UNIVERSITÀ DEGLI STUDI DI PADOVA

FACOLTÀ DI INGEGNERIA

Corso di Laurea Triennale in
Ingegneria Meccanica e Meccatronica

Tesi di Laurea

Algoritmi di controllo di un veicolo autonomo a retroazione visiva

Laureando

Kevin A. Calderon Regalado

611542 - IMM

Relatore

Prof. Roberto Oboe

ANNO ACCADEMICO 2012 – 2013

Sommario

Con il seguente elaborato si vuole offrire uno strumento di ulteriore approfondimento sul controllo in retroazione di una smart car, con la speranza risulti utile ai futuri concorrenti della “Freescale Cup”. Il Sistema in esame è il classico sistema meccatronico dove sono richieste competenze nell’ambito elettromeccanico, elettronico ed in particolare una notevole dimestichezza con i linguaggi di programmazione sia di alto che basso livello. Dopo una visione di insieme sull’Hardware e Software utilizzati e sul loro interfacciamento, mi soffermerò in particolare sull’analisi e sulla elaborazione dei segnali acquisiti dalla telecamera ed infine sull’implementazione del controllo del veicolo. Con questo documento si vuole offrire al lettore un punto di partenza per la realizzazione di un sistema di controllo innovativo contenente un algoritmo di controllo avanzato capace di sfruttare al massimo le potenzialità del “Kit Freescale Cup”.

Indice

Capitolo 1 – Introduzione

Capitolo 2 – Scheda di controllo principale

Capitolo 3 – Scheda di potenza

Capitolo 4 – Motore in corrente continua

Capitolo 5 – Servo comando

Capitolo 6 – Line Scan Camera

Capitolo 7– Il controllo in retroazione

CAPITOLO 1

Introduzione

1.1. Freescale Cup

Il seguente elaborato nasce nell'ambito della competizione *Freescale Cup*, essa permette alle università di tutto il mondo di fronteggiarsi in una sfida che coinvolge gruppi di studenti dell'ambito meccatronico. La sfida contrappone diversi gruppi di studenti che sono chiamati ad assemblare, programmare e testare un veicolo autonomo in grado di seguire una traiettoria costituita da una linea nera su una pista bianca.
L'Università di Padova, e nello specifico il Corso di Laurea Triennale in Ingegneria Meccanica e Meccatronica ha aderito a questo progetto per la prima volta nell'a.a. 2012/2013, ma la storia della competizione risale a molti anni prima, obiettivo del paragrafo seguente è quello di contestualizzare la gara fornendo una rapida escursione delle tappe più importanti che hanno portato alla competizione odierna.
La storia di "Freescale Cup" ebbe inizio nell'anno 2003 presso la Hanyang University in Corea, sotto ilvnome di "Smart Car Race", a quel tempo la competizione coinvolse circa ottanta gruppi di studenti, decretando così il principio di una lunga serie di sfide a cui, stagione dopo stagione, hanno aderito paesi come Cina, India, Malesia, America Latina, Nord America e, solo recentemente, Europa. L'evoluzione fu rapida, riuscendo a coinvolgere più di 500 università e fino a 1500 studenti da tutto il mondo ogni anno. Dall'anno 2011 anche l'Europa, o più in generale l'EMEA (*Europe, Middle East and Africa*) ha aderito al progetto, con studenti provenienti da Germania, Francia, Romania e Repubblica Ceca. Con un incremento così elevato dei paesi che ospitano *Freescale Cup* si è resa necessaria l'istituzione di più livelli di competizione, partendo da sfide locali fino ad arrivare alla finale internazionale. Nell'autunno dell'anno 2012 *Freescale* ha aperto una nuova edizione della sfida, lanciando il "Freescale Cup Kit" che permetterà a qualunque gruppo di studenti di realizzare il proprio veicolo. Solo in Europa, nella stagione 2012/2013, 33 università si sono unite a questo progetto, con più di 300 studenti al lavoro sui propri veicoli con l'obbiettivo di renderli più veloci, intelligenti e performanti di quelli avversari.

1.2. L’obiettivo

La squadra vincitrice sarà quella in grado di completare il circuito nel minore tempo rispettando le regole di gara. Le finalità didattiche della competizione sono quelle di offrire agli studenti l’opportunità di comprendere a fondo il funzionamento dei microcontrollori, delle diverse applicazioni dell’elettronica a problemi di questo tipo e infine presentare l’opportunità di uno scambio reciproco con i propri colleghi stranieri impegnati a risolvere i medesimi problemi. La realizzazione del veicolo richiede conoscenze nell’ambito della programmazione di sistemi embedded, basi di elettronica, teoria dei circuiti digitali, fondamenti di macchine e azionamenti elettrici, misure per l’automazione e nozioni di controlli automatici. *Freescale Cup* risulta dunque una sfida tra studenti non laureati di tutto il mondo che mette in competizione team diversi ma è anche una preziosa opportunità per collaborare, imparare e crescere attraverso lo scambio reciproco di idee ed esperienze.

1.3. Il “Kit Freescale Cup’’

Il kit fornito dalla Freescale comprende:

- Telaio in materiale plastico completo di ruote e sospensioni
- Batteria di alimentazione
- Due motori DC “standart motor rn260c” per la trazione dei veicolo
- Servo comando Futaba S3010
- Scheda di controllo TRK-MPC5604B
- Motor drive per alimentazione motori e comando sensori e attuatori
- Line Scan Camera TSL1401-DB
- Software dedicato: CodeWarrior

CAPITOLO 2

Scheda di controllo principale

2.1. Microcontrollore: Qorivva TRK-MPC5604B

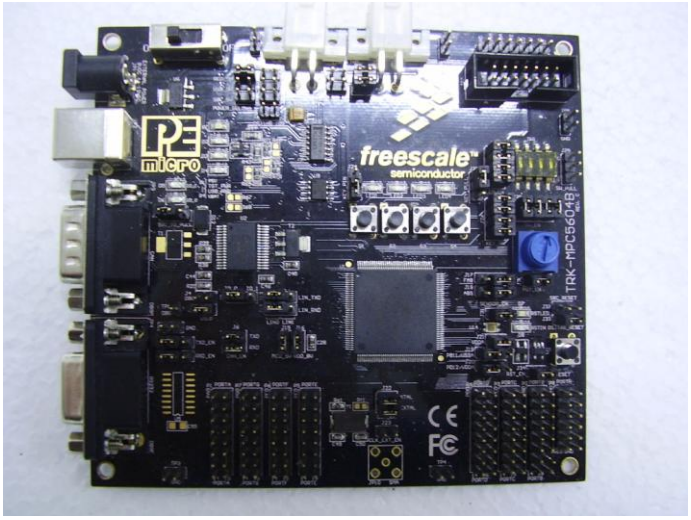


Fig.2.1.1 – Microcontrollore MPC5604B

2.1.1. Introduzione

Il Qorivva MPC5604B rappresenta una nuova generazione di microcontrollori a 32 bit basati su Power Architecture®, esso appartiene a una famiglia più ampia di prodotti ideati per applicazioni automobilistiche, mirati ad affrontare e gestire il numero sempre in crescita di applicazioni elettroniche a bordo del veicolo. Le caratteristiche del chip a bordo del dispositivo sono le seguenti:

Core
- PowerPC e200z0 core running 48-64MHz
- VLE ISA instruction set for superior code density
- Vectored interrupt controller
- Memory Protection Unit with 8 regions, 32byte granularity

Memory
- 512Kbyte embedded program Flash, 64KByte data flash
- 64Kbyte embedded data Flash (for EE Emulation)
- Up to 64MHz non-sequential access with 2WS
- ECC-enabled array with error detect/correct
- 48Kbyte SRAM (single cycle access, ECC-enabled)

Communications
- 3x enhanced FlexCAN___
- 64 Message Buffers each, full CAN 2.0 spec
- 4x LINFlex
- 3x DSPI, 8-16 bits wide & chip selects
- 1x I^2C

Analog

- 5V ADC 10-bit resolution

Timed I/O
- 16-bit eMIOS module

Other
- CTU (Cross Triggering Unit) to sync ADC with PWM Channels
- Debug: Nexus 2+___
- I/O: 5V I/O, high flexibility with selecting GPIO functionality
- Packages: 100LQFP, 144LQFP, 208MAPBGA (Development only)
- Boot Assist Module for production and bench programming

2.1.2. Schema funzionale

Lo schema a blocchi del microcontrollore e bus di comunicazione “Crossbar Switch” sono visibili in Fig.2.1.2, in particolare si notano il processore e i blocchi ausiliari che comprendono la generazione dei clock di sistema e la gestione degli Interrupt e Debug, le memorie di massa di tipo flash e la memoria centrale. In basso sono infine rappresentati i blocchi di I/0, i moduli eMIOS e il modulo ADC (*analog-to-digital converter*).

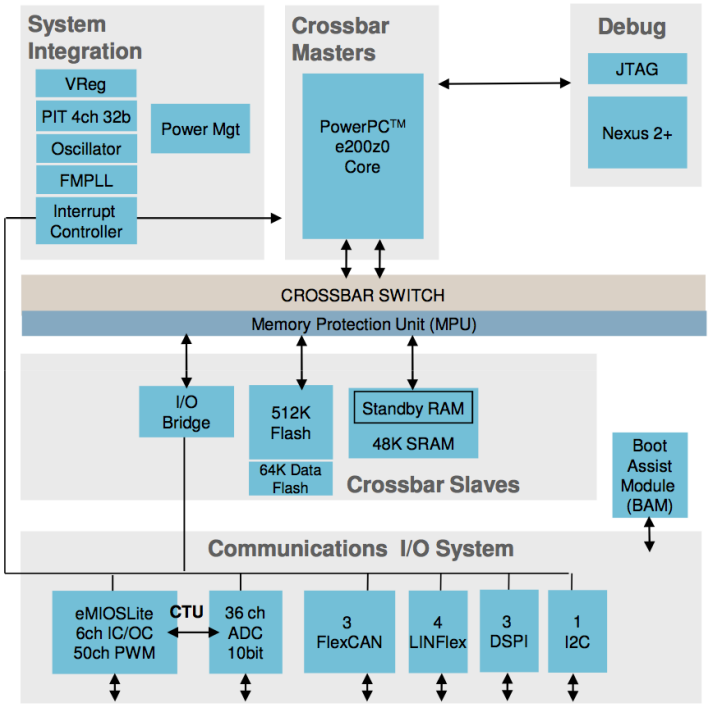


Fig.2.1.2 – Diagramma a blocchi generale della MPC5604B

2.1.3. Clock di sistema

I principali clock di sistema provengono da tre sorgenti:

- Fast external oscillator 4-16 MHz (FXOSC)
- Fast internal RC oscillator 16 MHz (FIRC)
- Frequency modulated phase locked loop (FMPLL)

Il microcontrollore dispone inoltre di ulteriori due oscillatori a bassa potenza:

- Slow internal RC oscillator 128 kHz (SIRC)
- Slow external crystal oscillator 32 kHz (SXOSC)

2.1.4. Architettura dei moduli eMIOS

Il Modulo di generazione dei clock (*CGM)* che genera il clock principale lavora secondo l'architettura mostrata in Fig.2.2.1.

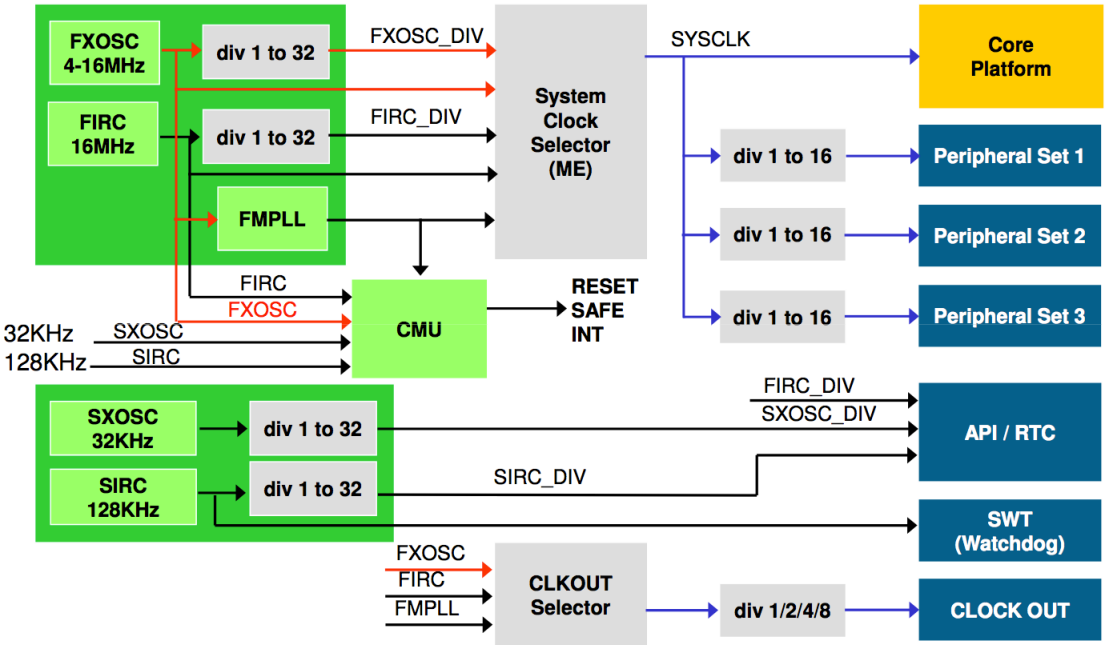


Fig.2.1.3 – Generazione dei clock di sistema

In particolare questo componente agisce attraverso un multiplexer di ingresso che seleziona il clock sorgente, questo può venire poi diviso o moltiplicato e inviato alle periferiche come System Clock (SYSCLK) come si può notare dalla Fig.2.1.4.

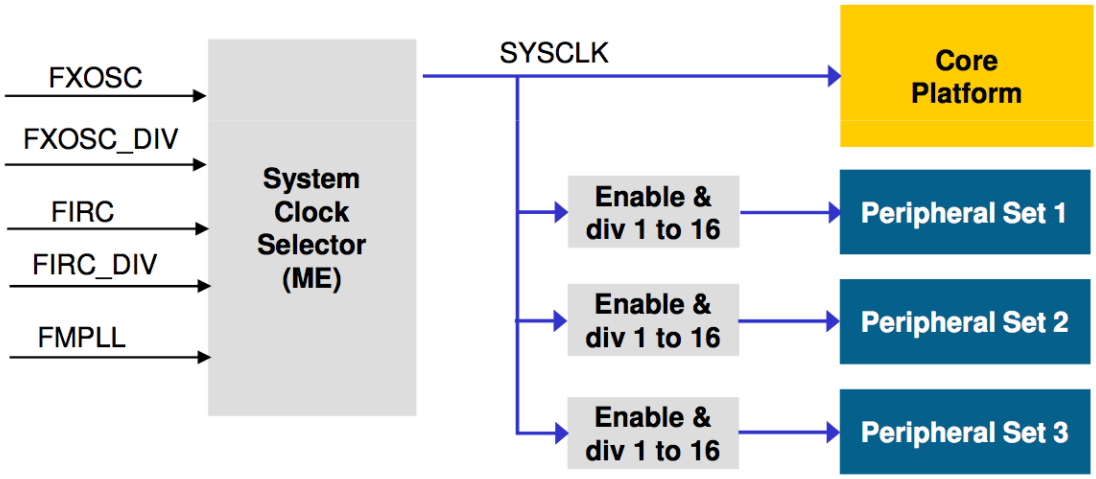


Fig.2.1.4 – CGM System clock

2.1.5. Architettura dei moduli eMIOS

I moduli eMIOS offrono molte funzionalità specifiche per generare o misurare eventi temporali. Ogni modulo offre una combinazione di funzionalità di tipo Pulse Width Modulation (PWM) come uscite, ingressi e funzioni di comparazione. Il controllore ha implementato diversi tipi di canali, inoltre le funzionalità variano a seconda del modulo eMIOS che si sta considerando. Ogni canale al suo interno dispone di un proprio internal counter che garantisce una risoluzione di 16 bit, per permettere la sincronizzazione tra canali diversi si dispone di un certo numero di counter buses che possono essere utilizzati per avere un riferimento temporale comune. I counter buses possono essere utilizzati assieme ai contatori individuali dei canali per implementare funzionalità avanzate. Il modulo eMIOS riceve in ingresso i SYSCLK a 64 MHz, assieme alle altre periferiche appartenenti al Peripheral Set 3 (eMIOS, ADC, CTUL).

Specifiche dei moduli eMIOS:

- 2 blocchi eMIOS da 28 canali ciascuno, che si possono osservare in Fig 2.1.5.
- 50 canali dispongono di funzionalità di tipo Output PWM Triggered (OPWMT)
- 6 canali con funzioni esclusivamente di tipo IC/OC
- Un Prescaler globale
- Registri a 16 bit
- 10 counter buses a 16 bit
- I contatori B,C,D,E possono essere comandati solo dai Unified Channel 0, 8, 16 e 24 rispettivamente
- Il contatore A può essere comandato solo dal Canale 23

I Canali eMIOS possono essere configurati via software per operare in uno dei seguenti modi:

- Single Action Input Capture (SAIC)
- Single Action Output Compare (SAOC)
- Input Pulse Width Measurement (IPWM)
- Input Period Measurement (IPM)
- Double Action Output Compare (DAOC)
- Modulus Counter (MC)
- Modulus Counter Buffered (MCB)
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB)
- Output Pulse Width Modulation Buffered (OPWMB)
- Output Pulse Width Modulation with Trigger (OPWMT)
- Center Aligned Output Pulse Width Modulation Buffered (OPWMCB)

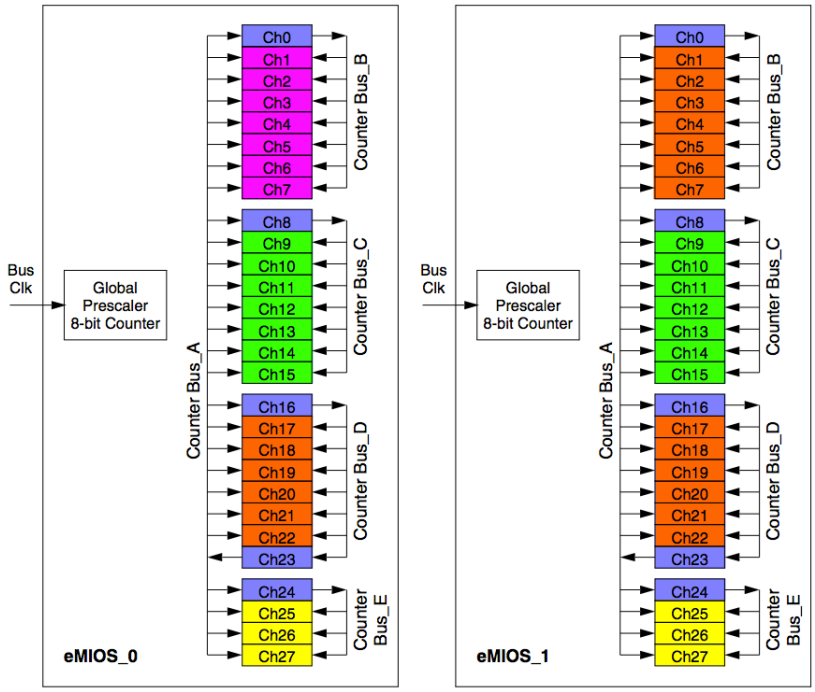


Fig.2.1.5. - Canali dei blocchi eMIOS

I canali eMIOS possono essere utilizzati con diverse modalità di funzionamento, come si può osservare in figura 2.1.6.

