti dai sensori di corrente montati all'interno della scatola di alluminio (modello HC5FW 300-S/SP1 della LEM International SA). Questi tre segnali sono contrassegnati come "LEM A", "LEM B", "LEM C" e sono proporzionali alle correnti di fase A, B, C. Dato che la lettura dei sensori non è filtrata all'interno dell'inverter, è stata predisposta una scheda millefori contenente, per ogni segnale: un filtro passa-basso RC, un inseguitore di tensione, un partitore per adattare il range 0-5 V in uscita dai sensori al range 0-3 V, accettato in ingresso dal microcontrollore.

I parametri del filtro sono $R = 30 \text{ k}\Omega$ e $C = 1 \text{ nF}$ che portano ad avere una frequenza di taglio $f_t = \frac{1}{2\pi RC} = 5305 \text{ Hz}$, corrispondente a poco più di tre volte il valore massimo della fondamentale del motore. Il filtro è necessario per evitare di introdurre negli anelli di controllo del rumore ad alta frequenza che deteriorerebbe le prestazioni dell'algoritmo di controllo.

La scheda millefori va alimentata a 5 V. I collegamenti +5 V e GND possono essere presi direttamente dalla scheda Texas. Le uscite A, B, C rappresentano le correnti di fase filtrate e possono essere portate ai pin del microcontrollore, come indicato nel pinout.

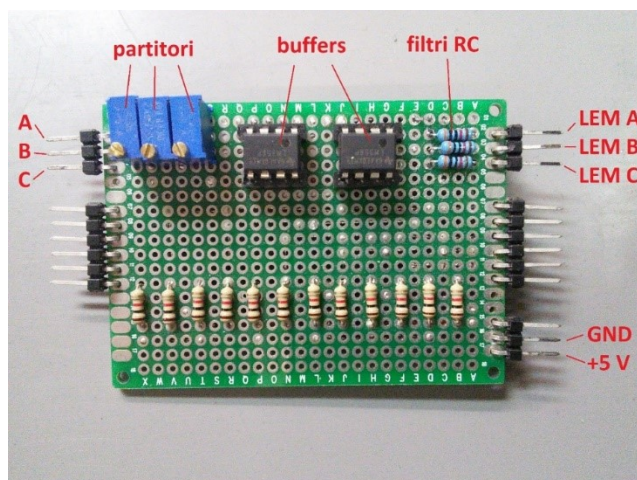


Fig. 6.23 - Scheda millefori

L'alimentazione del DC-link può essere fornita da un alimentatore da banco in corrente continua. La taglia 30 V - 10 A è sufficiente per far ruotare il motore a velocità inferiori o uguali a 1000 rpm, con coppia massima di 1 Nm.

Si consiglia sempre di controllare nel codice che sia commentato il regolatore PI che gestisce il deflussaggio, poiché esso non è mai stato testato ma interverrebbe comunque con una tensione ridotta lato continua, quali i 30 V.

Tabella 6.3

Nome segnale	Pin	Descrizione
LSS_W	35	PWM switch T_{C-}
HSS_W	36	PWM switch T_{C+}
LSS_V	37	PWM switch T_{B-}
HSS_V	38	PWM switch T_{B+}
LSS_U	39	PWM switch T_{A-}
HSS_U	40	PWM switch T_{A+}
DGND	GND	Ground digitale da cavo connettore 50 pin
A	30	ADC corrente fase A
B	28	ADC corrente fase B
C	27	ADC corrente fase C
Vdc	23	ADC tensione DC (previsto ma non usato)
AGND	7	Ground analogico da cavo connettore 50 pin
GT_EN + EN_LV	+5 V	Abilitazione LV inverter
DISC_DIS	+5 V	Disabilita discharge inverter

L'ultimo collegamento riguarda l'encoder. L'apposito cavo va collegato da un lato sul connettore del motore, dall'altro alla scheda TIDM-1008 montata sulla LAUNCHXL-F28379D.

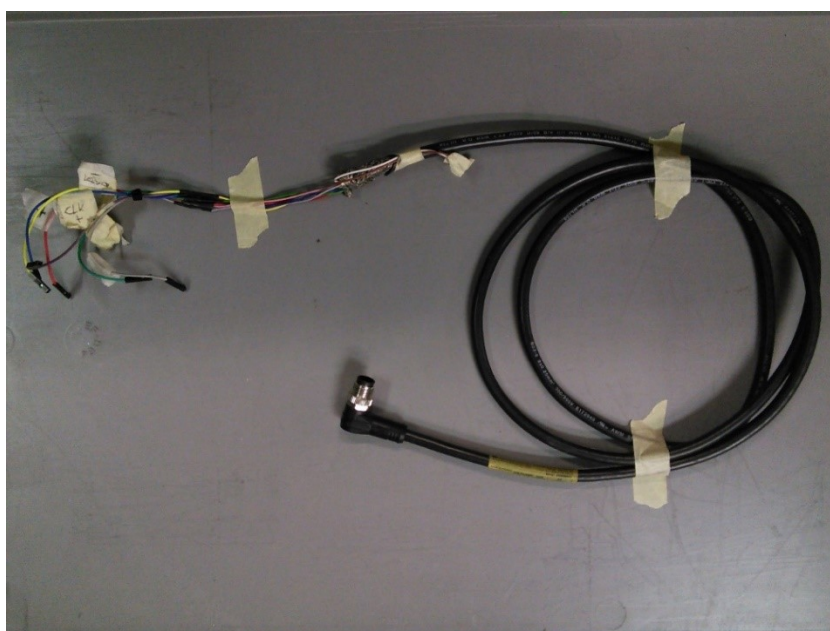


Fig. 6.24 - Cavo per la lettura dell'encoder

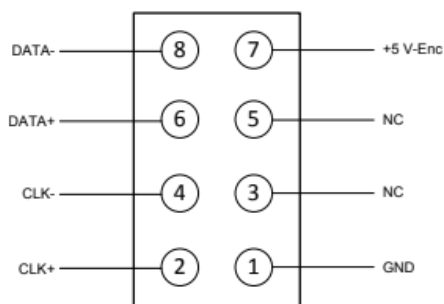


Fig. 6.25 - Pinout del connettore Abs-Enc-1 (J7) sulla scheda TIDM-1008. I pin utilizzati formano una "C", con i pin NC situati verso il bordo esterno della scheda

Tabella 6.4

Nome segnale	Colore cavo
+5 V	rosso
GND	grigio
DATA +	verde
DATA -	giallo
CLK +	Blu
CLK -	Viola

Una volta effettuati i collegamenti, tramite Code Composer Studio è possibile avviare la modalità di debug. Una volta caricato il programma nella memoria del microcontrollore, è possibile avviarlo e monitorare l'andamento delle variabili globali nella finestra Expressions.

```

X+y
=? "theta_me"
X+y
=? "speed_ref"
X+y
=? "speed_rpm"
X+y
=? "ia"
X+y
=? "ib"
X+y
=? "ic"
X+y
=? "pos_torque_lim"
X+y
=? "torque_req"
X+y
=? "Vdc"
X+y
=? "pwm_status"

```

Fig. 6.26 - Set minimo di variabili da monitorare in debug

Occorre poi attivare l'aggiornamento automatico delle variabili. Se l'encoder viene letto correttamente, il riquadro contenente la variabile `theta_me` dovrebbe illuminarsi di giallo ed assumere un valore compreso tra 0 e 10π radianti elettrici. In caso contrario, Code Composer Studio blocca l'esecuzione del firmware per i motivi descritti nel paragrafo 6.4.

I LEDs della scheda LAUNCHXL-F28379D forniscono una rapida indicazione dello stato del microcontrollore: in assenza di errori, devono essere accesi solo LEDs verdi, in caso contrario anche quello rosso da solo (errore generico) o assieme a quello blu (errore relativo all'encoder). In seguito, si consiglia di verificare che le letture di

corrente di fase non abbiano offset di alcun tipo, ovvero “ia”, “ib” ed “ic” dovrebbero avere, mediamente, valore nullo o quasi. Eseguita poi la precarica dei condensatori di DC-link, si misura la tensione del DC-bus e si aggiorna la variabile “Vdc”. Questa operazione è necessaria per l’inverter di alpitronic ma non lo sarà con quello progettato in casa, in quanto il primo non è dotato di sensore di tensione.

È possibile, infine, far ruotare il motore, impostando prima un riferimento di velocità tramite la variabile speed_ref, in giri al minuto e poi il limite di coppia, tramite pos_torque_lim, in Newtonmetri. Il motore comincerà ad accelerare fino a raggiungere il riferimento, erogando una coppia massima pari a pos_torque_lim.

Durante le prove, se eseguite alimentando l’inverter tramite un alimentatore da banco, si consiglia di non frenare mai assegnando un valore nullo a speed_ref ed un valore negativo a neg_torque_lim poiché, così facendo, il motore si comporterebbe da generatore ma l’alimentatore stabilizzato non può restituire alla rete l’energia cinetica del gruppo ruota, in quanto è unidirezionale. Per questo motivo, per frenare è sufficiente porre a zero “pos_torque_lim”, disabilitando quindi le uscite PWM del microcontrollore e facendo rallentare il rotore tramite gli attriti nei cuscinetti e con l’aria.

Per stabilire la massima velocità raggiungibile dal motore, data una certa tensione DC, si può fare riferimento alla velocità a vuoto dello stesso. Si ricorda che essa corrisponde alla massima velocità raggiungibile dal motore senza deflussaggio, quando la FCEM eguaglia la tensione applicabile dall’inverter ai morsetti del motore:

$$\Omega_0 = \frac{V_{DC}}{\sqrt{3} \Lambda_{mg}} \text{ [rad. elettrici/s]}$$

Tale valore va poi convertito in giri al minuto. Impostata speed_ref ad un valore pari al 90% del valore così calcolato si è certi di lavorare nella regione a coppia costante. Come regola di massima, aumentando di 30 V la tensione DC si aggiungono circa 1000 rpm alla velocità massima raggiungibile.

Per ottenere tensioni lato continua e, quindi, anche velocità di rotazione sufficientemente elevate, è possibile mettere più alimentatori in serie, a patto che i rispettivi terminali di Ground (GND) siano isolati tra di loro.

Il verso di rotazione del motore può essere impostato agendo sulla macro SDR, presente in user.h. La macro può assumere valore 1 o -1, a seconda che il verso di rotazione desiderato sia orario o antiorario. Essa moltiplica le variabili date in ingresso (riferimento e reattorazione di velocità) e quella in uscita (richiesta di coppia) dal regolatore di velocità. Così facendo, una volta stabilito se il motore controllato è collocato sul lato sinistro o destro del veicolo, si imposta il verso di rotazione e la velocità potrà essere positiva soltanto. Il regolamento di Formula Student, infatti, vieta ai veicoli di muoversi in retromarcia, quindi con velocità negative.

Infine, per passare dal controllo di velocità a quello di coppia, è sufficiente cancellare SPEED_CONTROL dai Predefined Symbols. In tal modo le righe di codice relative al regolatore di velocità vengono escluse dalla compilazione e si può imporre direttamente da debug una richiesta di coppia, in Newtonmetri, scrivendo nella

variabile torque_req. Occorre prestare attenzione poiché la velocità massima del motore, senza regolatore, è limitata soltanto dalla coppia resistente. Con motore a vuoto, ciò potrebbe essere pericoloso in quanto gli attriti dovuti ai cuscinetti ed alla rotazione sono molto bassi.