MODE[0:6]	Mode of operation
000_0000	General Purpose Input/Output (input)
000_0001	General Purpose Input/Output (output)
000_0010	Single Action Input Capture
000_0011	Single Action Output Compare
000_0100	Input Pulse width Measurement
000_0101	Input Period Measurement
000_011b	Double Action Output compare
000_1000 000_1111	Reserved
001_0bbb	Modulus Counter
001_1000 010_0101	Reserved
010_0110	Output Pulse Width Modulation with Trigger
010_0111 100_1111	Reserved
101_000b	Modulus Counter Buffered (Up counter)
101_0010 101_0011	Reserved
101_010b	Modulus Counter Buffered (Up/Down counter)
101_0110 101_1001	Reserved
101_10b0	Output Pulse Width and Frequency Modulation Buffered
101_11bb	Center Aligned Output Pulse Width Modulation Buffered
110_00b0	Output Pulse Width Modulation Buffered
110_0100 111_1111	Reserved

Fig.2.1.6. – Modalità di funzionamento dei canali eMIOS

Ora vediamo come generare un segnale PWM per il pilotaggio dei DC motor e del servo.

2.1.5. Output Pulse Width Modulation Buffered (OPWMB):

La modalità OPWMB viene utilizzata per generare impulsi con fronte di salita e fronte di discesa posizionabili a piacere. Un contatore esterno, pilotato attraverso la modalità MCB, come ad esempio il Counter Bus A, in uscita dal canale 23, viene selezionato tra i counter in ingresso al canale. Il registro A1 definisce la posizione del primo fronte, mentre B1 definisce quella del secondo. La polarità del segnale di uscita è definita dal parametro EDPOL; se EDPOL vale 1, nel momento in cui il counter è pari al valore di A1 si alzerà un fronte di salita e al match con B1 un fronte di discesa, altrimenti il segnale uscirà in logica negata. Entrambi i registri A1 e B1 possono venire aggiornati tramite rispettivamente i registri A2 e B2, nei quali si possono caricare i nuovi valori desiderati e dai quali questi verranno copiati in A1 e B1 al ciclo successivo.
Il duty cicle del segnale ad onda quadra in uscita sarà quindi pari al rapporto tra B1-A1 e il periodo di ciclo definito nel MCB. In figura Fig.2.2.8. è possibile osservare un esempio del funzionamento di questa modalità.

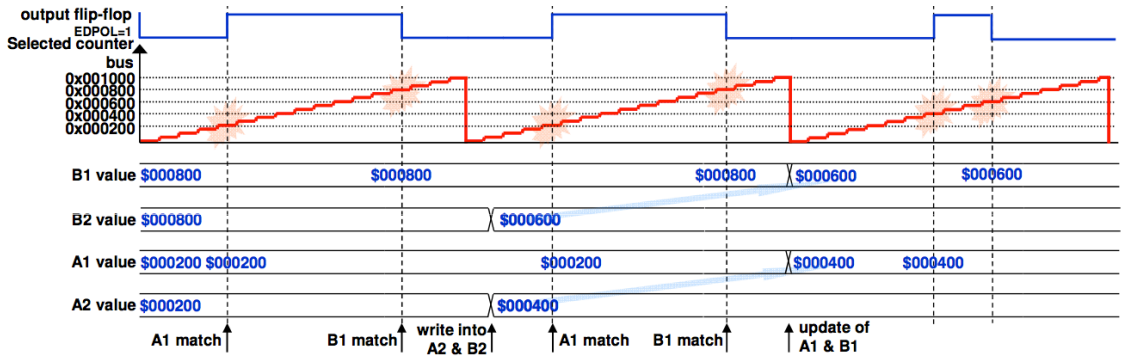


Fig.2.1.6. – Output Pulse Width Modulation Buffered

2.1.6. Generazione di un segnale PWM usando il linguaggio C

Prima di tutto si inizializza che verrà generato dal Clock Generation Module (CGM).

```
void initModesAndClock(void) {
ME.MER.R = 0x0000001D;              /* Enable DRUN, RUN0, SAFE, RESET modes */
                                    /* Initialize PLL before turning it on: */

CGM.FMPLL_CR.R = 0x02400100;    /* 8 MHz xtal: Set PLL0 to 64 MHz */
ME.RUN[0].R = 0x001F0074;           /* RUN0 cfg:
                                    16MHzIRCON,OSC0ON,PLL0ON,syclk=PLL */

ME.RUNPC[1].R = 0x00000010;       /* Peri. Cfg. 1 settings: only run in RUN0 mode */
ME.PCTL[32].R = 0x01;               /* MPC56xxB ADC 0: select ME.RUNPC[1] */
ME.PCTL[57].R = 0x01;               /* MPC56xxB CTUL: select ME.RUNPC[1] */
ME.PCTL[48].R = 0x01;               /* MPC56xxB/P/S LINFlex 0: select
                                    ME.RUNPC[1] */
ME.PCTL[68].R = 0x01;               /* MPC56xxB/S SIUL:  select ME.RUNPC[1] */
ME.PCTL[72].R = 0x01;          /* MPC56xxB/S EMIOS 0:  select ME.RUNPC[1] */
                                    /* Mode Transition to enter RUN0 mode: */
ME.MCTL.R = 0x40005AF0;             /* Enter RUN0 Mode & Key */
```

```
ME.MCTL.R = 0x4000A50F;          /* Enter RUN0 Mode & Inverted Key */
while (ME.GS.B.S_MTRANS) {}      /* Wait for mode transition to complete */

while(ME.GS.B.S_CURRENTMODE != 4) {}  /* Verify RUN0 is the current mode */

}
```

Nel caso in esame il clock viene inviato alle periferiche 1 e 3 (eMIOS e ADC):

```
void initPeriClkGen(void) {
CGM.SC_DC[0].R = 0x80;           /* MPC56xxB/S: Enable peri set 1
                                    sysclk divided by 1 */
CGM.SC_DC[2].R = 0x80;           /* MPC56xxB: Enable peri set 3 sysclk
                                    divided by 1*/
}
```

Il seguente metodo disabilità la funzione Watchdog che prevede la esecuzione di particolari righe di programma ogni intervallo di tempo prestabilito; molto utile per gestire i bug e i Loop infiniti evitando il crash del sistema. Ma questo tipo di gestione degli interrupt non verrà trattato in questo elaborato.

```
void disableWatchdog(void) {
SWT.SR.R = 0x0000c520;           /* Write keys to clear soft lock bit */
SWT.SR.R = 0x0000d928;
SWT.CR.R = 0x8000010A;           /* Clear watchdog enable (WEN) */
}
```

Ora viene inizializzato il modulo eMIOS_0, in particolare, una volta ricevuto in ingresso il System Clock a 64 MHz, generato dal FMPLL, si setta il prescaler in modo da dividerlo fino a raggiungere un internal clock pari 1 MHz che viene così abilitato e inviato a tutti i canali eMIOS tramite il Counter Bus A.

```
void initEMIOS_0(void) {
EMIOS_0.MCR.B.GPRE= 63;          /* Divide 64 MHz sysclk by 63+1 =
                                    64 for 1MHz eMIOS clk*/
EMIOS_0.MCR.B.GPREN = 1;         /* Enable eMIOS clock */
EMIOS_0.MCR.B.GTBE = 1;          /* Enable global time base */
EMIOS_0.MCR.B.FRZ = 1;           /* Enable stopping channels when
                                    in debug mode */
}
```

L’unico canale in grado di settare il Counter Bus A è il canale 23, esso imposta la modalità di funzionamento dell’eMIOS come “Modulus Counter Buffered”, utilizza il

Counter Bus A come ingresso e abilita il preescaler lasciando a 1 il fattore di divisione, genera così un segnale di clock come onda triangolare che viene inviato agli altri canali dell'eMIOS_0. La frequenza del clock è impostata scrivendo il valore 999 all'interno del registro A1, così facendo la transizione di discesa avverrà ogni volta che l'internal counter avrà contato fino a 1000. Se il segnale di ingresso ciclava a 1 MHz, il clock di uscita avrà così un periodo di 1 ms.

```
void initEMIOS_0ch23(void) {                /* EMIOS 0 CH 23: Modulus Up Counter */
EMIOS_0.CH[23].CADR.R = 999;                /* Period will be 999+1 = 1000 clocks (1
                                            msec)*/
EMIOS_0.CH[23].CCR.B.MODE = 0x50;           /* Modulus Counter Buffered (MCB) */
EMIOS_0.CH[23].CCR.B.BSL = 0x3;             /* Use internal counter */
EMIOS_0.CH[23].CCR.B.UCPRE=0;               /* Set channel prescaler to divide by 1 */
EMIOS_0.CH[23].CCR.B.UCPEN = 1;             /* Enable prescaler; uses default divide by
                                            1*/
EMIOS_0.CH[23].CCR.B.FREN = 1;              /* Freeze channel counting when in debug
                                            mode*/
}
```

Il motore DC sinistro è pilotato da un segnale PWM generato dal canale 6 del modulo eMIOS_0. Ciò avviene attraverso la modalità di funzionamento Output PWM Buffered (OPWMB), impostata tramite i bit di mode. Nel dettaglio, una volta selezionato tra i due clock in ingresso al canale il Counter Bus A si settano i registri A1 e B1 con le soglie inferiore e superiore alle quali, ad ogni ciclo, cambiare polarità del segnale ad onda quadra in uscita. Ponendo il campo EDPOL a 1, nell'intervallo temporale compreso tra A e B il segnale PWM in uscita sarà ad un livello di tensione alto (1), mentre sarà basso (0) per il resto del ciclo. L'ultima istruzione assegna al canale il pin di uscita del segnale, in questo caso il Pin 14 sulla Port B. Per il motore di destra, pilotato attraverso il canale eMIOS n° 7 il codice è analogo, il pin di uscita del segnale PWM è il Pin 15 della Port B.

```
//controllo motore a sinistra

void initEMIOS_0ch6(void) {          /* EMIOS 0 CH 6: Output Pulse Width
Modulation*/
EMIOS_0.CH[6].CADR.R = 0;            /* Leading edge when channel counter bus=0*/
EMIOS_0.CH[6].CBDR.R = 0;            /* Trailing edge when channel counter bus=500*/
EMIOS_0.CH[6].CCR.B.BSL = 0x0;       /* Use counter bus A (default) */
EMIOS_0.CH[6].CCR.B.EDPOL = 1;       /* Polarity-leading edge sets output */
EMIOS_0.CH[6].CCR.B.MODE = 0x60;     /* Mode is OPWM Buffered */
SIU.PCR[30].R = 0x0600;              /* MPC56xxS: Assign EMIOS_0 ch 6 to pad */
}

//controllo motore di destra

void initEMIOS_0ch7(void) {          /* EMIOS 0 CH 7: Output Pulse Width
Modulation*/
EMIOS_0.CH[7].CADR.R = 0;            /* Leading edge when channel counter
bus=0*/
EMIOS_0.CH[7].CBDR.R = 0;            /* Trailing edge when channel's counter
bus=999*/
```

```
EMIOS_0.CH[7].CCR.B.BSL = 0x0;   /* Use counter bus A (default) */
EMIOS_0.CH[7].CCR.B.EDPOL = 1;           /* Polarity-leading edge sets output*/
EMIOS_0.CH[7].CCR.B.MODE = 0x60;         /* Mode is OPWM Buffered */
SIU.PCR[31].R = 0x0600;  }               /* MPC56xxS: Assign EMIOS_0 ch 7 to pad */
```

Per concludere, all'interno del main vengono richiamati tutti i metodi creati. Si ottiene così un segnale PWM in uscita sui pin PB[14] e PB[15] della porta B.

```
void main (void)
{
initModesAndClock();
initPeriClkGen();
disableWatchdog();
initEMIOS_0();
initEMIOS_0ch6();
initEMIOS_0ch7();
}
```

2.2. ADC (Analog to Digital Converter)

2.2.1. Caratteristiche generali

Il microcontrollore MPC5604B dispone al suo interno di un ADC ad approssimazioni successive (SAR), nel seguente paragrafo si analizzeranno le caratteristiche principali e il principio di funzionamento. In Fig. 2.1.1 si può osservare lo schema di funzionamento.

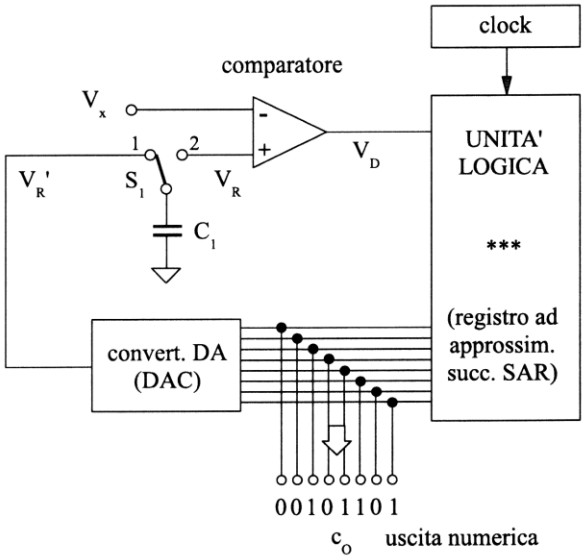


Fig. 2.1.1 Architettura di un ADC ad approssimazioni successive

La conversione in digitale della tensione di ingresso V_x è effettuata per mezzo di confronti ripetuti tra V_x e una o più tensioni V_R di riferimento. In particolare, V_R è fatta variare in modo noto fino al raggiungimento di una condizione di equilibrio, in corrispondenza della quale la quantità differenziale $\Delta V = V_x - V_R$ raggiunge il valore minimo, compatibilmente con la risoluzione dello strumento. In tale condizione, il risultato della misurazione V_x è assunto pari al valore finale V_R di V_R, cioè $V_x = V_R$. Nel

caso in esame la tensione di riferimento V_R è fatta variare per mezzo di un convertitore digitale analogico (DAC).

Analizziamo ora le caratteristiche dell'ADC di tipo SAR integrato nel microcontrollore MPC5604B:

- Risoluzione a 10 bit
- Tempo di conversione supportato: fino a 650 ns
- Fino a 36 canali di input, espandibili a 64 con l'aggiunta di multiplexer esterni
- 16 canali di precisione
- 16 canali standard
- 4 canali standard per il collegamento di multiplexer esterni
- Multiplexer interno per acquisizioni multiple
- Generazione interna di un segnale di tipo address decoder per coordinare gli eventuali multiplexer esterni
- differenti registri per i tempi di campionamento e conversione CTR[0:2] per i canali interni di precisione, i
canali standard e i canali esterni
- 64 registri dedicati per i risultati della conversione di ogni canale
- Precampionamento
- Funzionamento in modalità one-shot, scan, injection e triggered injected
- Configurazione indipendente dei parametri di ogni canale

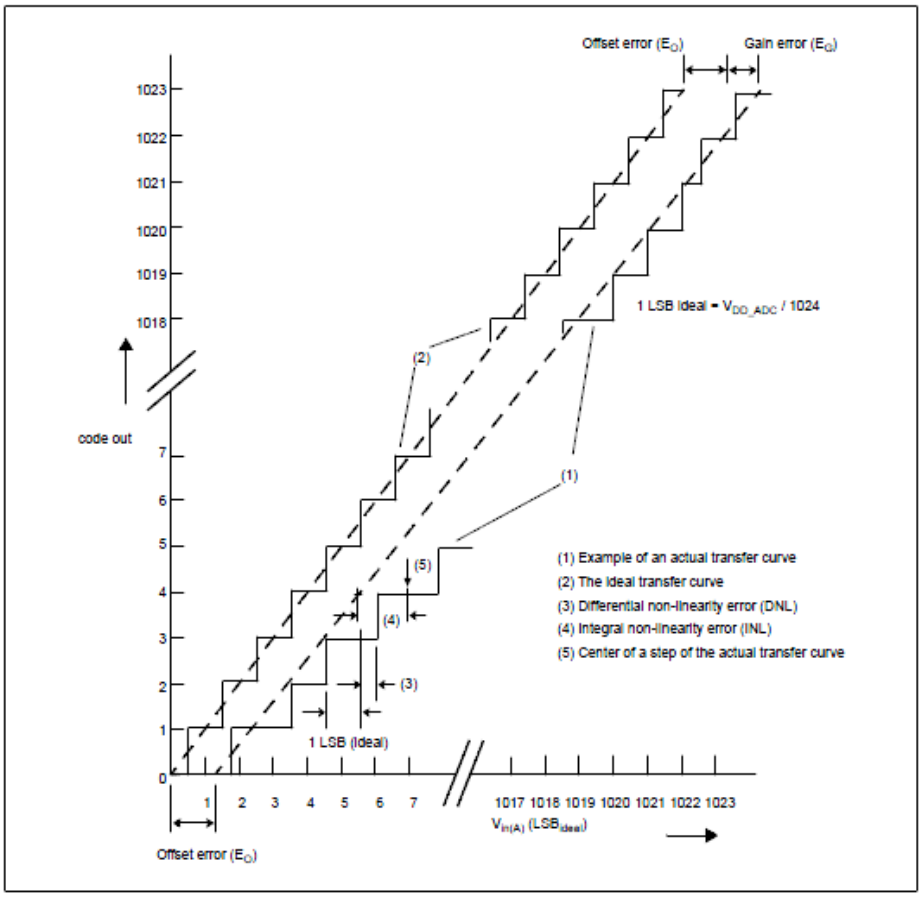


Fig. 2.1.1 Transcaratteristica e definizioni dei tipi di errore

Il modulo ADC (Analog to Digital Converter) offre funzionalità di conversione rapida e accurata per un largo numero di applicazioni. L'ADC dispone di funzionalità avanzate per ogni modalità di conversione. E' possibile, inoltre, triggerare l'inizio della conversione via software o via hardware.

Tipologie di canali:

• Canali interni di precisione ADC0_P[n]
• Canali interni standard ADC0_S[n]
• Canali esterni ADC0_X[n]

Per configurare un certo canale da acquisire basta settare a 1 il rispettivo bit nel corrispondente Conversion Mask Register.

2.2.2. Modalità di conversione

Il modulo ADC permette tre modalità di conversione:
• Normal conversion
• Injected conversione
• CTU triggered conversion

Normal Conversion:

Questa modalità di funzionamento è la più comune e la più utilizzata, essa accetta in ingresso un canale per acquisizione, che può essere abilitato settando a "1" il campo corrispondente del NCMR (normal conversion mask register). Il registro di maschera deve essere programmato prima dell'inizio della conversione e il suo valore non può essere modificato finché la conversione di tutti i canali selezionati è
terminata.
La conversione può iniziare in due modi:
• Via software: la catena di misura si attiva nel momento in cui si alza il bit NSTART nel registro MCR
• Via trigger: la conversione inizia solo se il bit NSTART è settato e se viene rilevato il livello di tensione impostato attraverso il segnale di trigger.
Nel momento in cui inizia l'operazione di conversione il bit NSTART viene rimesso pari a zero, in modo che si possa impostare una nuova conversione, questa avrà inizio al termine di quella in atto. Esistono inoltre due
modalità di funzionamento:
• One shot
• Scan
Per entrare in una di queste modalità è necessario programmare il MCR[MODE] bit
In modalità One Shot (MODE = 0) la conversione sequenziale dei canali specificata nel registro NCMR viene attuata una sola volta, al termine di ogni acquisizione, il valore digitalizzato del campione viene caricato nel corrispondente registro dati.
In modalità Scan (MODE = 1) la conversione sequenziale degli N canali specificati nel registro NCMR è continuamente attuata, come nel caso precedente i valori digitalizzati vengono caricati nei registri specifici al termine di ogni acquisizione.

2.2.3. Sequenza di istruzioni in C per l'acquisizione di segnali analogici

Ora si analizzano tutte le istruzioni per abilitare l'ADC ed acquisire correttamente un segnale di tensione. A titolo di esempio si acquisisce il feedback di corrente proveniente dal motore sinistro e disponibile come input sul pin PB[7]. Se si analizza la *Tabella 4-3. Functional port pin descriptions* disponibile sul manuale della scheda, si può osservare che il pin PB[7] è collegato al canale di precisione n°3 dell'ADC. In conclusione per permettere l'acquisizione è necessario abilitare il pin come GPI, ovvero come ingresso analogico dell'ADC.

```
void initPad (void) {
SIU.PCR[23].R = 0x2000;      /*inizializzo il pin PB[7] come GPI[3], corrente motore sinistro */
}

void initADC(void) {                          /*inizializzo l'ADC*/

ADC.MCR.R = 0x00000000;                       /* ADC modalità one shot */
ADC.NCMR[0].R = 0x00000008;                   /*abilita la conversione canale n°3 */
ADC.CTR[0].R = 0x00008606;                    /* Conversion times for 32MHz ADClock */
ADC.CTR[1].R = 0x00008606;                    /* Conversion times for 32MHz ADClock */
}

void initCTU(void) {
CTU.EVTCFGR[3].R = 0x00008003;                /* motore sx: Configura l'evento sull'eMIOS_0
                                              Ch 3 per triggerare il pin GPI[3] canale 3 ADC*/
}


void LEFT_MOTOR_CURRENT(void)
{
for(i=0; i<10;i++)
{
ADC.MCR.B.NSTART=1;                           /*Inizializza la conversione normale */
while (ADC.MSR.B.NSTART == 1) {};             /*Attendi la fine della conversione*/
curdataSx = ADC.CDR[3].B.CDATA;               /*Salva i dati nella variabile globale curdataSx*/
}
}
```

Infine si richiamano nel main tutti i metodi creati. Per coerenza aggiungiamo anche un metodo per abilitare il motore sinistro.

```
void main()
{
initCTU();
initADC();
initPad();
ENABLE_MOTOR_LEFT ();
LEFT_MOTOR_CURRENT();
}
```

CAPITOLO 3

Scheda di potenza

3.1. Motor drive

3.1.1. Introduzione

Il seguente paragrafo ha lo scopo di introdurre il lettore al vasto mondo degli azionamenti elettrici ed in particolare approfondire sul ponte H, con il fine di offrire le basi per l'implementazione della catena di controllo in retroazione del motore CC fornito dalla freescale.

Spesso il tipo di alimentazione di cui si necessita per pilotare un carico non coincide con quella disponibile, risulta quindi necessario operare una conversione dell'ampiezza o della frequenza di tale tensione, attraverso dispositivi chiamati convertitori. La soluzione più moderna ai problemi di conversione è data dai convertitori statici, basati sull'impiego di interruttori elettronici allo stato solido (diodi, transistori, tiristori), che derivano il loro nome al fatto di non includere alcun organo in movimento. La formidabile diffusione dei convertitori statici, in continuo crescendo a partire dagli anni '60, trova spiegazione nella loro economicità, flessibilità d'impiego e affidabilità, e negli elevati rendimenti energetici che li caratterizzano. Alla base dello sviluppo di convertitori statici di caratteristiche sempre più avanzate sta l'evoluzione delle tecnologie di produzione dei componenti elettronici di potenza, che ha reso disponibili interruttori elettronici sempre più potenti, veloci ed affidabili. Altri motivi che hanno contribuito alla sempre maggiore diffusione dei convertitori statici sono legati allo sviluppo di dispositivi di segnale a semiconduttore, di elevatissima affidabilità e di grande velocità operativa, che hanno reso possibile l'applicazione di tecniche di comando e di controllo sempre più sofisticate.

3.1.2. Principio di funzionamento ponte H

Come tutti i convertitore CC/CC, alla base del principio di funzionamento del ponte H c'è un semplice circuito on/off e lo si può osservare in fig. 3.1.1.

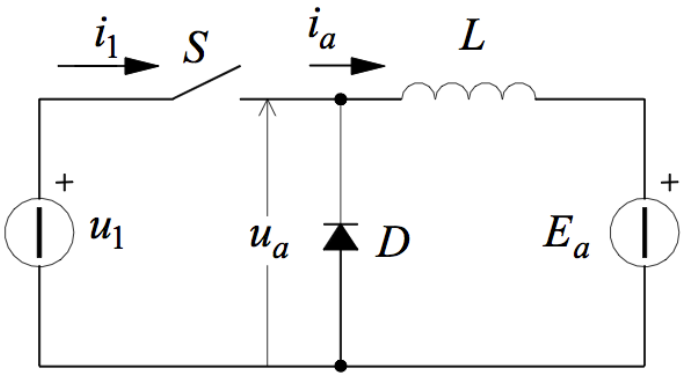


Fig.3.1.1 – schema convertitore c.c./c.c. abbassatore

Nello schema S rappresenta un dispositivo elettronico di commutazione, che andrà scelto in base alle specifiche di potenza e velocità di commutazione della particolare applicazione; si suppone che esso sia in grado di condurre solo nel verso assegnato alla corrente i_1. La sorgente di tensione continua all'ingresso è stata denominata U_1 e nel seguito della trattazione sarà ritenuta costante. Il motore in c.c. è stato schematizzato con la serie di una induttanza *La* e di un generatore di forza elettromotrice $Ea \leq U_1$; tensione e corrente ai capi di tale carico sono stati denotati rispettivamente con u_a ed i_a, dove il carattere minuscolo sta ad indicare che si tratta di grandezze variabili nel tempo. Per spiegare il principio di funzionamento del circuito si può partire da una condizione iniziale di riposo, in cui la corrente i_a sia nulla ed *S* sia aperto. Alla chiusura di *S* il diodo *D* (diodo di libera circolazione o *free wheeling*) si trova interdetto, dunque ai capi di *La* si manifesterà una differenza di potenziale *U1-Ea* e la corrente *ia* inizierà a crescere linearmente, con pendenza (*U1-Ea*)/*La .*

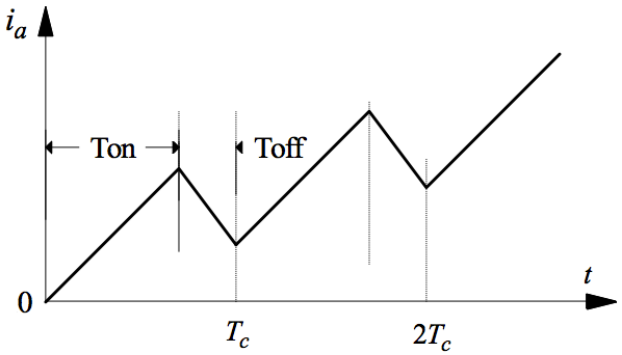


Fig.3.1.2 - Andamento della corrente i_a con carico *La-Ea*

Se all'istante t_{ON} si riapre *S*, la corrente i_a (che non può interrompersi bruscamente, essendo associata ad una certa quantità di energia magnetica accumulata nell'induttore), si richiude attraverso il diodo *D*. Se si assume che nel periodo di durata costante (indicata con T*c* in Fig.3.1.2) la corrente non si annulli mai, (*funzionamento continuo*), la decrescita continuerà fino al termine del ciclo di commutazione. Di seguito, *S* viene nuovamente comandato in chiusura per un intervallo di tempo t_{ON}, e la corrente in *La* ricomincia a crescere, con la stessa pendenza del ciclo precedente. Il convertitore dunque funziona con una continua successione di fasi di conduzione di durata t_{ON} e di interdizione di durata $T_{OFF} = T_C - t_{ON}$. Il rapporto $\delta = t_{ON} / T_C$ è detto *duty cycle*; La tensione media Tc nel periodo Tc è data da $Ua = U_1 * \delta$. Si può dimostrare che, dopo un certo periodo di transizione, il convertitore giunga ad un funzionamento a regime stabile. Gli andamenti della tensione e della corrente in funzionamento a regime sono riportati in Fig.3.1.3. A regime, la tensione media ai capi del carico *La-Ea* coincide con la *fem Ea*; si ha cioè: $Ea = U_1 * \delta \Longrightarrow (U_1 - Ea) * t_{ON} = Ea * T_{OFF}$ che graficamente significa che le due aree A*1* ed A*2* di Fig.3.1.3*(b)* devono coincidere.

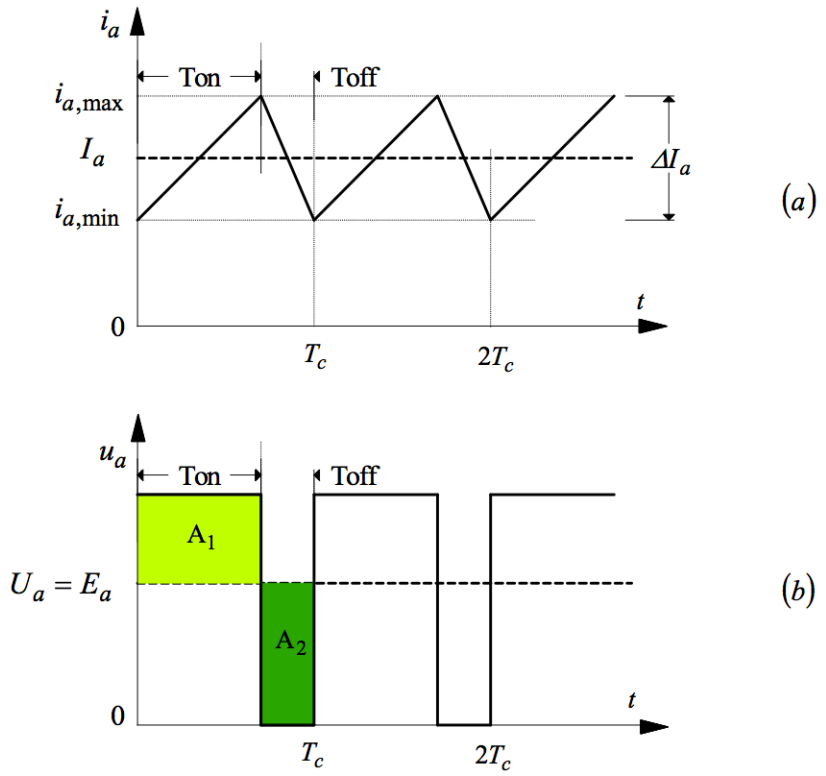
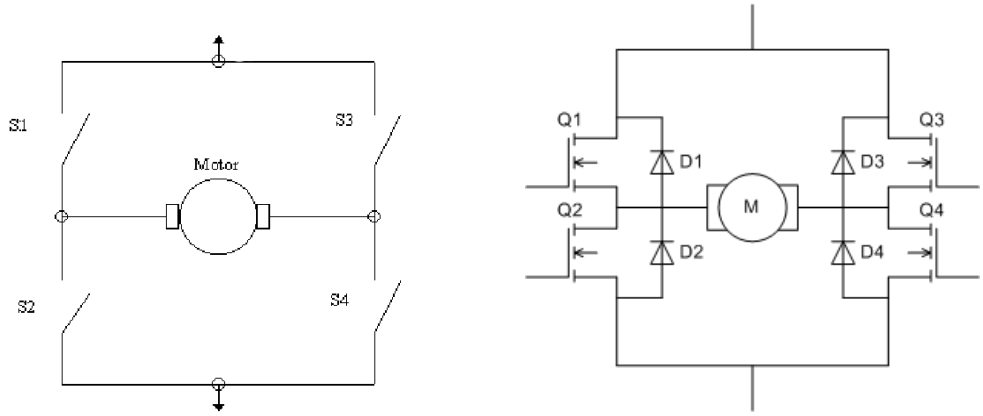


Fig.3.1.3 - Andamenti *(a)*di corrente e *(b)* di tensione, nel funzionamento continuo a regime.

La corrente *ia* a regime presenta una ondulazione Δ*Ia* la cui ampiezza dipende dal duty cycle δ, L'andamento di Δ*Ia* in funzione di δ quale si nota che la situazione più sfavorevole si ha con duty cycle pari al 50%:

Chopper a quattro quadranti o H Bridge

Per capire il funzionamento di un H bridge si consideri la figura 3.1.5:



(a) (b)

Fig.3.1.5. – H bridge, schema di principio

La Fig.3.1.5(a) mostra lo schema di principio del chopper a quattro quadranti a ponte, realizzato con quattro interruttori ideali. La Fig.3.1.5(b) riguarda invece una realizzazione a transistori. Si osservi che in antiparallelo al transistore sono disposti dei diodi che hanno la funzione di fornire una via di richiusura alla corrente di carico quando gli interruttori sono aperti. Quando sono chiusi gli interruttori *S1* ed *S4* (*S2* ed *S3* aperti), la tensione u_O del carico coincide con la tensione continua d'ingresso applicata con polarità positiva (+Ui), permettendo così il senso di rotazione positivo del motore. Quando invece sono chiusi *S2* ed *S3* (*S1* ed *S4* aperti) la tensione del carico vale *-Ui*, invertendo di conseguenza il senso di marcia e fornendo la possibilità di implementare algoritmi di frenata, sterzata attiva e differenziale elettronico in curva. Gli interruttori *S1* ed *S3* non possono essere chiusi simultaneamente, dato che ciò provocherebbe un cortocircuito del generatore di alimentazione. Lo stesso vale per *S2* ed *S4*. Peraltro vi debbono essere sempre almeno due interruttori chiusi, uno per ogni ramo del ponte, per fornire una via di circolazione alla corrente di carico. Le stesse relazioni valgono anche, naturalmente, nel caso dell'H bridge a transistori di Fig.3.1.6(b). Si deve però notare che i transistori, contrariamente agli interruttori ideali di Fig.3.1.6(a), possono condurre corrente in un solo verso. È quindi necessario connettere, in antiparallelo ai transistori su entrambi i rami del ponte ad H (*bridge leg*), dei diodi che consentano il passaggio della corrente anche nel verso opposto (free-wheeling diodes). In particolare, nel semiperiodo t_0-t_2, in cui al carico viene applicata una tensione positiva, la coppia T_1, T_4 può solo condurre nel tratto t_1-t_2, nel quale la corrente è positiva e fluisce nel verso consentito dai transistori. Nel tratto t_0-t_1 in cui è richiesta una tensione positiva, ma la corrente è negativa, conducono invece i diodi *D1* e *D4*. In maniera analoga, nel semiperiodo successivo, i diodi *D2*, *D3* conducono nell'intervallo t_2-t_3, mentre i transistori T_2, T_3 conducono in t_3-t_4. Il funzionamento degli inverter secondo la tecnica illustrata prende nome di funzionamento a onda quadra, a causa della forma d'onda rettangolare di tensione applicata al carico. La regolazione della frequenza si ottiene semplicemente variando la durata degli intervalli di conduzione degli interruttori. È infatti evidente che la tensione u_O applicata al carico ha una frequenza $f = 1/T$, ove T è il periodo di ripetizione della forma d'onda rettangolare. L'ampiezza potrebbe essere regolata invece agendo sul duty cycle.