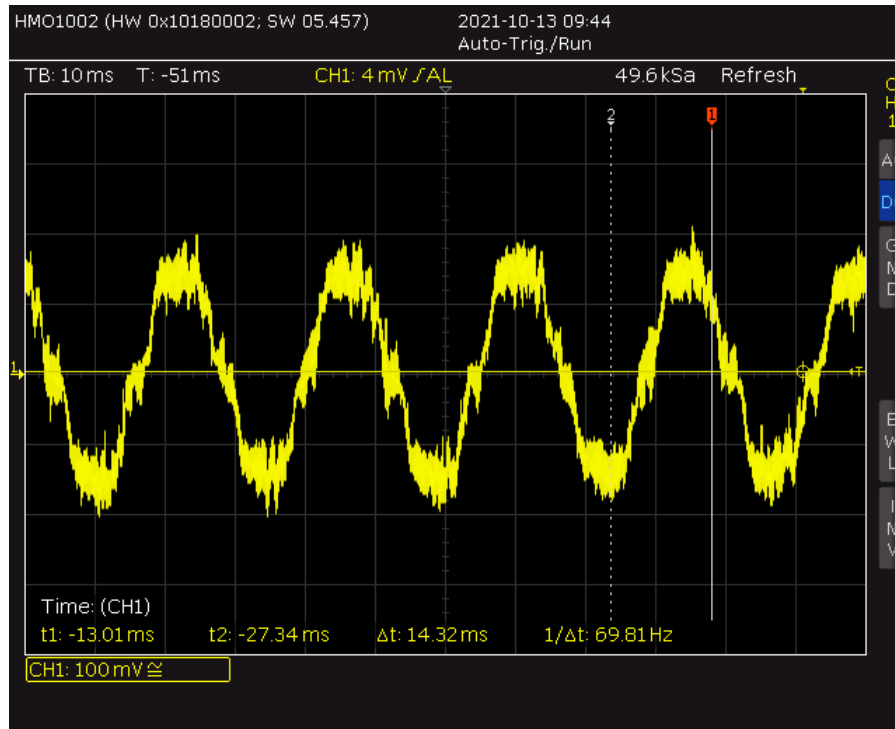


Fig. 6.27 – Corrente di fase misurata con pinza amperometrica avente sensibilità 100 mV/A e visualizzata tramite oscilloscopio

L'algoritmo di controllo è stato testato alimentando l'invertitore di alpitronic con una tensione massima di 150 V DC, ottenuti tramite un raddrizzatore a diodi in cascata ad un trasformatore monofase collegato alla rete elettrica di bassa tensione. Il motore, a vuoto, ha raggiunto correttamente il riferimento, posto pari a 5000 rpm.

7 Lavori futuri

L'invertitore qui descritto è quasi interamente realizzato in casa dalla Divisione Elettrica. Ne consegue che saranno necessarie numerose prove per verificare che tutto il sistema funzioni correttamente e, soprattutto, sia sicuro.

Tutto il codice elaborato è stato testato in modalità di debug caricandolo nella memoria RAM del microcontrollore. Conseguentemente, una volta tolta l'alimentazione, tutto il codice viene perso. Per poter avere un invertitore funzionante in autonomia, cioè senza un computer ad esso collegato, occorre caricare le istruzioni nella memoria FLASH [5]. Potrebbe essere necessario modificare alcune parti di codice per questioni legate alla memoria disponibile all'interno del microcontrollore, tuttavia ciò non è stato verificato per motivi di tempo.

La modalità di debug è disponibile in entrambi i casi, però si consiglia di testare estensivamente tutto il codice prima di adattarlo alla memoria FLASH.

Una modifica che necessariamente andrà fatta sarà l'implementazione della comunicazione tramite protocollo CAN, in modo che l'invertitore possa ricevere dalla centralina della vettura elettrica l'abilitazione, il riferimento di velocità, i limiti di coppia e qualsiasi altra informazione si voglia. Chiaramente, è possibile e vantaggioso anche l'invio di dati da parte dell'invertitore, quali le temperature degli IGBTs e del motore, la coppia erogata ed altre informazioni.

Come si è visto nel paragrafo 5.2.2, le prove a banco sul motore non sono ancora complete. Per ottenere un funzionamento dell'azionamento alla massima efficienza, occorre tracciare tutto il luogo dei punti a massima coppia. Inoltre, vanno effettuate delle prove nella regione di deflussaggio, trattandosi di un funzionamento particolarmente delicato.

Infine, implementando un algoritmo per la stima della posizione rototorica, si otterrebbe un funzionamento che non necessita del sensore di posizione (cosiddetto "sensorless"). Ciò consentirebbe di rimuovere l'interfaccia encoder, riducendo complessità, costi ed ingombri del sistema. Si noti, nel paragrafo 6.5, come la lettura del sensore di posizione costituisca, all'interno del periodo del controllo T_s , l'operazione di durata maggiore. Se si volesse aumentare la frequenza di esecuzione del controllo, pertanto, essa potrebbe divenire il fattore limitante. Si ricorda inoltre che l'encoder viene montato sul motore dal costruttore dello stesso e pertanto, se si volesse cambiare motore, probabilmente andrebbero modificate sia l'interfaccia fisica (TIDM-1008) che quella software tra microcontrollore e sensore. Si potrebbe valutare la possibilità di sfruttare una soluzione commerciale, invece dello sviluppo in proprio degli algoritmi, ad esempio lo stimatore FAST™ (Flux Angle Torque Speed estimator) disponibile nella memoria di alcuni microcontrollori commercializzati dalla Texas Instruments.