3.2. MC33931 – Throttle Control H-bridge

Questo dispositivo è stato concepito per un ampio uso nel campo automotive, in particolare per il controllo delle valvole a farfalla elettroniche. Garantisce quindi una elevata affidabilità in termini di robustezza e capacità termiche. E' applicabile a qualsiasi motore DC a bassa tensione che rispetti i limiti elettrici descritti nel datasheet.
In particolare il 33937 H-Bridge è in grado di controllare carichi induttivi con correnti fino a 5 A di picco ed è dotato di un regolatore interno di corrente che viene attivato per correnti intorno a 6.5 A ± 1.5 A. Inoltre è possibile generare tensioni di output con frequenze fino a 11 kHz.

3.2.1. Caratteristiche

• 5.0 to 28 V continuous operation (transient operation from 5.0 to 40 V)
• 235 mΩ maximum RDS(ON) @ TJ=150 °C (each H-bridge MOSFET)
• 3.0 V and 5.0 V TTL / CMOS logic compatible inputs
• Over-current limiting (regulation) via internal constant-off-time PWM
• Output short-circuit protection (short to VPWR or GND)
• Temperature-dependant current-limit threshold reduction
• All inputs have an internal source/sink to define the default (floating input) states
• Sleep mode with current draw < 50 µA (each half with inputs floating or set to match default logic states)

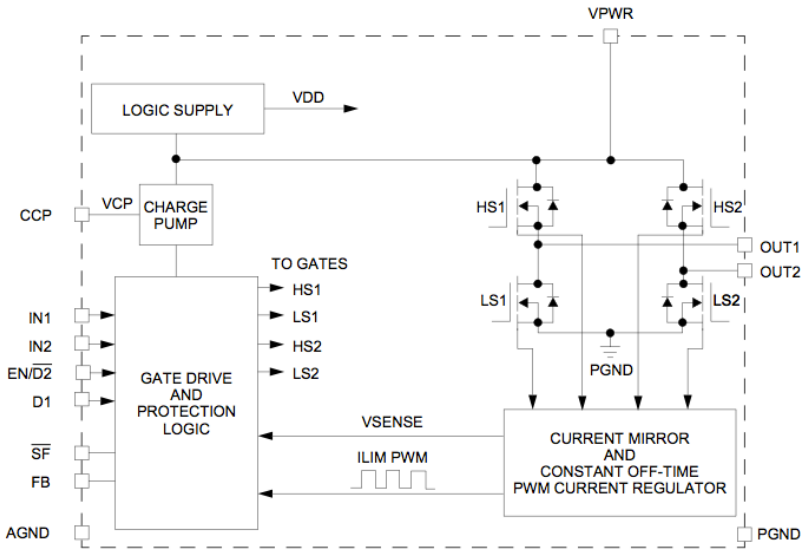


Fig 3.2.1 – Diagramma a blocchi *MC33931* semplificato

Dal diagramma a blocchi si può osservare la architettura generale del MC33931:

- Circuito analogico di controllo e di protezione dei mosfet
- Ponte H
- Specchio di corrente per il Feedback

Il 33931 è stato progettato per un uso a 360 gradi nel controllo di motori a bassa tensione. Sono forniti due input indipendenti per il controllo della polarità in uscita dagli half-bridge e altri due input permettono di forzare alla condizione tristate l'uscita del ponte H. Inoltre dei sensori di temperatura e di corrente integrati direttamente nei mosfet di uscita garantiscono la massima affidabilità e la protezione degli stadi di uscita. Un regolatore di tensione integrato gestisce la logica interna, riassunta nella tabella sottostante (Fig 3.2.2).

Device State	Input Conditions				Status	Outputs	
	EN/$\overline{D2}$	D1	IN1	IN2	$\overline{SF}$	OUT1	OUT2
Forward	H	L	H	L	H	H	L
Reverse	H	L	L	H	H	L	H
Freewheeling Low	H	L	L	L	H	L	L
Freewheeling High	H	L	H	H	H	H	H
Disable 1 (D1)	H	H	X	X	L	Z	Z
IN1 Disconnected	H	L	Z	X	H	H	X
IN2 Disconnected	H	L	X	Z	H	X	H
D1 Disconnected	H	Z	X	X	L	Z	Z
Under-voltage Lockout	H	X	X	X	L	Z	Z
Over-temperature	H	X	X	X	L	Z	Z
Short-circuit	H	X	X	X	L	Z	Z
Sleep Mode EN/$\overline{D2}$	L	X	X	X	H	Z	Z
EN/$\overline{D2}$ Disconnected	Z	X	X	X	H	Z	Z

Fig 3.2.2 - Logica di controllo

Come si vedrà in seguito, attraverso il software si può accedere a ciascun PIN di controllo e quindi è possibile ottenere i vari stati del dispositivo in modo molto flessibile.

Per esempio in fig. 3.2.3 si può osservare come circola la corrente in quattro stati diversi.

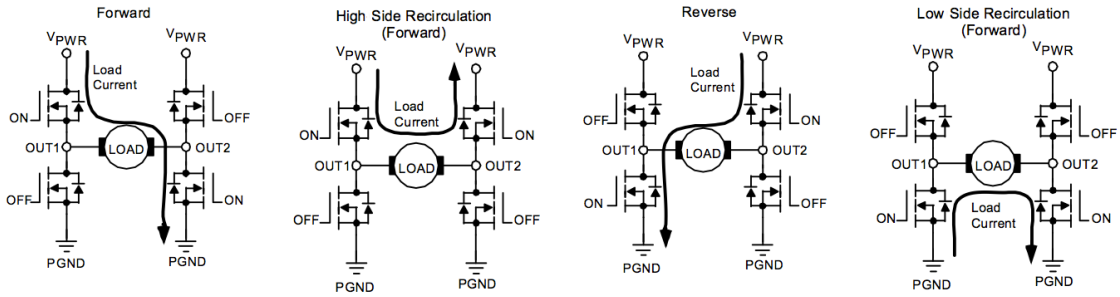


Fig 3.2.3 – Possibili configurazioni del ponte H

Feedback di corrente

Il 33931 ha un PIN dal quale è possibile leggere il feedback per monitorare in tempo reale la corrente con la quale si alimenta il motore da controllare. E' possibile quindi creare un controllo in catena chiusa di velocità e coppia del motore. Quando siamo nella configurazione Forward o Reverse, il pin FB fornisce un segnale analogico che rappresenta lo 0.24% della corrente in uscita dai pin OUT1 e OUT2. Quindi per acquisire il segnale è necessario l'uso di un circuito esterno che mi permetta di leggere una tensione proporzionale alla corrente attraverso un ADC. Per avere una conversione corrente–tensione lineare il range della resistenza da utilizzare all'interno del circuito di condizionamento è 100 Ω < R_{FB} < 300 Ω. Per concludere all'interno del circuito di condizionamento è necessario aggiungere un condensatore a piccolo valore di capacità in parallelo alla resistenza R_{FB} per sopprimere gli spike di tensione dovuti alla modulazione PWM.

3.3. Istruzioni C per il pilotaggio del H-bridge

A titolo di esempio si vedrà come pilotare il ponte H nella funzione Reverse per controllare il motore sinistro. Si supponga di avere già abilitato il modulo eMIOS_0 e i canali eMIOS_0 CH6. Dal manuale della MPC5604B al capitolo *Signal description* si può osservare che il segnale PWM abilitato è disponibile sia al pin PB[14] sia al pin PA[6]. Quindi si collega fisicamente il pin IN1 del ponte al pin PB[14] della scheda e il pin IN2 al pin PA[6]. Prima di tutto si abilita il motore sinistro settando ad un livello logico alto (High) il pin EN/D2 collegato al pin PB[0]:

```
SIU.PCR[16].R = 0x0200;          /* PB[0] abilitato come output*/
SIU.PGPDO[0].R |= 0x00008000;    /*setto a 1 il bit n°16 della porta B*/
```

Offset[1]	Register	Field																														
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0C00	PGPDO0	Port A																Port B														
0x0C04	PGPDO1	Port C																Port D														
0x0C08	PGPDO2	Port E																Port F														
0x0C0C	PGPDO3	Port G																Port H														

Fig 3.3.1. Mappa dei registri PGPDO

Come si può osservare in Fig 3.3.1. le porte A e B sono internamente collegate al registro PGPDO[0] formato da 32 bit, uno per ciascun pin esterno. Quindi per settare il livello logico dei bit interessati è necessario solo una riga di codice in C. Per il pin D1 invece non occore settare nulla perché è gia collegato a massa della *motor drive.* Per concludere si setta ad un livello logico basso il pin IN1 ed infine al pin IN2 arriverà la PWM creata dal modulo eMIOS.

```
SIU.PCR[6].R = 0x0200;                  /* PA[6] abilitato come output*/
SIU.PGPDO[1].R &= 0xfffffffc;           /*metto ad un livello logico basso il pin PA[6] */
SIU.PCR[30].R = 0x0600;                 /*abilita la porta come output,canale 6 eMIOS_0*/
EMIOS_0.CH[6].CBDR.R = PWML;            /*Assegna al canale 6 eMIOS_0 la PWML
                                        desiderata*/
```

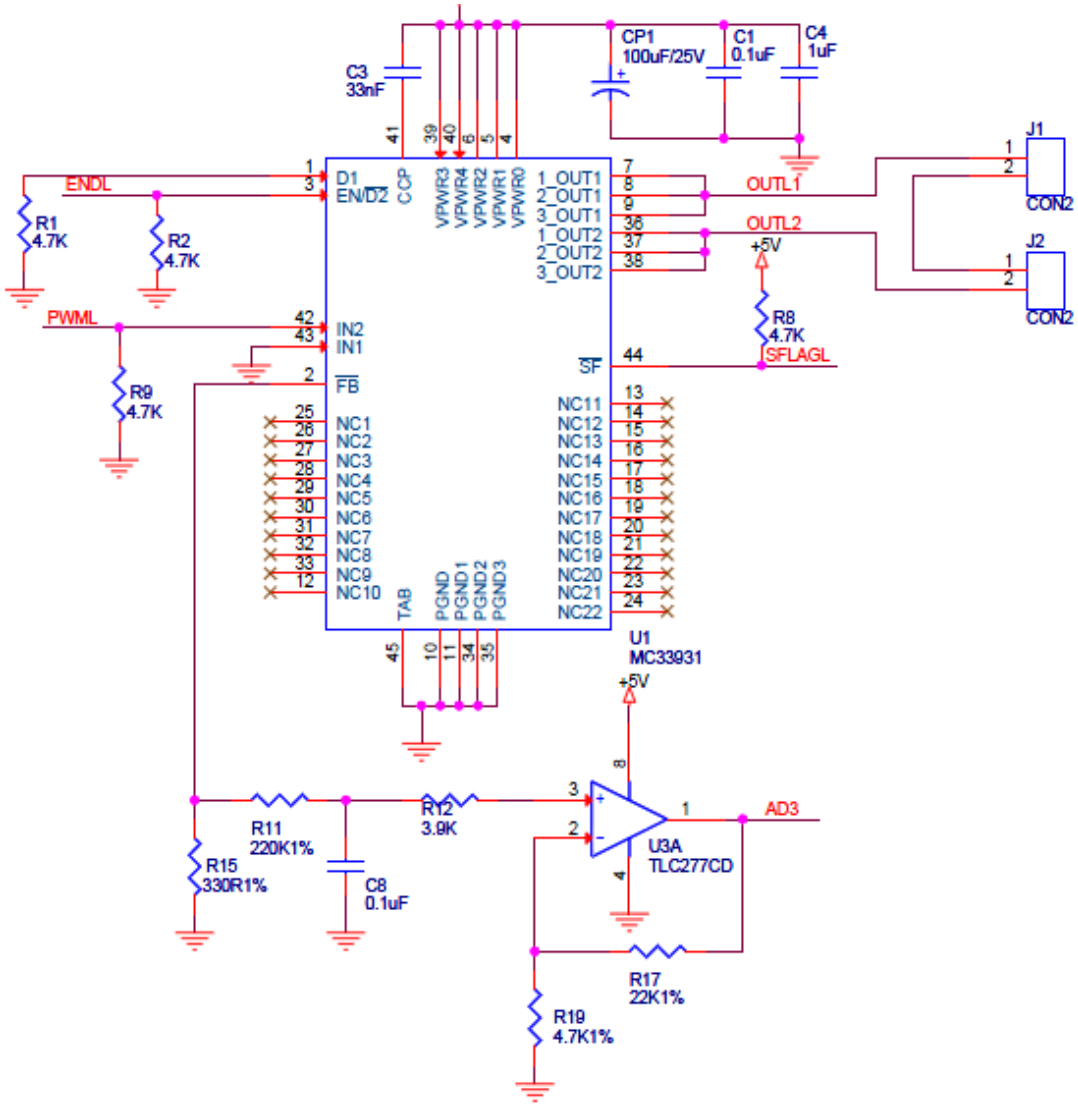


Fig.3.3.2 Circuito di connessione del ponte H alla Motor Drive

CAPITOLO 4

Motore in corrente continua

4.1. Struttura e funzionamento

Il motore elettrico è il componente che trasforma l'energia elettrica erogata dal convertitore statico in energia meccanica necessaria al moto delle parti meccaniche. E' dunque un attuatore. Nella scrittura delle equazioni, spesso si usa separare il motore in due blocchi funzionali, il primo che tenga conto degli avvolgimenti e delle interazioni elettromagnetiche, ed il secondo che consideri gli aspetti meccanici, quali le relazioni tra coppia prodotta, attriti e momenti d'inerzia. In Fig.4.1.2 è riportato per una maggiore comprensione la catena di elementi "causa-effetto" che agiscono in un qualunque motore elettrico. Essa è schematicamente riportata nella Fig.4.1.1.

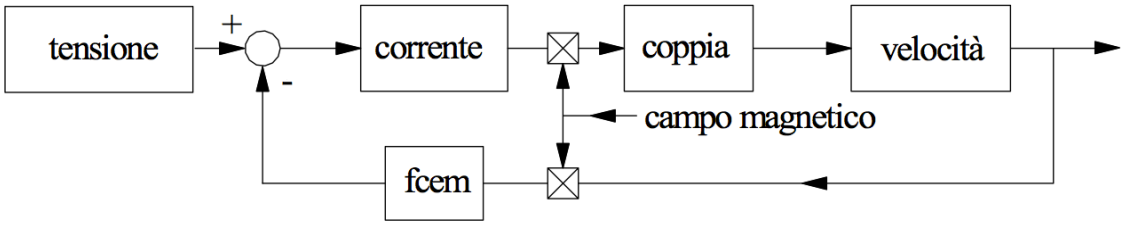


Fig.4.1.1 – Schema a blocchi di un motore elettrico

4.1.1. Motore CC a magneti permanenti

Ogni motore elettrico è costituito essenzialmente da due parti: statore e rotore. In Fig 4.1.1 si può una rappresentazione semplificata. Il motore fornito dalla *Freescale Cup* funziona a corrente continua e il suo circuito di eccitazione è costituito da magneti permanenti.

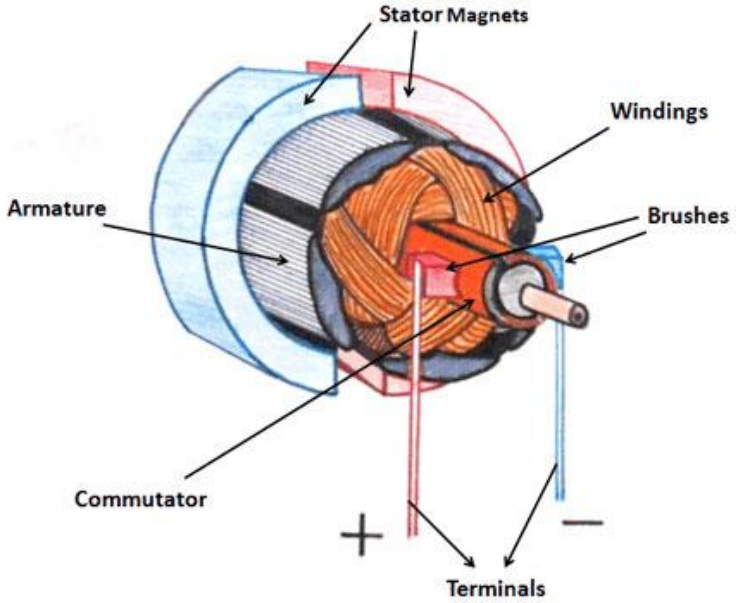


Fig 4.1.1 – Magneti permanenti di statore e avvolgimenti di rotore

Per capire il principio di funzionamento si ipotizzano le seguenti semplificazioni:

- Circuito di eccitazione costituito da solo due magneti permanenti
- Presenza di due conduttori per ciascuna cava di rotore
- Flusso magnetico di eccitazione perfettamente radiale al traferro

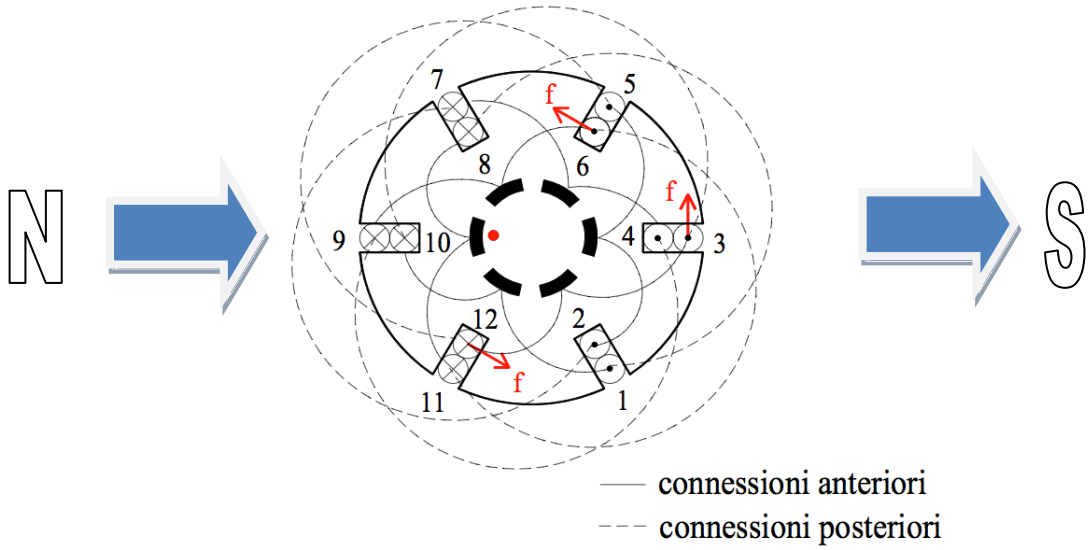


Fig 4.1.2 – Eccitazione avvolgimenti di rotore

A titolo di esempio, consideriamo che il conduttore attivo 3 di figura sia sede di una corrente positiva *i* con il verso indicato. Se B è l'induzione radiale al traferro nella posizione occupata dal conduttore in esame, su di esso verrà ad agire la forza tangenziale mostrata in figura, di intensità $f = l\,(i \times \mathcal{B})$ essendo *l* la lunghezza assiale del conduttore. Se la stessa corrente i percorre anche tutti gli altri conduttori sotto il polo *S* e, con il verso contrario come evidenziato in figura, anche quelli sotto il polo *N*, su tutti verranno ad agire forze tangenziali di versi concordi (alcune indicate in figura), proporzionali al valore dell'induzione radiale al traferro nella posizione occupata dai conduttori stessi. In queste condizioni il rotore è sottoposto ad una coppia risultante diversa da zero che tende a metterlo in rotazione. Il rotore è alimentato da un sistema spazzole-collettore e grazie alle opportune connessioni tra i vari conduttori e le lamelle (mostrate in Fig 4.1.2) si ha una distribuzione delle correnti rispetto ai poli che rispetta quella di Fig 4.1.2 qualunque sia la posizione del rotore sicché la coppia prodotta per una data corrente mantiene sempre lo stesso valore e segno. Se il numero di cave è sufficientemente alto, si può senz'altro assumere che anche durante la commutazione si venga a produrre sempre la stessa coppia. Essa sarà proporzionale all'intensità delle correnti nei conduttori, e quindi ad i_a, e all'induzione media sotto ciascun polo, e quindi a φ, e si può esprimere:

$$\tau = K\tau\, ia\, \varphi$$

Dove $K\tau$ rappresenta un coefficiente che dipende dal tipo di avvolgimento e dal numero di conduttori attivi.
Ora se si ipotizza che il rotore sia in movimento con una velocità angolare ω, grazie alla legge di Faraday-Lenz si può calcolare la f.e.m. elementare indotta su ciascun conduttore. Essa risulta proporzionale al valore B dell'induzione radiale al traferro nel punto ove giace il conduttore, e alla velocità periferica di rotazione secondo la relazione $e = B\, l\, v$. In presenza di una velocità positiva, cioè nel verso delle coppie

positive (secondo la *convenzione di segno dei motori),* i versi secondo i quali agiscono le f.e.m. elementari dovute al flusso induttore sono mostrati in Fig.4.1.3. Da un confronto con la Fig.4.1.2, dove sono indicati i versi positivi delle correnti, si deduce che tali *f.e.m.* per come sono state orientate sono più propriamente forze controelettromotrici. La *f.e.m.* di armatura e_a che si manifesta fra le spazzole si ottiene come somma delle f.e.m. elementari dei conduttori che si incontrano percorrendo l'avvolgimento indotto dal morsetto negativo a quello positivo lungo una delle vie interne che esso presenta.

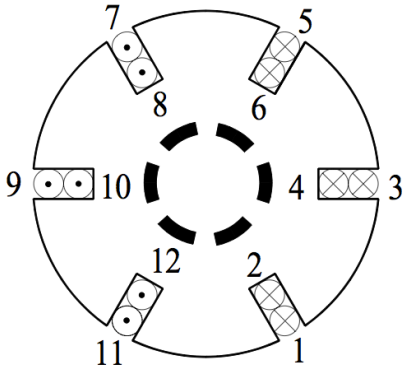


Fig 4.1.3 – Versi delle f.e.m. elementari indotte sui conduttori

La *f.e.m*. di armatura disponibile alle spazzole sarà proporzionale alla velocità periferica del rotore, e quindi a quella di rotazione ω, e all'induzione media sotto ciascun polo, e quindi a φ, e potrà in ultima analisi essere espressa con la

$$\mathrm{e_a} = Ke\ ia\ \varphi$$

Dove K_e rappresenta un coefficiente che dipende dal tipo di avvolgimento e dal numero di conduttori attivi. Durante la rotazione vengono a cambiare i conduttori attivi che sono in serie nelle vie interne dell'avvolgimento fra una spazzola e l'altra, ma la *f.e.m.* risultante rimane invariata. La *f.e.m.* indotta di armatura è abbinata ad un assorbimento di potenza pari a

$$p = i_a\, e_a$$

Avendo adottato la convenzione di segno dei motori, il prodotto

$$p = \tau\ \omega$$

fornisce il valore della potenza meccanica sviluppata dalla macchina. A questo punto, se si uguagliano le due espressioni della potenza, si ottiene che

$$K_e = K_\tau$$

Per concludere si deve tener conto della resistenza dell'avvolgimento e della sua induttanza, si ottiene la seguente espressione

$$u_a = R_a\ i_a + L_a \frac{di_a}{dt} + e_a$$

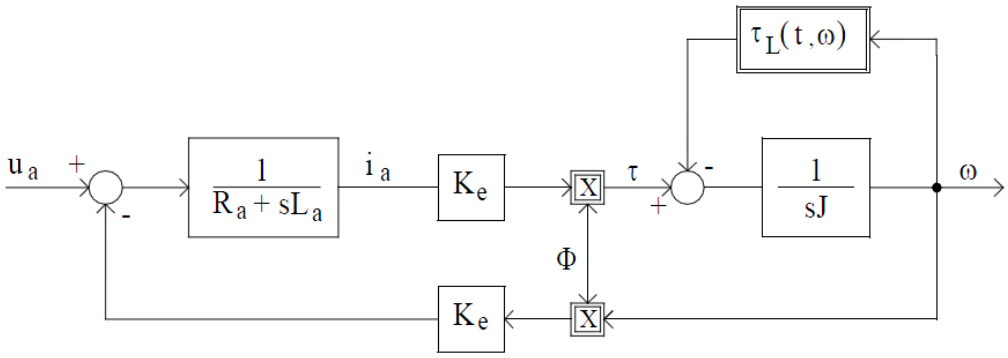


Fig 4.1.4 – Schema a blocchi motore CC

Alle equazioni sopra scritte corrisponde lo schema a blocchi di Fig.4.1.4 valido *nel dominio del tempo*, anche se alcuni blocchi (lineari) sono contrassegnati con la funzione di trasferimento in *s* che li descrive. Inoltre si può osservare anche l'influenza del momento d'inerzia totale J riferita all'albero e della coppia di carico τ_L. In questa trattazione però sono stati trascurati questi due ultimi termini al fine di avere un sistema di partenza molto semplificato.

4.1.2 Funzionamento a tensione impressa

L'espressione T=T(Ω) consente di individuare le strategie di controllo del motore, ossia come controllarne la coppia, e quindi la velocità, per un prefissato carico. Tali strategie sono in genere distinte in controllo di armatura e controllo di campo. La prima si può attuare sia a corrente impressa che a tensione impressa. In questo paragrafo si analizza il comportamento a tensione impressa. Le *caratteristiche meccaniche T=T*(Ω) forniscono la coppia sviluppata in funzione della velocità di rotazione per il funzionamento a regime in diverse condizioni di alimentazione. Esse consentono in particolare di evidenziare i comportamenti peculiari del motore a corrente continua e di individuare le grandezze su cui si può agire per controllarne la coppia o la velocità. Il comportamento a regime si può analizzare risolvendo le equazioni del paragrafo precedente dopo aver posto a zero ogni derivata e ipotizzando il flusso ø di eccitazione costante (magneti permanenti). Si ottiene allora

$$Ua = Ra\, Ia + Ke\text{ø}\, \Omega$$
$$\mathrm{T} = Ke\text{ø}\, Ia$$
$$\text{ø} = cost$$

Dalla prima equazione si ricava

$$Ia = \frac{Ua - Ke\emptyset\, \Omega}{Ra}$$

E sostituendo la espressione della corrente *Ia* si ottiene

$$T = Ke\emptyset \frac{Ua - Ke\emptyset\,\Omega}{Ra}$$

E' l'espressione cercata T(Ω). Essa rappresenta una retta con intersezioni in T_S ed Ω_0

$$Ts = Ke\emptyset \frac{Ua}{Ra}$$

$$\Omega o = \frac{Ua}{Ke\emptyset}$$

Che rappresentano rispettivamente la coppia di spunto e la velocità a vuoto.

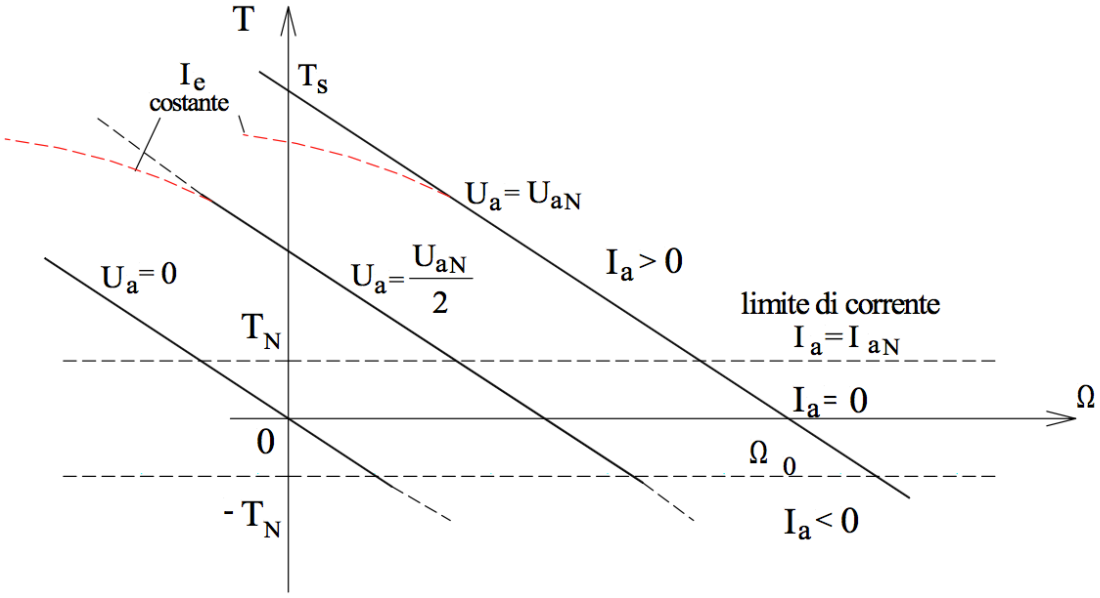


Fig 4.1.5 – Caratteristiche meccaniche *T*=*T*(Ω) con tensione d'armatura impressa

Se si dispone di una convertitore in grado di erogare una tensione di riferimento (modulazione PWM) e ipotizzando una coppia di carico costante, è possibile spostare il punto di lavoro del motore a piacere per ottenere la velocità desiderata a regime.

4.2. Standart Motor "rn260c" winding 18130

4.2.1. Misura della resistenza di armatura

E' stato eseguito un primo test attraverso la modalità di misura a quattro terminali ma è stato riscontrato il problema che le correnti iniettate negli avvolgimenti del motore dallo strumento sono troppo deboli per vincere le resistenze di contatto spazzole-collettore. Per la seconda prova, invece, è stato usato il metodo di misura Volt-Amperimetrico. Grazie all'alimentatore GW Instek GPS-33030 è stata iniettata una corrente e si è misurata la caduta di tensione attraverso un multimetro tradizionale. I risultati sperimentali sono stati riportati nella seguente tabella. Si può osservare una piccola diversità di resistenze di armatura dei due motori a causa delle normali differenze nella costruzione dei motori.

Motore Destro		Motore Sinistro	
V [V]	I [A]	V [V]	I [A]
0,21	0,09	0,29	0,11
0,24	0,11	0,33	0,12
0,26	0,13	0,36	0,13
0,31	0,15	0,44	0,15
0,37	0,18	0,53	0,18
0,41	0,21	0,62	0,21
0,51	0,25	0,72	0,27
0,68	0,31	0,81	0,31
0,88	0,41	1,05	0,41
0,95	0,45	1,16	0,47
1,05	0,51	1,25	0,51
RD = 2,095 Ω		RS = 2,297 Ω	

4.2.2. Misura sperimentale della costante $K_E Ø$

La misurazione consiste nell'applicare una velocità di rotazione costante al rotore per mezzo di un trapano. La velocità di rotazione viene rilevata da un tachimetro digitale. Per rilevare la forza elettromotrice si collegano le fasi del motore ad un multimetro tradizionale. I risultati ottenuti evidenziano una legame lineare tra la velocità di rotazione del rotore e la f.e.m. che si instaura all'interno dei conduttori e quindi si può calcolare il valore medio.

Motore Sinistro		Motore Destro	
Velocità(rpm)	Fcem(mV)	Velocità(rpm)	Fcem(mV)
0	0	0	0
50	160	81	244
100	280	148	450
150	460	195	587
204	625	244	740
275	840	270	820
307	940	377	1140
349	1066	473	1437
456	1395	503	1530
532	1620	535	1630
613	1870	612	1820
Keø =	0,029098	Keø =	0,028908

4.2.3. Dati di catalogo

Ora si procede alla lettura dei dati di catalogo del motore “rn260c”

Data Sheet		
Rated Voltage	V	4,5
No Load Speed	rpm	10000
No Load Current	m A	130
Max efficiency Current	m A	510
Max efficiency Speed	rpm	7950
Max efficiency Torque	gcm	18
Max output Current	m A	1070
Max output Speed	rpm	5000
Max output Torque	gcm	44
Stall current	m A	2000
Stall Torque	gcm	88

Stall current, corrente di spunto, è la massima corrente che il motore può sostenere alla partenza. Da questa informazione si può ottenere un riscontro della resistenza degli avvolgimenti di armatura, ricavandola come rapporto tra tensione nominale e corrente di spunto.

$$Ra = \frac{Ua}{Is} = \frac{4{,}5}{2} = 2{,}25\ \Omega$$

No-load speed, no-load current. Rappresentano il punto di lavoro del motore a vuoto, cioè senza carico all'albero quando è alimentato alla tensione nominale. E' possibile fare un riscontro con il coefficiente $\mathrm{Ke}\phi$

$$\mathrm{Ke}\phi = \frac{\mathrm{U_{a,N}} - \mathrm{R_a I_{P0}}}{\Omega_{\mathrm{P0}}} = \frac{4{,}5 - 2{,}25\ 0{,}13}{1000\ \frac{2\pi}{60}} = 4{,}02\ 10^{-3}\ \mathrm{Vs/rad}$$

Inoltre si può calcolare la coppia nel funzionamento alla tensione nominale, e che non contribuisce alla coppia utile

$$\mathrm{T_{po}} = \mathrm{Ke}\phi\ \mathrm{I_{po}} = 5{,}23\ 10^{-3}\ \mathrm{Nm} = 5{,}329\ \mathrm{gcm}$$

infine si calcola anche la coppia di spunto generata e quella effettiva

$$T_{s,gen} = K e\phi I_s = 8{,}04\ 10^{-3} Nm = 82\ gcm$$

$$T_s = T_{s,gen} - T_{po} = 82 - 5{,}329 = 76{,}67\ gcm$$

Rated speed, rated current, rated torque. Rappresentano il punto di funzionamento nominale.

$$T_{N,gen} = K e\phi I_N = 2{,}05\ 10^{-3} Nm = 21\ gcm$$

Per calcolare la coppia effettiva si sottrae a quella generata la coppia spesa per matenere in moto il rotore senza carico

$$T_N = T_{N,gen} - T_{po} = 21 - 5{,}329 = 16\ gcm$$

Ora è possibile calcolare la potenza nominale in uscita (Rated output power)

$$P_N = T_N \Omega = 0{,}00155 \cdot 7950 \cdot \frac{2\pi}{60} = 1{,}287\ W$$

4.3. Breaking e filtro motore DC

A causa della progettazione poco curata della Motor Control Board, il segnale analogico AD0 proveniente dalla telecamera e la corrente che alimenta i motori hanno il GND in comune. Di fatto si avevano grandi interferenze nei segnali acquisiti. Per sopprimere questi rumori sono stati usati tre condensatori da 100nF per ciascun motore. Lo schema di collegamento dei condensatori è mostrato in figura fig. 4.3.1. Inoltre è stata fatta la scelta di inviare il segnale AD0 direttamente dalla telecamera alla scheda MPC5604B. Con questo by-pass, la motor drive viene esclusa completamente dal mondo analogico dei “piccoli segnali”.