8 Conclusioni

Questo lavoro di tesi nasce dalla decisione, da parte della squadra di Formula Student dell'Università di Padova, di sostituire gli invertitori attualmente utilizzati a bordo della monoposto elettrica.

A partire dalla descrizione degli invertitori attualmente montati nel veicolo, si descrivono brevemente i componenti e le risorse hardware scelte per la realizzazione del prototipo, in particolare il microcontrollore che sovrintende tutte le operazioni. Successivamente, vengono sviluppate prima le routine di comunicazione tra microcontrollore e gate drivers e, poi, gli algoritmi di controllo per motori brushless, simulati su Simulink e poi tradotti in codice. Tutte le fasi di progettazione sono seguite dal collaudo.

Come risultato, questa tesi è un pratico manuale che spiega tutte le principali scelte effettuate in fase di progettazione. Essa è affiancata da due modelli Simulink, utilizzati per la parte di controllo motor e rappresentati nell'Appendice D ed un programma, `inverter_v1.11.8`, che può essere caricato nel microcontrollore tramite Code Composer Studio.

`inverter_v1.11.8` è ottenuto dall'unione di altri due programmi: `inverter_v1.11.7` e `gateDrivers`. Il primo contiene solo le routine di controllo del motore, il secondo invece è stato scritto appositamente per sviluppare la comunicazione SPI.

Gli algoritmi di controllo sono stati validati tramite un inverter commerciale programmabile, mentre la comunicazione SPI è stata validata con l'hardware che andrà a costituire il prototipo di invertitore per la squadra di Formula Student, rappresentato in figura 8.1. Al momento della stesura di questo lavoro di tesi, il prototipo è ancora in fase di costruzione. Non è stato pertanto possibile verificare tutte le funzionalità del programma `inverter_v1.11.8`.

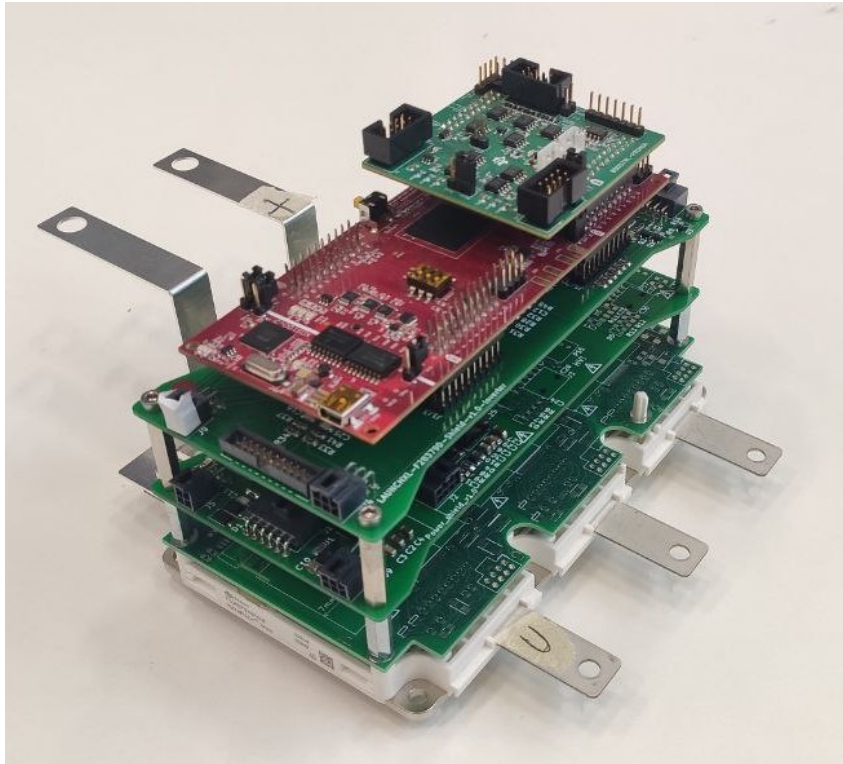


Fig. 8.1 - Prototipo di invertitore assemblato. Dall'alto verso il basso: interfaccia per lettura encoder, LAUNCHXL-F28379D, scheda di condizionamento segnali, scheda di alimentazione, scheda con gate drivers e modulo IGBT

Nelle varie appendici si riportano le istruzioni per modificare rapidamente il codice qualora si volesse testare senza encoder (Appendice B) o qualora si volesse modificare la modulazione vettoriale (Appendice C).

L'auspicio dell'autore è che un giorno la vettura elettrica possa montare degli invertitori interamente progettati dai membri della squadra. Se anche ciò non dovesse succedere, questa tesi può costituire un utile manuale e base di partenza per la formazione dei membri della squadra nell'ambito del controllo dei motori elettrici. La Formula Student è, infatti, prima di tutto un'attività a carattere didattico.

Ringraziamenti

Desidero ringraziare tutti quanti abbiano contribuito alla riuscita di questo progetto di tesi.

Innanzitutto un doveroso grazie va ai miei relatori, per avermi consentito di lavorare a questo progetto che si è trasformato, per me, in una passione.

All'Ing. Parrino ed a Infineon Technologies AG, per la fornitura di componenti e per il prezioso supporto fornito.

A Philipp Niedermayr e ad alpitronic SRL, per l'invertitore di prova e tutti i consigli utili.

Un grazie anche al prof. Meneghetti, da sempre faculty advisor del progetto Formula Student presso l'Università degli Studi di Padova.

Questo lavoro, tuttavia, non sarebbe mai stato possibile senza il sostegno delle persone che mi circondano ogni giorno.

Ad Anna, sono mille i motivi per cui ti dovrei ringraziare, li riassumo tutti dicendoti grazie per l'infinita pazienza nel sopportare le mie giornate passate a lavorare sull'inverter e per i momenti di distrazioni che mi concedi.