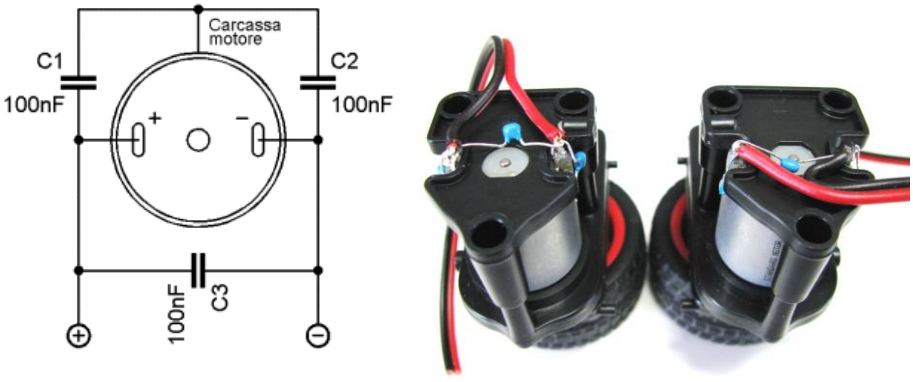
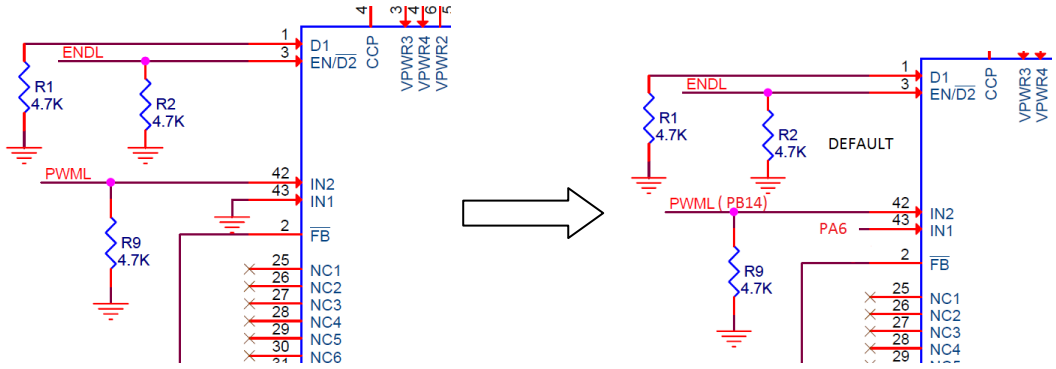


Fig 4.3.1 – Filtro motore DC

Breaking

Per garantire elevate prestazioni al veicolo è indispensabile la frenatura. Come si può osservare in figura 4.3.2 la motor drive fornita dalla Freescale non è stata predisposta per permettere ai motori di andare in retromarcia. Infatti il PIN IN1 è collegato alla massa della motor drive. Per questo motivo il PIN IN1 è stato sollevato e collegato al PIN PA6 della scheda MPC5604B, nel quale avremo in uscita o un segnale PWM oppure GND a seconda se si vuole andare in retromarcia oppure in avanti. Un procedimento analogo è stato fatto per il controllo del motore destro.



DEFAULT

Fig 4.3.2 – Collegamento pin IN1 e IN2

CAPITOLO 5

Servo comando

5.1. Struttura e funzionamento

I servo motori sono nati per essere pilotati nel modo più semplice possibile. Questi dispositivi sono noti per la grande affidabilità, prontezza ed elevata coppia. Per questo motivo hanno trovato fin da subito un vasto impiego nel campo della robotica. Come si può osservare dallo schema a blocchi di Fig. 4.3.3. il servo è controllato da un segnale analogico in ingresso. Ma il suo punto di forza è rappresentato dal suo sistema di retroazione costituito da un treno di ingranaggi e da un potenziometro. Infatti, grazie all'elettronica di controllo (in questo caso un NE544), istante per istante viene calcolato l'errore tra la posizione attuale del rotore e il riferimento di posizione. Una volta raggiunto il riferimento, il rotore rimane bloccato in attesa di un nuovo segnale di comando.

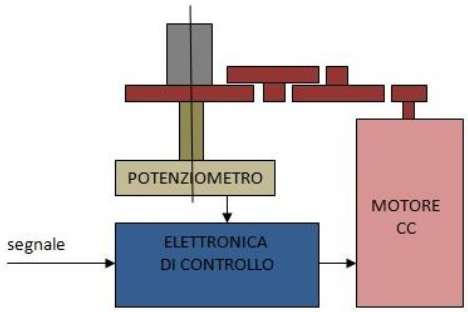


Fig 5.1.1 – Schema a blocchi di un servomotore analogico

Un servo RC dispone solitamente di soli tre fili attestati ad un connettore femmina per pin strip a passo 2.54mm come visibile in Fig. 4.3.4. Due di questi fili sono riservati all'alimentazione in corrente continua a 5 volt Il positivo è di colore rosso, il negativo di colore nero. L'assorbimento massimo è di circa 180mA per un Futaba S3010 ma può variare a seconda delle caretteristiche del servo. Il terzo filo, normalmente di colore bianco, è riservato per il controllo del posizionamento. Su questo filo è necessario applicare un segnale impulsivo o PWM le cui caratteristiche sono "quasi" univoche per qualsiasi Servo RC disponibile in commercio. Il servo attende un impulso ogni 20 ms, la durata dell'impulso determinerà la posizione angolare che dovrà assumere il rotore. Generalmente con un impulso di durata pari a 1.5ms il perno del servo RC si posiziona esattamente al centro del suo intervallo di rotazione. Da questo punto il perno può ruotare fino a -90 gradi (senso antiorario) se l'impulso fornito ha una durata inferiore a 1.5ms e fino +90 gradi (senso orario) se l'impulso fornito ha durata superiore a 1.5ms. Il rapporto esatto tra la rotazione del perno e la larghezza dell'impulso fornito può variare tra i vari modelli di servo. Per tutto il tempo in cui il segnale rimane attivo in ingresso al circuito di controllo il servo manterrà la medesima posizione angolare, nel momento in cui il segnale varia il proprio periodo, l'albero si riposizionerà di conseguenza. Perché l'albero non si sposti durante l'esercizio, per come è costituito

l'azionamento, il segnale di controllo necessita di essere aggiornato in continuazione, questo è il motivo per il quale si utilizza un segnale periodico per pilotare il circuito di controllo.

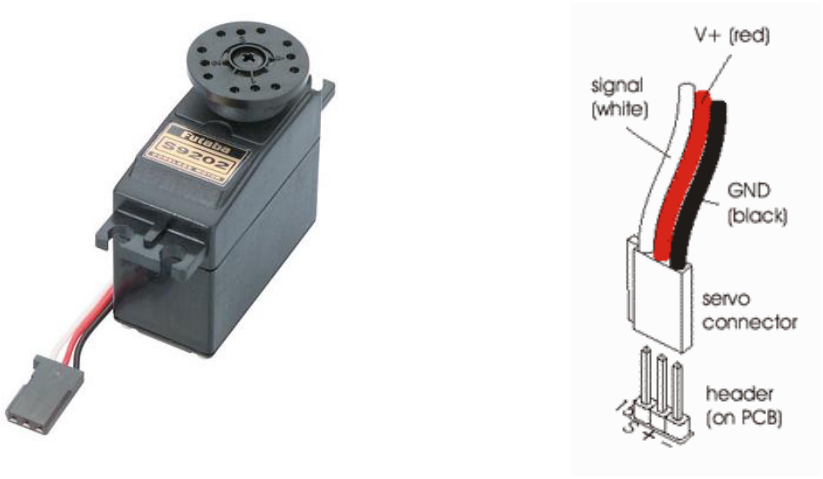


Fig 5.1.2 – Connettori del servo Futaba

5.2. Futaba S3010 Standart High-Torque BB Servo

5.2.1. Specifiche tecniche

Il servo comando in dotazione è il "Futaba S3010",che in questa specifica applicazione viene pilotato con una tensione di 5 V attraverso la Motor Drive, di seguito si elencano le specifiche tecniche:

• **Velocità:**
0.20 sec/60° a 4.8 V
0.16 sec/60° a 6 V
• **Coppia:**
5200 gcm a 4.8 V
6500 gcm a 6 V
• **Dimensioni:**
40 x 20 x 38 mm
• **Peso:**
41 g

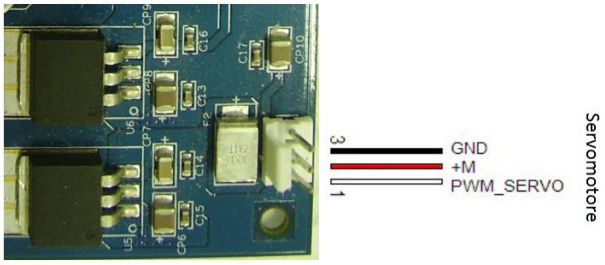


Fig 5.2.1 – Collegamento tra Motor Drive e Servo

5.2.2. Taratura servo

La taratura dello sterzo è il punto fondamentale prima di partire con la implementazione del controllo di posizione del veicolo. Una centratura errata causerebbe un errore di posizione costante rispetto al riferimento da seguire (linea nera). E nel caso sia stato implementato un controllore che prevede un calcolo dell'errore tramite una componente proporzionale al proprio integrale (controllori PI e PID) comporterebbe una saturazione quasi immediata del controllore che, in assenza di adeguate precauzioni, porterebbe il sistema alla instabilità.
Per la centratura sono state eseguite diverse prove in rettilineo, alla fine sono stati ottenuti i seguenti risultati:

- Servo tutto a destra 1050
- Centro Servo 1410
- Servo tutto a sinistra 1770

Questi sono i valori da inviare tramite software ai registri del canale 4 dell'eMIOS, che produrrà in uscita alla porta PB[12] una PWM proporzionale a tale valore.

5.2.3. Istruzioni in linguaggio C per il controllo dello sterzo

Come accennato prima, il comando del servo è collegato attraverso la Motor Drive alla porta PB[12].

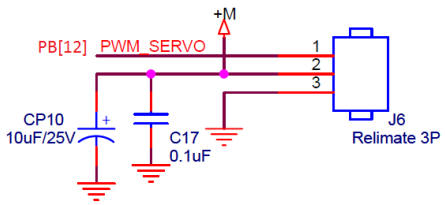


Fig 5.2.2 – Collegamento tra Motor Drive e microcontrollore MPC5604B

Di seguito viene descritta la procedura per attivare questa porta in modo da avere come uscita una PWM.

```
void initEMIOS_0ch4(void) {                /* EMIOS 0 CH 4: PWM come output*/
    EMIOS_0.CH[4].CADR.R = 0;              /* Leading edge when channel counter bus=0*/
    EMIOS_0.CH[4].CBDR.R = centroServo;    /* Inizializzo il servo in posizione centrale*/
    EMIOS_0.CH[4].CCR.B.BSL = 0x01;        /* Use counter bus B */
    EMIOS_0.CH[4].CCR.B.EDPOL = 1;         /* Polarity-leading edge sets output */
    EMIOS_0.CH[4].CCR.B.MODE = 0x60;       /* Mode is OPWM Buffered */
    SIU.PCR[28].R = 0x0600;                /* MPC56xxS: Assign EMIOS_0 ch 6 to pad */
}
```

Una volta inizializzato il canale eMIOS, per riposizionare il servo basta richiamare il registro CBDR del canale 4 dell'eMIOS ed assegnare il valore desiderato.

```
EMIOS_0.CH[4].CBDR.R = controlloServo;
```

CAPITOLO 6

Line Scan Camera

6.1. TSL1401–DB Line Scan Camera

In questo paragrafo si spiega come funziona la camera TSL1401-DB e in particolare come può essere usata per lo specifico scopo di seguire una traiettoria composta da una linea nera su uno sfondo bianco in base alle specifiche dettate dalla competizione Freescale Cup. Fondamentalmente la TSL1401-DB consiste in un array da 128x1 fotodiodi e una lente montata da 7.9 mm. Tutto questo permette un campo di visualizzazione uguale alla distanza dal soggetto. Alcuni dei vantaggi nel impiegare questa camera sono i seguenti:

- Semplicità
- Frequenza di acquisizione manipolabile dall'utente
- Buona qualità di definizione per le applicazioni di *line following*
- Lente rimovibile ed intercambiabile per una vasta scelta di risoluzioni

Oltre ai benefici sposti sopra, ci sono due svantaggi da tenere in considerazione. Il primo è l'uscita puramente analogica, che significa che l'utente deve essere "creativo" per elaborare il segnale in modo da renderlo comprensibile. Il secondo svantaggio è che la camera è modo dimensionale, vale a dire che legge un array di pixel alla volta e quindi se non si sceglie una frequenza di acquisizione adeguata si rischia di perdere preziose informazioni dell'andamento del percorso.

6.1.1 Struttura e funzionamento

La camera è di tipo lineare, prodotto dalla TAOS, modello: "TSL1401CL", esso è costituito da un array di 128x1 fotodiodi associati a dei circuiti integratori e a dei circuiti di sample&hold, che permettono di fornire simultaneamente il tempo di inizio e di fine integrazione per tutti i pixel. L'array comprende 128 fotodiodi, ognuno dei quali dispone di una superficie fotosensibile di 3524.3 μm^2 separata da quella adiacente da uno spazio di 8 μm. La gestione del dispositivo risulta agevole in quanto gli unici segnali da fornire alla scheda sono un clock e un serial input. Per comprendere al meglio la struttura interna del sensore è utile osservare lo schema a blocchi di Fig.6.1.1, dove AO sta per Analog Output, CLK sta per Clock e SI per Serial Input, ovvero il segnale che definisce l'inizio di ogni acquisizione; VDD infine rappresenta la tensione di alimentazione sia per il circuito analogico sia per quello digitale.

Caratteristiche:

- Organizzazione degli elementi: 128 x 1
- Risoluzione di 400 DPI
- Elevata linearità e uniformità
- Ampio Dynamic Range 4000:1 (72 dB)[1]
- Tensione di uscita analogica riferita a GND
- Basso ritardo di immagine
- Frequenza di lavoro fino a 8 MHz
- Alimentazione unica a 3 o 5 V
- Nessuna resistenza di carico richiesta per il funzionamento
- Dispositivo a norma RoHS

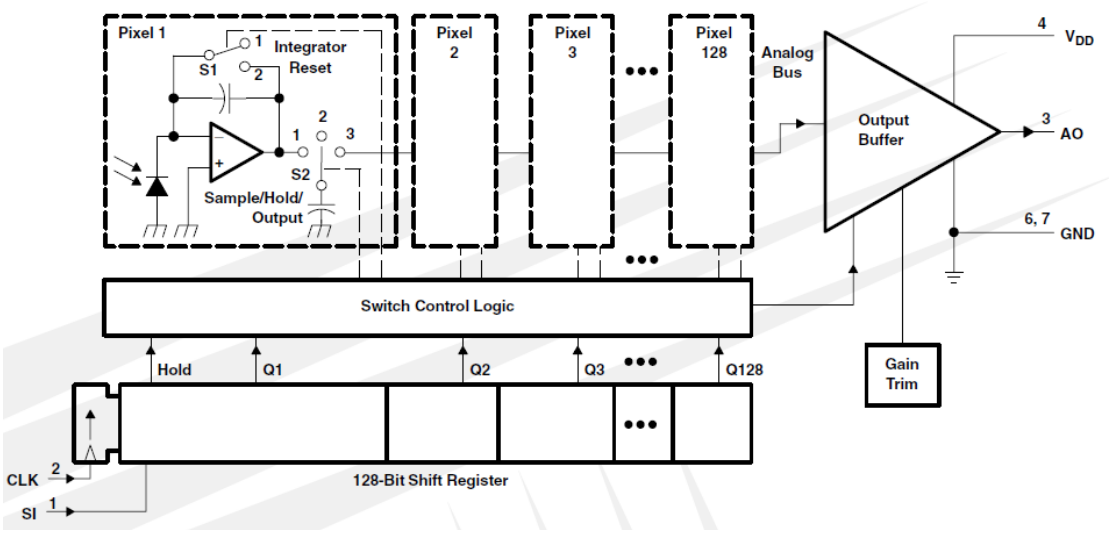


Fig 6.1.1– Diagramma a blocchi TSL1401-DB

La camera consiste quindi in 128 fotodiodi sistemati come un array lineare. L'energia luminosa incidente su un fotodiodo genera una piccola corrente. Durante il periodo di integrazione, tale corrente entra nel circuito integratore e va a caricare un condensatore collegato all'uscita attraverso un interruttore analogico. La quantità di carica accumulata dal condensatore è proporzionale all'intensità luminosa e al tempo di integrazione. Da questa ultima osservazione si può facilmente capire che per avere una buona qualità del segnale in uscita è importante avere una buona illuminazione. Infatti l'aumento del tempo di integrazione ha lo svantaggio di contribuire in modo inversamente proporzionale alla frequenza di acquisizione. La lettura della tensione analogica in uscita e l'operazione di reset degli integratori sono gestite dall'azione combinata di una logica di reset e da un registro di shift a 128 bit. Entrambe gestibili via software. Ciascun ciclo di lettura è inizializzato quando il segnale "SI" si porta al livello logico alto. Per un utilizzo corretto del dispositivo è necessario che l'impulso inviato su SI torni ad un livello logico basso prima del fronte di salita successivo del segnale di clock. Quando viene rilevato il livello logico "1" del segnale SI ha inizio la fase di HOLD. Viene generato un segnale di Hold che viene immediatamente trasmesso a tutti gli interruttori analogici che scollegano i condensatori di campionamento dai rispettivi circuiti di integrazione, contemporaneamente inizia anche il periodo di reset dei circuiti integratori. A questo punto ciascun condensatore viene collegato in modo sequenziale all'amplificatore di uscita che genera la tensione analogica AO. Dopo i primi 18 cicli di clock si ha il completo reset dei circuiti integratori, il nuovo ciclo di integrazioni comincia subito al 19° ciclo di clock. Al 129° ciclo di clock lo *shift register* ha scorso tutto l'array e l'uscita analogica *AO* assume uno stato ad alta impedenza. E' importante notare che il 129° fronte di salita è fondamentale per leggere l'uscita del 128° fotodiodo e per preparare la logica interna di controllo ad una nuova catena di acquisizioni. Se si desidera un tempo di integrazione più basso possibile è consentito inviare un nuovo impulso su *SI* con un piccolo delay (dovuto al tempo di trasferimento della carica di un pixel) dopo il 129° ciclo di clock. Il segnale AO è dato dall'uscita di un amplificatore operazionale in modalità buffer tri-state (adattamento di impedenza), quindi non è necessaria una resistenza esterna di pull-down. Questa progettazione dei circuiti analogici permette una tensione in uscita con un particolare andamento a scalini. Alla tensione di alimentazione V_{DD} = 5 V si ottiene:

- 0 V in assenza di luce
- 2 V per il normale livello di bianco
- 4,8 V in condizioni di saturazione di luce

Quando il dispositivo non è in modalità output, in uscita si ha uno stato di alta impedenza. Per ridurre l'effetto di eventuali disturbi un condensatore di bypass di 100nF dovrebbe essere connesso tra V_{DD} e GND il più vicino possibile al dispositivo.
La tensione disponibile in uscita è

$$V_{OUT} = V_{drk} + R_e \cdot E_e \cdot T_{int}$$

Dove:
V_{OUT} è la tensione di uscita in corrispondenza di luce bianca
V_{drk} è la tensione di uscita in assenza di luce
R_e è la responsività del sensore a una determinata lunghezza d'onda
E_e è la radiazione incidente
T_{int} è il periodo di integrazione

Periodo di Integrazione

Il periodo di integrazione dell'array lineare è il periodo durante il quale la radiazione luminosa viene assorbita dal fotodiodo e convertita in carica elettrica la quale viene accumulata sui condensatori dei circuiti di integrazione di ciascun pixel. La famiglia TAOS TSL14xx deve la sua flessibilità alla possibilità di variare tale periodo di integrazione, modificando così la tensione analogica di uscita corrispondente ai vari livelli di bianco, evitando di entrare in saturazione.

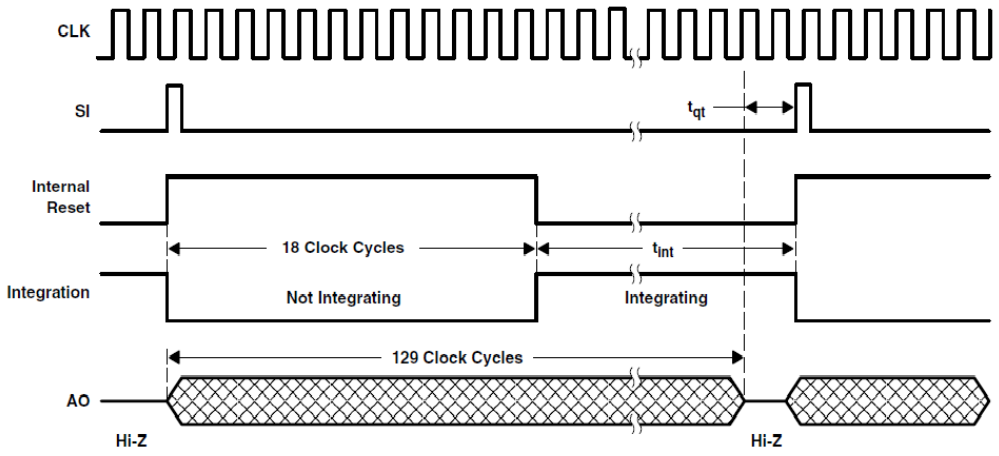


Fig.6.1.2 – Andamento dei segnali di controllo e di uscita

Il periodo di integrazione si definisce come l'intervallo di tempo compreso tra un impulso positivo su SI (*start integration)* e l'impulso positivo di *HOLD,* a questa quantità si debbono sottrarre i primi 18 cicli di clock. I TSL14xx linear array di norma questi tempi coincidono perché nella configurazione il pin di *SI* e di *HOLD* sono connessi insieme. Inviando un impulso a *SI* ha inizio un ciclo di lettura della tensione di output e di reset degli integratori (18 cicli di clock), al termine di tale ciclo è possibile ripetere l'operazione eventualmente con un leggero ritardo, tuttavia è bene considerare che per non perdere l'accuratezza di misura, un impulso su *SI* e il successivo non possono essere distanziati di più di 100 ms. Introdurre un modesto ritardo permette di ottenere un'uscita analogica ad una tensione più alta, ciò risulta utile soprattutto nelle applicazioni a bassa luminosità. Ciascun pixel dell'array lineare consiste in un diodo fotosensibile che converte l'intensità luminosa in una tensione che carica un

condensatore di campionamento. La fase di carica ha inizio quando lo switch S2 in Fig.6.1.1 si porta in posizione 1 (fase di campionamento) Come è possibile osservare in Fig.6.1.2, prima di ogni fase di integrazione è necessario resettare i circuiti integratori per iniziare una nuova conversione, ciò è possibile commutando lo switch S1 di Fig.6.1.1 in posizione 2. Alla ricezione dell'impulso su *SI* tutte le tensioni disponibili all'uscita dei circuiti integratori di ogni pixel vengono prelevate commutando gli switch S2 di ciascun pixel dalla posizione 1 alla posizione 2 (fase di hold). A questo punto la tensione accumulata ai capi dei condensatori di campionamento di ciascun pixel viene letta portando in sequenza il corrispondente interruttore S2 in posizione 3 (fase di output). Al raggiungimento del 18° ciclo di clock (tempo di reset del circuito integratore) gli switch S1 si portano in posizione 1, permettendo così una nuova fase di carica. Le prime 18 tensioni di output vengono lette contemporaneamente alla fase di reset, terminata quest'ultima gli switch S2 dei pixel corrispondenti tornano in posizione 1, permettendo ai condensatori di campionamento di caricarsi ancora una volta. Dal diciannovesimo ciclo in poi i vari pixel vengono interrogati sequenzialmente portando S2 in posizione 3, leggendo il valore di tensione in uscita e commutando al ciclo successivo il medesimo selettore in posizione 1. Il minimo periodo di integrazione per un dato array è determinato dal tempo richiesto per scorrere tutti i pixel e dal tempo necessario al trasferimento della carica; ma dato che quest'ultimo è una costante si può affermare che il periodo di integrazione è funzione esclusivamente della frequenza del clock e del numero di pixel dell'array. Abbassando la frequenza di clock si ha un aumento del minimo periodo di integrazione e una riduzione della massima intensità luminosa che porta a saturazione l'uscita.

Il periodo minimo di integrazione può essere calcolato attraverso l'equazione

$$T_{int,min} = \frac{1}{frequenza\ massima\ di\ clock} \cdot (n - 18)\ pixel + 20\ \mu s$$

Dove n è il numero di pixel e 20μs è il tempo necessario per il campionamento (carica condensatore).

Nel caso del TSL1401, caratterizzato da una frequenza massima di clock di 8 MHz, il periodo minimo di integrazione vale

$$T_{int,min} = 0{,}125\ \mu s\ \cdot (128 - 18) + 20\ \mu s = 33{,}75\ \mu s$$

6.2. Interpretazione, elaborazione segnale e scelte strategiche

6.2.1. Interpretazione segnale

Come accennato prima la camera è una combinazione di fotodiodi e una lente da 7.9 mm. La luce proveniente dall'ambiente entra attraverso la lente, la quale devia la luce nei fotodiodi. Questi ultimi generano una corrente proporzionale alla luce incidente. Ogni fotodiodo è collegato tramite un circuito integratore e degli switch di controllo ad una condensatore di campionamento. Quindi la carica che si ottiene ai capi dei condensatori risulta dipendente dall'intensità luminosa. La tensione dei condensatori associata ad ogni pixel viene poi letta in modo sequenziale. Infine in uscita si ottiene un unico segnale analogico visibile da un oscilloscopio

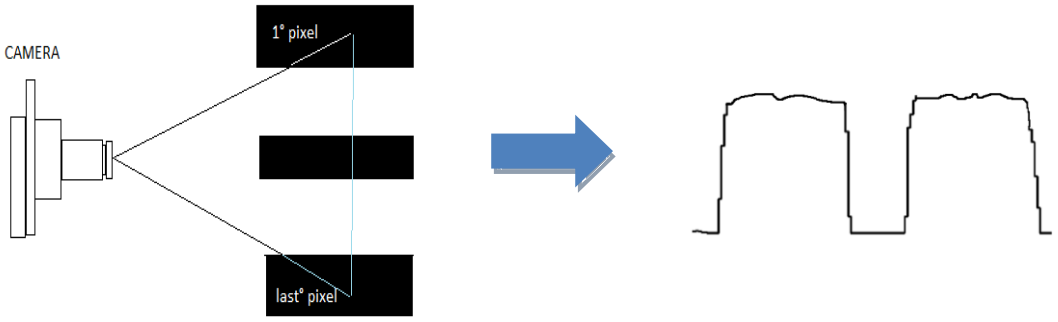


Fig.6.2.1 – Interpretazione segnale AO

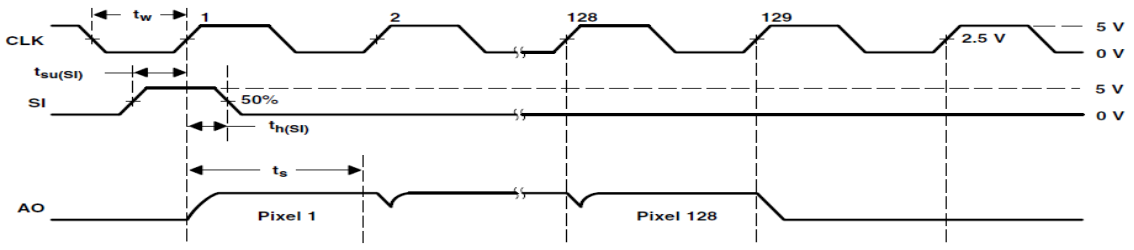
Data la natura lineare del sensore, è impossibile ottenere una visione d'insieme del panorama attraverso un'unica acquisizione, quello che si ottiene è invece l'acquisizione di campo di visione lineare, l'uscita analogica fornirà dei livelli di tensione alti per i pixel molto irradiati, e bassi per i pixel che non ricevono nessuna radiazione in ingresso.

Per un utilizzo normale della camera, l'utente deve curare la gestione di tre segnali:

• CK (clock)
• SI (serial input)
• AO (analog output)

dove CK e SI sono segnali di input, mentre AO rappresenta l'uscita analogica del sensore. L'utente può manipolare in modo flessibile la frequenza di acquisizione della camera agendo sul segnale di clock e sul segnale SI. Aumentando la frequenza del clock in ingresso, aumenta la velocità di lettura dei valori di ciascun pixel e diminuisce anche la distanza temporale tra un segnale SI e il successivo, quindi si ottiene una frequenza di acquisizione della telecamera maggiore. Ma è importante sottolineare che una frequenza di acquisizione elevata comporta meno tempo dedicato alla carica dei condensatori di campionamento.
E' fondamentale evidenziare come questi tre segnali risultino sincronizzati, ciò comporta che, per un utilizzo corretto della *Line Scan Camer,* è necessario prestare attenzione ai seguenti fattori:

• Entrambi i segnali devono avere lo stesso periodo
• I due segnali di ingresso devono essere perfettamente sfasati di mezzo periodo
• L'uscita analogica AO sarà in fase con il segnale di clock (CK)



Una volta che i 129 cicli di clock siano passati, è importante spegnere il segnale di clock in ingresso alla camera, questo permette una migliore carica dei condensatori nel rimanente periodo di integrazione.

6.2.2. Posizionamento camera e miglioramento qualità segnale

Il problema principale in questa competizione è quello del posizionamento della camera. Infatti rappresenta l'unico anello di retroazione utile allo scopo di inseguire la traiettoria voluta. Se da un lato posizionare la telecamera molto in alto permette di anticipare molto l'andamento della traiettoria e quindi velocità più elevate, dall'altro lato la telecamera bassa garantisce un'ottima qualità del segnale. Occorre quindi ottenere un buon compromesso. La soluzione adottata è stata quella di usare due telecamere, permettendo così una maggiore flessibilità del veicolo sia ad alti che a bassi regimi.

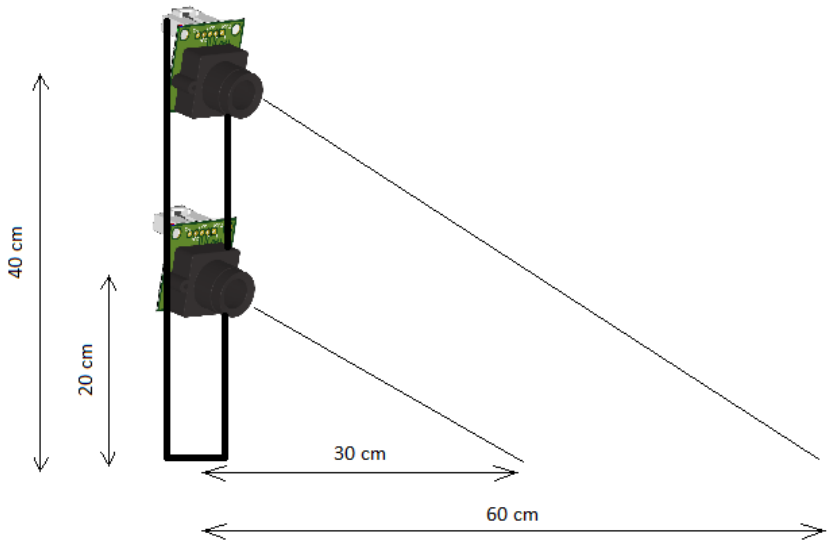


Fig.6.2.2 – Posizionamento telecamere

Il passo successivo è la messa a fuoco della camera. Il modo più immediato è quello di collegarla ad uno oscilloscopio. Certamente occorre prima fornire i segnali SI ed CLK in ingresso tramite un software opportuno, ma del quale si parlerà nei paragrafi successivi. Quindi, supponendo di avere già impostato tutti i segnali in ingresso necessari e che la camera focalizzi un particolare del percorso di gara, il segnale obiettivo da ottenere è quello caratterizzato da un andamento simile ad un'onda quadra.

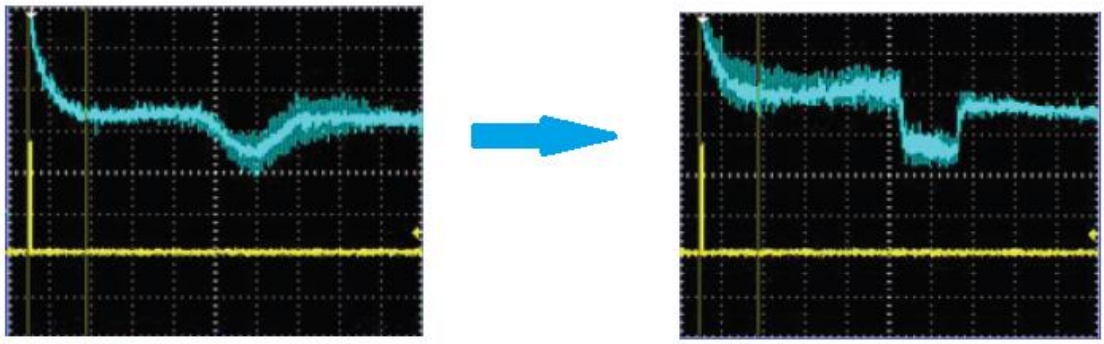


Fig.6.2.3 – Confronto tra due segnali, il primo con camera senza messa a fuoco

Miglioramento qualità segnale

A seconda del metodo adottato per l'analisi del segnale, le condizioni di illuminazione ambientale possono influenzare notevolmente le acquisizioni della camera. Con queste ultime considerazioni è stata presa la decisione di aggiungere una illuminazione dedicata grazie a dei Led. In figura 6.2.4. si può osservare il circuito progettato.

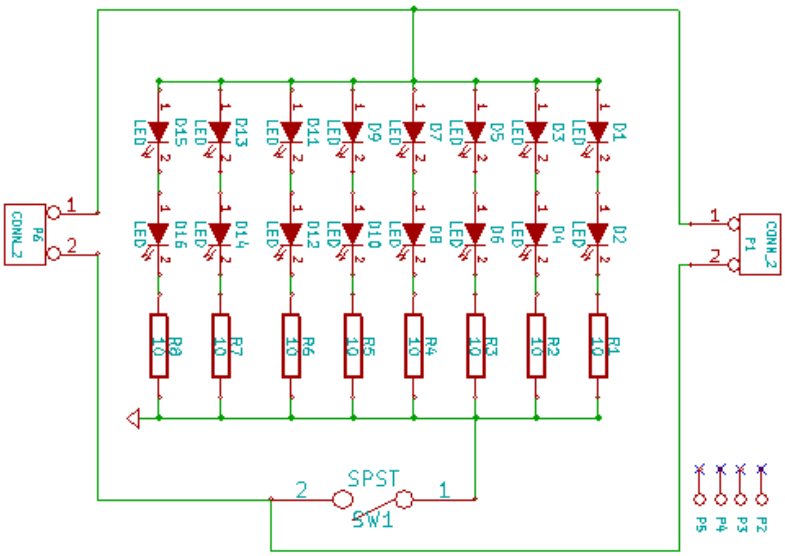


Fig.6.2.4 – Schema Circuito con i Led

I Led utilizzati sono i Nichia White Led, modello NSPW500GS-K, da 5V, assorbono una corrente di 30 mA e hanno una intensità luminosa di 33 cd. Per stabilizzare la tensione a 5 V è stato utilizzato un regolatore di tipo LM7805, che ha garantito una illuminazione costante al variare della tensione della batteria. Grazie all'illuminazione dedicata è stato possibile migliorare notevolmente il contrasto bianco/nero del percorso di gara. Inoltre ciò ha risolto i problemi riconducibili alla scarsa illuminazione in presenza di tunnel, previsti dal percorso di gara.

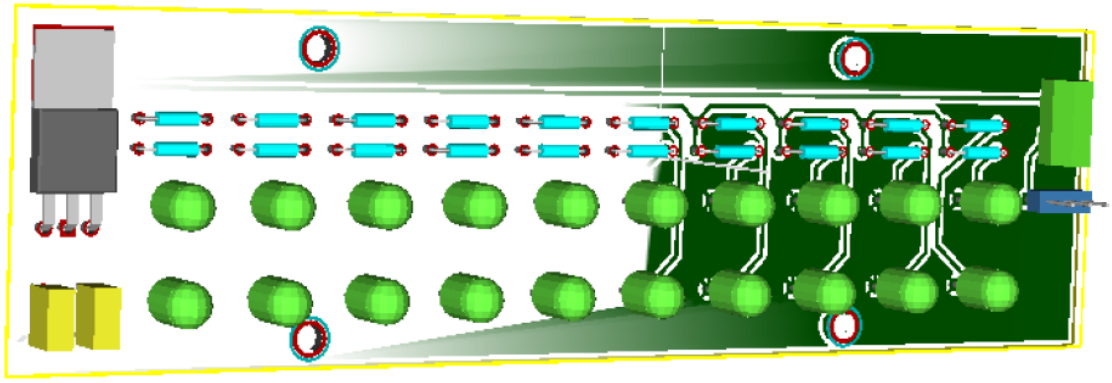


Fig.6.2.4 – Rendering 3D della scheda PCB con i Led

6.2.3. Elaborazione del segnale

Per comprendere a fondo il segnale è essenziale studiare le specifiche di gara. La linea nera ha spessore 2.5 mm e il circuito di gara è formato da uno sfondo bianco di ampiezza costante pari a 600 mm. In figura 6.2.5 sono rappresentate le dimensioni ufficiali della linea di partenza.