Senza la mia famiglia, non avrei avuto le forze di fermarmi tre anni e mezzo nella Divisione Elettrica del RaceUP Team.

Agli amici stretti, Andrea, Luca ed Antonio, tra poco non vi dovrete più chiedere cos'ho di tanto importante da fare il sabato mattina da spingermi a scappare presto la sera prima.

A Giovanni e Mattia, senza di voi la realizzazione del prototipo e, quindi, di metà del mio lavoro di tesi non sarebbe mai stata possibile.

Ringrazio infine tutto il RaceUP Team, Divisione Elettrica, i cui partecipanti ho avuto il piacere di conoscere a partire dalla stagione 2018/19, ma anche a tutti gli studenti degli anni precedenti che hanno contribuito allo sviluppo di questo fantastico progetto.

Grazie anche a voi, Riccardo e Francesco, che non vi siete spaventati di fronte alle difficoltà iniziali e che, mi auguro, possiate scoprire nella progettazione dell'inverter la stessa passione che ho trovato io.

Appendice A

La configurazione hardware, ovvero l'insieme delle connessioni dei connettori da JP1 a JP6, della scheda LAUNCHXL-F28379D varia nelle due principali situazioni di test. Quando si eseguono dei test in debug con la scheda scollegata dal prototipo di invertitore di figura 8.1, l'alimentazione viene fornita al microcontrollore dal computer ad esso collegato, pertanto solo i connettori JP1, JP2, JP3, JP4 vanno cortocircuitati. Quando la scheda è montata sull'inverter, invece, gli unici connettori da cortocircuitare sono JP4, JP5, JP6, anche se si esegue il debug a computer. In tal modo il microcontrollore risulta essere galvanicamente isolato dal computer [5].

Appendice B

Può accadere che sia necessario testare il firmware pur non disponendo di un motore AMK e, quindi, dell'encoder ECI 1118. Il programma è scritto in modo da bloccarsi qualora non venga rilevata la presenza del sensore di posizione. Si riportano di seguito alcune modifiche al codice per far funzionare la routine di controllo anche senza encoder.

- commentare **setupEncoder** nel **main.c** e **readEncoder** in **epwm_management.c**
- nel **main.c** spostare **setupEPWM** sopra a **setupTimer**
- nel **main.c**, sotto a **Device_initGPIO** incollare:

```
Interrupt_initModule();  
Interrupt_initVectorTable();  
EINT;  
ERTM;
```

- in **epwm_management.c**, sotto a

```
// Enable Global Interrupt (INTM) and realtime interrupt (DBGM)
```

commentare:

```
EINT;  
ERTM;
```

Appendice C

Questo appendice riporta le Istruzioni per passare rapidamente da Space Vector Modulation simmetrica ad asimmetrica, e viceversa, nel firmware del microcontrollore.

Immaginando che il tipo di modulazione preimpostato sia di tipo simmetrico e che si voglia invece ottenere una frequenza di controllo doppia rispetto a quella di commutazione (passando quindi alla modulazione asimmetrica) occorre apportare le seguenti modifiche al codice:

- 1) nel file `user.h`, impostare il valore desiderato della frequenza di commutazione modificando le macro `FSW` (in hertz) e `EPWM_TIMER_TBPRD` (come spiegato nel paragrafo 6.1)
- 2) impostare il modulo `ePWM1` in modo che faccia scattare l'interrupt service routine sia ad inizio che a metà del periodo di commutazione
- 3) lo stesso modulo `ePWM1` deve triggerare le conversioni ADC sia ad inizio che a metà del periodo di commutazione
- 4) la funzione di shadowing dei registri che governano la PWM deve consentire l'aggiornamento dei duty-cycles sia ad inizio che a metà del periodo di commutazione
- 5) il periodo di campionamento, utilizzato nei regolatori PI e nel calcolo della velocità di rotazione del motore, deve essere quello corrispondente alla frequenza del controllo e non a quella di commutazione

I punti 2), 3), 4) e 5) si risolvono modificando i Predefined Symbols del microcontrollore. Aprendo il codice in Code Composer Studio, bisogna cliccare col tasto destro sul nome del progetto > properties > Build > C2000 Compiler > Predefined Symbols. Nell'elenco deve comparire solo uno tra `SVM_ASIMMETRICA` o `SVM_SIMMETRICA`, a seconda del tipo di modulazione desiderato. La modifica va fatta sia per `CPU1_RAM` che per `CPU1_FLASH`. In tal modo, il compilatore esclude determinate parti di codice (evidenziate in grigio) dalla compilazione.

Con queste impostazioni, la frequenza del controllo risulterà essere doppia rispetto a quella di commutazione, permettendo di fatto di operare con Space Vector Modulation asimmetrica.

Appendice D

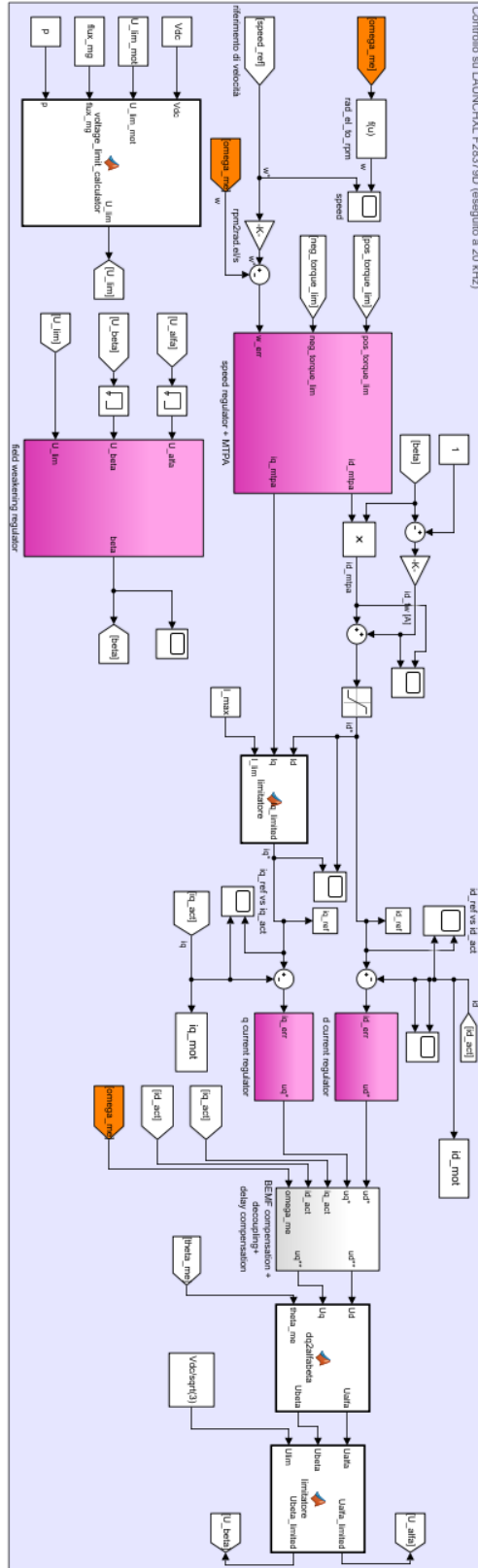


Fig. D.1 - File Simulink "Control.slx". Vista nel dettaglio dell'algoritmo di controllo

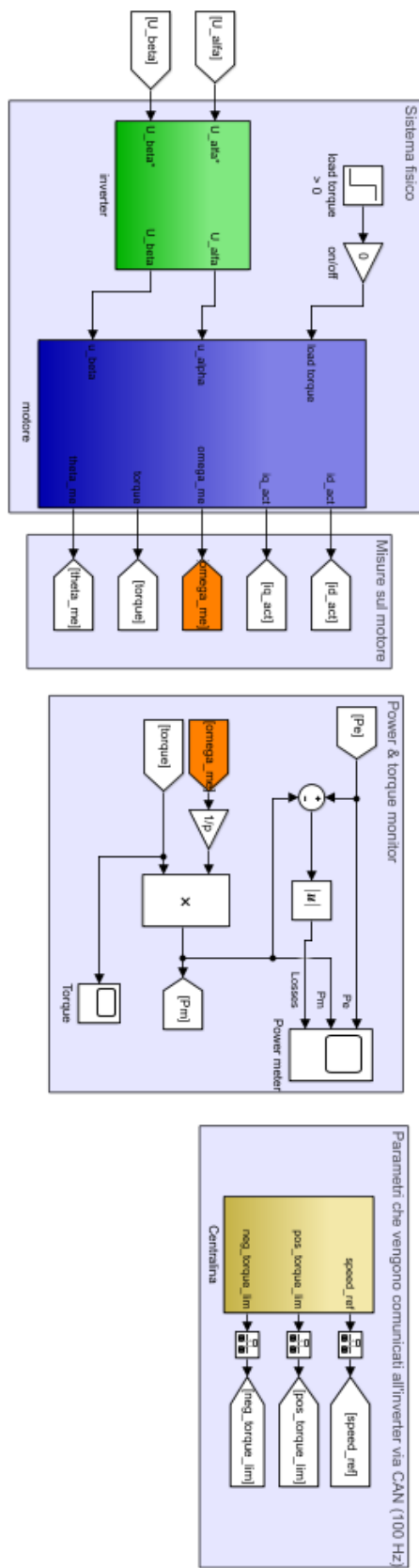


Fig. D.2 - File Simulink "Control.slx". Vista nel dettaglio dei modelli di inverter e motore, della centralina e di altri sottosistemi per il calcolo del rendimento del motore e della coppia erogata

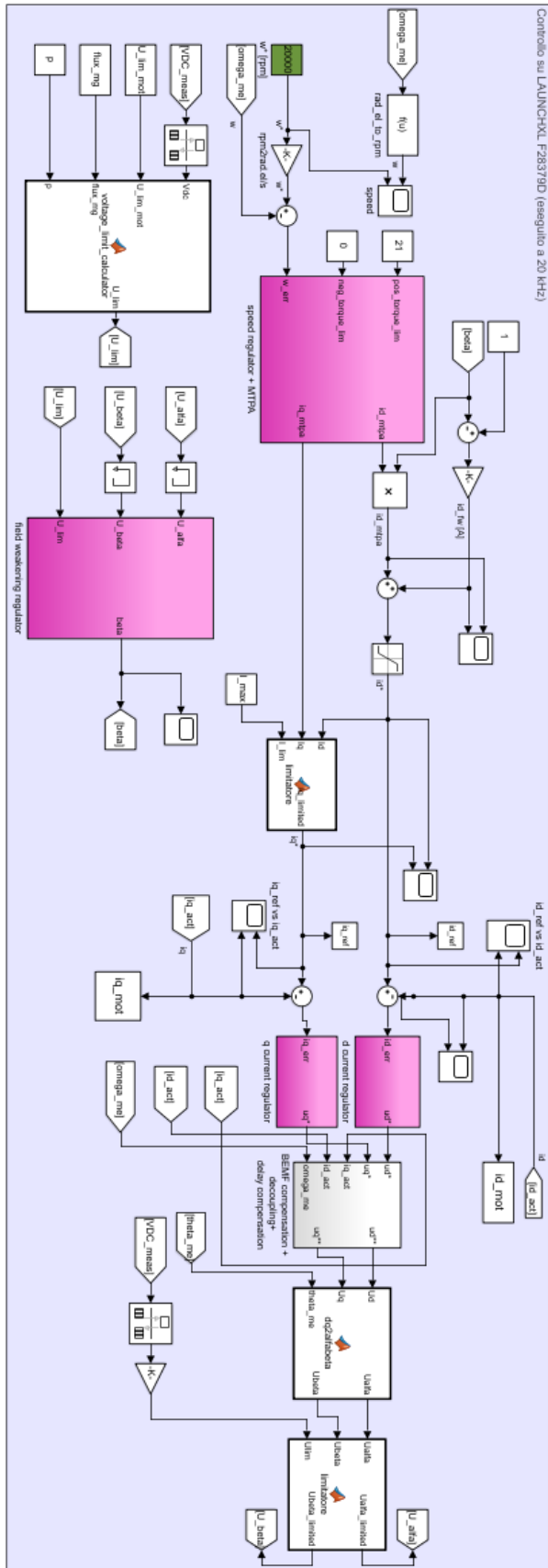


Fig. D.3 - File Simulink "Power_stage.slx". Vista nel dettaglio dell'algoritmo di controllo

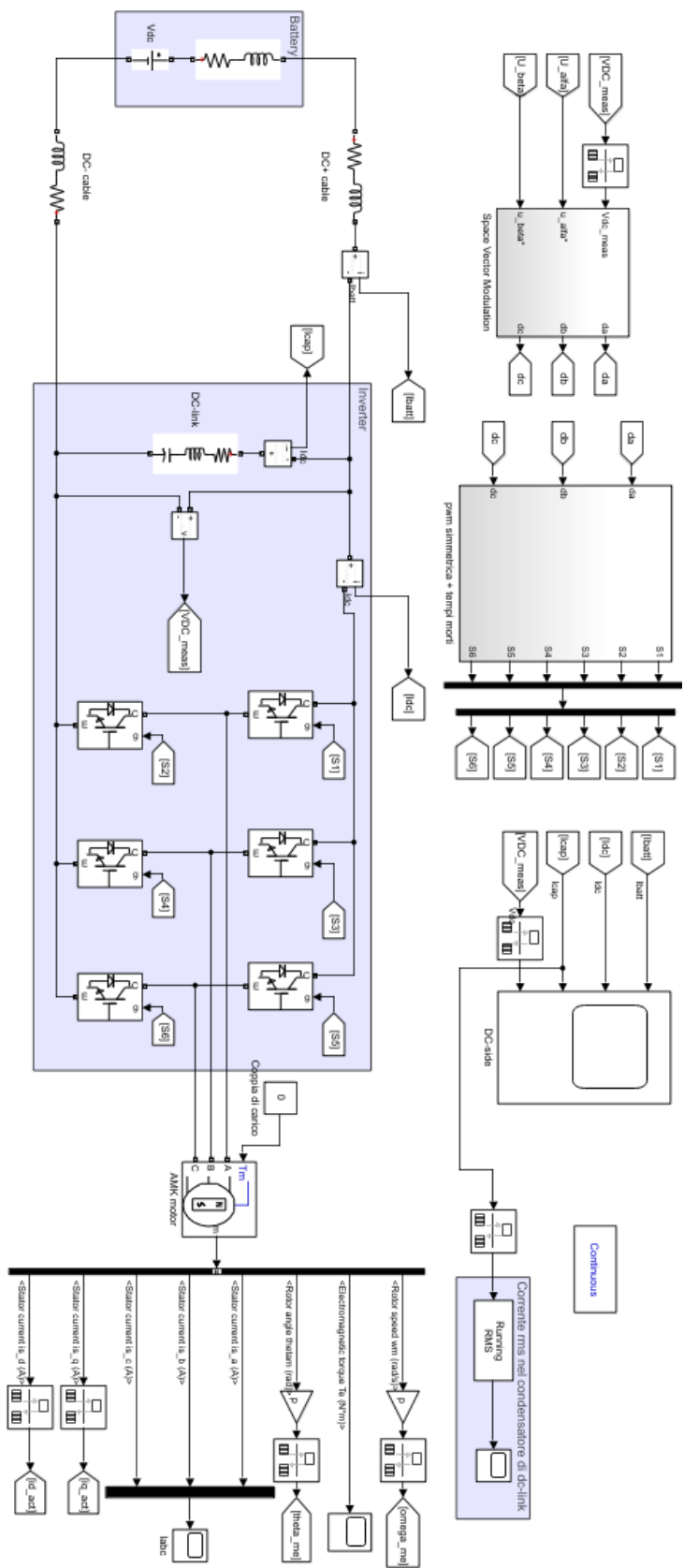


Fig. D.4 - File Simulink "Power_stage.slx". Vista nel dettaglio della modulazione vettoriale e dei modelli di invertitore e motore

BIBLIOGRAFIA

- [1] Virginia Manzolini, "Advanced Algorithms for Flux-Weakening and Sensorless Control of Interior PM and Reluctance Synchronous Motor Drives", 30 settembre 2018
- [2] Bon-Ho Bae, Seung-Ki Sul, "A Compensation Method for Time Delay of Full-Digital Synchronous Frame Current Regulator of PWM AC Drives", *IEEE TRANSACTIONS ON INDUSTRY APPLICATIONS*, vol. 39, n. 3, maggio/giugno 2003
- [3] Corso di "TECNOLOGIE PER IL CONTROLLO DI CONVERTITORI E AZIONAMENTI ELETTRICI CON LABORATORIO 2020-2021 - INP9087259"
<https://elearning.unipd.it/dii/course/view.php?id=2358>, febbraio 2022
- [4] M. Vujacic, M. Hammami, M. Srndovic, G. Grandi, "Analysis of dc-Link Voltage Switching Ripple in Three-Phase PWM Inverters", *Energies*. **2018**, 11, 471
- [5] A. Fano, "Guida all'utilizzo del LAUNCHXL-F28379D Development Kit per i membri del Race Up Team, Electric Division", ultima revisione: 8 Maggio 2021
- [6] L. Peretti, M. Zigliotto, "Azionamenti Elettrici Industriali – A.A. 2012-2013", 21 marzo 2013, pp. 1-13
- [7] "E2E™ design support forums"
<https://e2e.ti.com/support/microcontrollers/c2000-microcontrollers-group/c2000/f/c2000-microcontrollers-forum/925637/ccs-launchxl-f28379d-endat22-library-on-a-c2000ware-project/3420106#3420106>, febbraio 2022
- [8] G. Filippi, "Progettazione di un inverter trifase per la vettura elettrica formula SAE", 21 febbraio 2022
- [9] "EiceDRIVER™ SIL High Voltage IGBT Driver for Automotive Applications 1EDI2004AS Single Channel Isolated Driver for Inverter Systems AA Step", Rev. 2.0, 16 gennaio 2019, Infineon Technologies AG
- [10] "Application Note for EICEDRIVER™ 1EDI2004AS", Revision 1.0, 1 novembre 2019, Infineon Technologies AG
- [11] "AN2010-09_HYBRIDPACK-DSINFO HybridPACK™ Automotive Power Modules Explanation of Technical Information", Revision 2.1b, 7 gennaio 2010, Infineon Technologies AG
- [12] "AN2007-04 How to calculate and minimize the dead time requirement for IGBTs properly", v1.0, maggio 2007, p.6, Infineon Technologies AG
- [13] "HybridPACK™ Drive Module FS380R12A6T4B Final Data Sheet", v3.1, 10 settembre 2019, Infineon Technologies AG

- [14] "Current Transducers HO-S series Ref: HO 50-S, HO 100-S, HO 150-S, HO 200-S, HO 240-S, HO 250-S", version 6, 15 luglio 2020, LEM International SA
- [15] "LAUNCHXL-F28379D Overview - SPRUI77C", agosto 2016, revisionato a marzo 2019, Texas Instruments Incorporated
- [16] "TMS320F2837xD Dual-Core Microcontrollers - SPRUHM8I. Technical Reference Manual", dicembre 2013, revisionato a settembre 2019
- [17] "Texas Instruments Incorporated"
https://software-dl.ti.com/ccs/esd/documents/sdto_cgt_floating_point_optimization.html#--fp_mode, ultimo aggiornamento 30 giugno 2020
- [18] "Technical Explanation eMPack A4 Application Kit", revisione 4, 5 febbraio 2021, Semikron
- [19] Silverio Bolognani, "Appunti di Azionamenti Elettrici - APPENDICE A - Vettori spaziali per lo studio dei sistemi trifase", 11 ottobre 2015
- [20] Silverio Bolognani, "CAPITOLO 5 - Motore sincrono a magneti permanenti (brushless sinusoidale)", 17 maggio 2008
- [21] "PDK_205481_KW26-S5-FSE-4Q", versione 2017/33, AMK Arnold Müller GmbH & Co. KG
- [22] A. Cenedese, A. Beghi, "Introduction to Digital Control"
- [23] Silverio Bolognani, "CAPITOLO 6 – Azionamento con motore sincrono a magneti permanenti", 22 novembre 2015
- [24] Silverio Bolognani, "CAPITOLO 4 – Invertitore trifase in onda quadra e a PWM", 19 febbraio 2008
- [25] Appunti del corso di "CONVERSIONE STATICA DELL'ENERGIA ELETTRICA 2020-2021 - INL1001405",
<https://elearning.unipd.it/dii/course/view.php?id=2304>, febbraio 2022
- [26] Y. Changzhou, L. Chun, W. Qionglong, Z. Weitang, L. Sicong and Z. Xing, "Implementation of multi-sampling current control for grid-connected inverters using TI TMS320F28377x," *2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2017, pp. 1228-1233, doi: 10.1109/YAC.2017.7967600.
- [27] "Electronic Concepts, Inc.",
<https://www.ecicaps.com/tech-tools/technical-papers/low-inductance-dc-bus-capacitor-high-power-traction-motor-drive-inverters/>, febbraio 2022
- [28] S. Bolognani, "Azionamenti Elettrici - Lezioni a.a. 2020-2021 - prof. Silverio Bolognani – Parte II – Invertitore trifase di tensione", 14 ottobre 2020

[29] “EnDat 2.2 Absolute Encoder Master Interface Reference - Design for C2000™ MCUs – TIDUDE3A”, settembre 2017, revisionato in aprile 2018, Texas Instruments Incorporated

[30] “Formula Student Rules 2020”, versione 1.0,
<https://www.formulastudent.de/fsg/rules/>, febbraio 2022

[31] “Analog Devices, Inc”,
<https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html>, febbraio 2022

[32] “STMicroelectronics”
https://www.st.com/resource/en/application_note/dm00122161-igbt-datasheet-tutorial-stmicroelectronics.pdf, febbraio 2022

[33] “Texas Instruments Incorporated”
<https://www.ti.com/store/ti/en/p/product/?p=LAUNCHXL-F28379D>, febbraio 2022

[34] “Circuit Digest”
<https://circuitdigest.com/article/introduction-to-bit-banging-spi-communication-in-arduino-via-bit-banging>, febbraio 2022

[35] “AMDS4”
<https://blog.amds4.com.br/new-lems-smd-current-transducer-hmsr/>, febbraio 2022