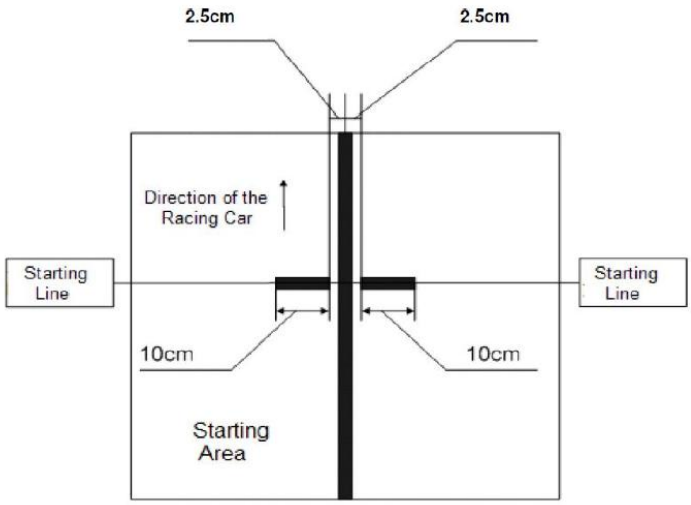


Fig.6.2.5 – Dettaglio della linea di partenza

A questo punto, note le specifiche di gara, si può procedere con la interpretazione del segnale. Tra i diversi metodi esistenti, quelli più utilizzati sono:

- Analisi del segnale con l'uso di un Threshold numerico
- Analisi del segnale grazie alle sue derivate

Il primo metodo consiste nel calcolare ad ogni ciclo di programma la media tra i vari elementi che compongono l'array *result* che contiene tutti i valori che rappresentano le diverse gradazioni di nero acquisite dalla camera. Il risultato ottenuto prende il nome di *valore di Threshold.* A questo punto si scansiona nuovamente l'array e si assegna l'etichetta "nero" a tutti gli elementi che hanno un valore minore del Threshold e viceversa.

Fig.6.2.5 – Elaborazione segnale con il metodo di Threshold

Il principale vantaggio di questo metodo è la semplicità. Infatti dal punto di vista della programmazione sono necessari solo due *cicli for* e alla fine del metodo si ottiene un'onda rettangolare perfetta di facile elaborazione. Anche dal punto di vista della potenza di calcolo necessaria risulta molto vantaggioso. Il principale svantaggio è la

sensibilità ai possibili disturbi che possono compromettere il calcolo del Threshold (ombre, variazioni di luce, disturbi elettromagnetici generati dai motori di trazione).
IL secondo metodo, invece, consiste nel calcolare la derivata in tutti i punti dell'array acquisito. Si osserva che i bordi della linea nera coincidono con i punti a derivata più grande in valore assoluto. Una volta individuata la linea da seguire si possono fare molte considerazioni. Nel caso in esame è stato preferito fare un'ulteriore semplificazione identificando un solo punto da seguire(centro linea). Questo approccio filtra eventuali disturbi di segnale a condizione di avere un'illuminazione costante. Infatti grazie all'utilizzo della scheda Led dedicata è stato possibile ottenere un buon contrasto bianco/nero della linea, una caratteristica identificabile facilmente al livello software, garantendo così un confronto ad ogni ciclo di programma della linea individuata con le specifiche della linea di gara. In questo modo possono essere implementate delle funzioni che riconoscono quando il veicolo va fuori dal circuito, contenenti una sequenza di istruzioni in grado di arrestare il veicolo in caso di emergenza. Di seguito verrà illustrato il codice per il riconoscimento del centro linea della telecamera alta. Prima di tutto il metodo prevede un taglio iniziale dell'array. Si eliminano a monte i primi e gli ultimi 10 pixel perché considerati poco attendibili. Un secondo filtraggio consiste nel focalizzare la ricerca a partire dall'ultimo centro trovato e in una banda di larghezza +/- 35 pixel attorno a tale punto. Il resto dell'array non viene preso in considerazione perché contiene informazioni poco interessanti per lo scopo "line following". Dopo le prime due operazioni di taglio si effettua la ricerca delle posizioni a gradiente più elevato in modulo.

```
void trovaCentroAlto(int* nonCiVedo,int* fineGara)
{
    double deltaY, deltaX, coefficiente, posPositivo, posNegativo;
    int media = 0;
    double coeffPositivo = 10;
    double coeffNegativo = -10;
    int fineGaraSx=0;
    int fineGaraDx=0;
    int centraleProvvisorio;
    int j1;


    int limiteSxRicerca = centraleAlto-35;
    int limiteDxRicerca = centraleAlto+35;


    if(limiteSxRicerca < 10)
        limiteSxRicerca = 10;
    if(limiteDxRicerca > 117)
        limiteDxRicerca = 117;


    for(i=limiteSxRicerca; i<limiteDxRicerca; i++)
    {
        deltaY = Result[i+3]-Result[i];
        deltaX = 4;
        coefficiente = deltaY/deltaX;
        if(coefficiente > coeffPositivo)
        {
            coeffPositivo = coefficiente;
            posPositivo = i+1;
        }
        else if(coefficiente < coeffNegativo)
        {
            coeffNegativo = coefficiente;
            posNegativo = i;
        }
        media = media + Result[i];
    }
    media = media/80;
```

Dopo che è stato trovato il centro della linea, esso viene denominano *provvisorio* e verrà considerato idoneo come *centro linea effettivo* solo dopo aver superato con successo i controlli sulla larghezza della linea e sul livello di nero trovato. In caso negativo viene alzato un Flag denominato “non ci vedo” che verrà gestito in seguito nel main del programma. Un ulteriore controllo viene effettuato per il riconoscimento della linea di fine gara. Per la telecamera bassa è stato utilizzato un algoritmo analogo.

```
centraleProvvisorio = posNegativo + ((posPositivo-posNegativo)/2)+1;

if(centraleProvvisorio > limiteDxRicerca || centraleProvvisorio <limiteSxRicerca
|| Result[centraleProvvisorio]>media|| coeffPositivo<=10 || coeffNegativo >= -10
|| (posPositivo-posNegativo)>larghezzaLineaAlto)
{
    *nonCiVedo = 1;
}


 else
 {
     *nonCiVedo = 0;
     centraleAlto = centraleProvvisorio;


     for(j1=(coeffNegativo - 2); j1> (coeffNegativo - 14); j1--)
         {
             if(Result[j1]<media && Result[j1+5]>media)
                 fineGaraSx = 1;
         }
     for(j1=(coeffPositivo +2); j1< (coeffPositivo +14); j1++)
         {
             if(Result[j1]<media && Result[j1-5]>media)
                 fineGaraDx = 1;
         }

     if(fineGaraSx == 1 && fineGaraDx == 1)
         *fineGara = 1;

 }}
```

6.2.4. Istruzioni in linguaggio C per acquisire un segnale

Come accennato nei paragrafi precedenti per comandare la camera occorre fornire come input il segnale di Clock e il segnale SI (start integration). Si attivano le porte PB[11] e PB[13] e si impostano come output dell’eMIOS. Si noti come i due segnali siano sfasati di un delay pari a mezzo periodo di oscillazione.

```
void CAMERA(void)
{


        SIU.PCR[27].R = 0x0200;             /* Program the Sensor read start pin as output*/
        SIU.PCR[29].R = 0x0200;             /* Program the Sensor Clock pin as output*/
        SIU.PGPDO[0].R &= ~0x00000014;      /* All port line low */
        SIU.PGPDO[0].R |= 0x00000010;       /* Sensor read start High */
        Delay();
        SIU.PGPDO[0].R |= 0x00000004;       /* Sensor Clock High */
        Delay();
        SIU.PGPDO[0].R &= ~0x00000010;      /* Sensor read start Low */
        Delay();
        SIU.PGPDO[0].R &= ~0x00000004;      /* Sensor Clock Low */
        Delay();
```

Supponiamo che l’ADC sia già stato inizializzato secondo le specifiche descritte nel paragrafo 2.2. A questo punto si può far partire l’acquisizione dell’ADC, in modalità di funzionamento one shot. Il ciclo for si occupa della generazione del segnale di clock e contestualmente dell’acquisizione digitale del segnale A0, il valore numerico in uscita dall’ADC, inteso in rapporto al numero di divisioni del range di ingresso all’ADC (N =

2^B) viene temporaneamente salvato in una variabile *adcdata* per essere successivamente salvato come elemento di un'array di 128 interi (ResultSotto[ ]).

```
for (i=0;i<128;i++)
{
    Delay();
    SIU.PGPDO[0].R |= 0x00000004;    /* Sensor Clock High */
    ADC.MCR.B.NSTART=1;              /* Trigger normal conversions for ADC0 */
    while (ADC.MSR.B.NSTART == 1) {};
    adcdata = ADC.CDR[6].B.CDATA;
    adcdataSopra = ADC.CDR[4].B.CDATA;
    Delay();
    SIU.PGPDO[0].R &= ~0x00000004;  /* Sensor Clock Low */
    Result[i] = (uint8_t)(adcdata >> 2);
    ResultSotto[i] = (uint8_t)(adcdata >> 2);
}
Delaycamera();}
```

CAPITOLO 7

Il controllo in retroazione

7.1. Introduzione

In questo capitolo è descritto in dettaglio il tipo di controllo implementato sul veicolo. Dopo il primo approccio con i diversi componenti disponibili, ora è possibile implementare l'algoritmo di controllo per fare interagire al meglio i componenti e sfruttare a pieno le potenzialità della *smart car.* Si possono individuare due principale macro sistemi. Il primo riguarda il controllo di posizione, consiste in un sistema di regolazione e viene gestito interamente dal servo motore, che deve essere in grado di correggere tempestivamente l'angolo di sterzo nel caso la linea si discosti dalla posizione centrale. il secondo riguarda il controllo di velocità, consiste in un sistema di asservimento. Infatti i riferimenti di velocità variano a seconda del raggio di curvatura della traiettoria. Uno degli aspetti più interessanti e complessi sta nell'iterazione non lineare tra questi due sistemi, dove entrano in gioco numerosi parametri legati alla taratura della ciclistica del veicolo.

7.1.1. Le catene di retroazione

Per capire bene il problema del controllo di posizione è utile osservare lo schema di figura 7.1.1. E' stato preso come riferimento costante il centro della smart car, si può osservare inoltre che l'andamento del circuito è visto come un disturbo del sistema. Per semplicità sono stati omessi i disturbi che ci possono essere a monte o a valle della camera, peraltro a questo punto della trattazione possono essere considerati trascurabili. Gli altri blocchi che compongono il sistema sono già stati ampiamente descritti nei capitoli antecedenti.

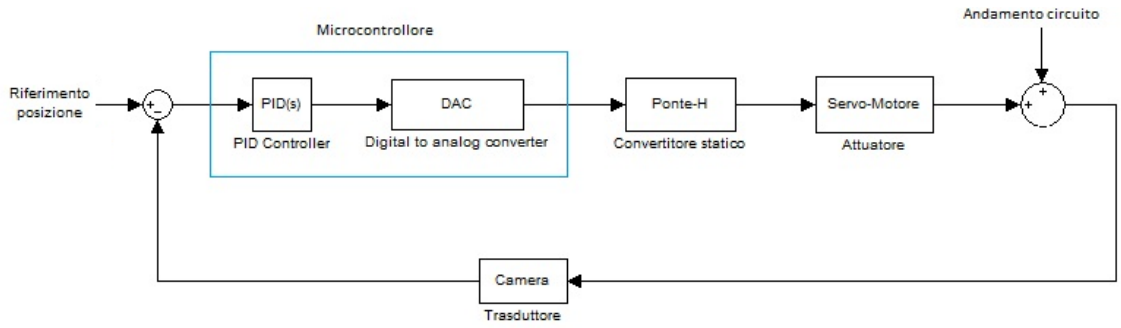


Fig. 7.1.1. – Il controllo di posizione

Volutamente non è stata inserita la seconda telecamera nello schema perché il software gestisce entrambe le camere in modo che operino in parallelo. In altre parole, grazie all'algoritmo progettato, ad ogni ciclo di programma la smart car acquisisce da entrambe le telecamere, elabora le informazioni e confronta i risultati ottenuti. Come risultato si ottiene una classificazione dello stato in cui si trova la smart car:

- Rettilineo
- Pre-Curva
- Curva

Ad ogni stato corrispondono distinti tuning dei controllori e riferimenti di velocità diversi. Solo dopo quest'ultima classificazione vengono comandati servo e motori di trazione in modo da ottenere le massime prestazioni. Per quanto riguarda il controllo di velocità si può osservare lo schema di figura 7.1.2.

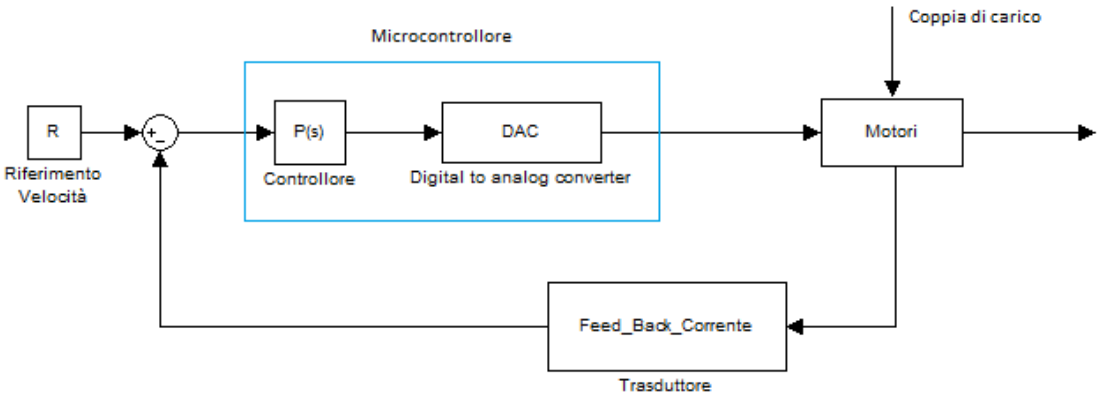


Fig. 7.1.2. – Controllo di velocità con feedback di corrente

In questo caso la retroazione è data da una misura indiretta della velocità, infatti dalla lettura della corrente di armatura del motore si ricava una stima della velocità a regime. Nei transitori infatti la corrente ha andamenti poco attendibili al fine di ricavare la velocità effettiva del veicolo. Quindi in assenza di una misura diretta della velocità, lo schema riportato sopra viene usato solo per la fase iniziale della frenatura (pre-curva). Per le altre fasi è stato adottato un controllo ad anello aperto, visibile in figura 7.1.3.

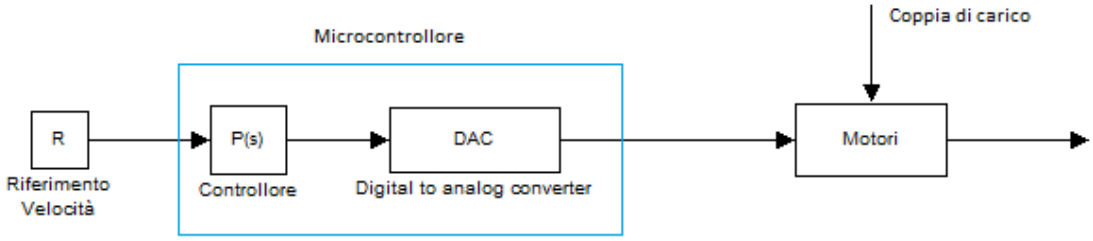


Fig. 7.1.3. – Controllo di velocità ad anello aperto

Il principale vantaggio di questo tipo di controllo in feedforward è l'assenza di sensori per la determinazione della velocità effettiva (encoder), quindi anche i possibili disturbi che si possono generare in fase di misurazione vengono eliminati. Per implementare questo tipo di controllo, è richiesta una perfetta conoscenza del sistema. A tale scopo sono state effettuate numerose misure sperimentali per ricavare il legame tra tensione applicata ai motori e velocità effettiva a regime.

7.1.1. Controllori digitali

Data l'importanza del controllo di posizione è stato necessario trovare un compromesso tra specifiche di progetto (errore a regime nullo), immediata realizzazione al livello software e semplicità nella taratura dei vari parametri. Dopo queste osservazioni la scelta migliore è caduta su un controllore di tipo PID, molto impiegato anche nel campo dell'industria grazie alla sua elevata efficacia per i sistemi di regolazione. Prima di utilizzare un controllore PID occorre fare attenzione al livello di prestazioni che offre grazie all'azione combinata di integrazione e derivazione dell'errore in ingresso. Inoltre bisogna studiare i fenomeni non lineare dovuti ai limiti fisici del servo motore. Infatti si possono verificare dei fenomeni di saturazione: il servo motore ha due fine corsa che ne limitano l'angolo di sterzo. Quando si raggiungono i limiti fisici si ha una brusca interruzione della catena di retroazione, e quindi il sistema si ritrova ad operare a catena aperta, dato che lo sterzo rimarrà costante al suo valore limite, indipendentemente dall'uscita del processo controllato. In questa situazione l'errore continuerà ad essere integrato, senza però effetti sulla cariabile di controllo. Il termine integrale può allora raggiungere valori molto elevati, situazione che in inglese viene denominata come "wind up". In altre parole quando il sistema entra in saturazione insorgeranno larghe sovraoscillazioni transitorie.

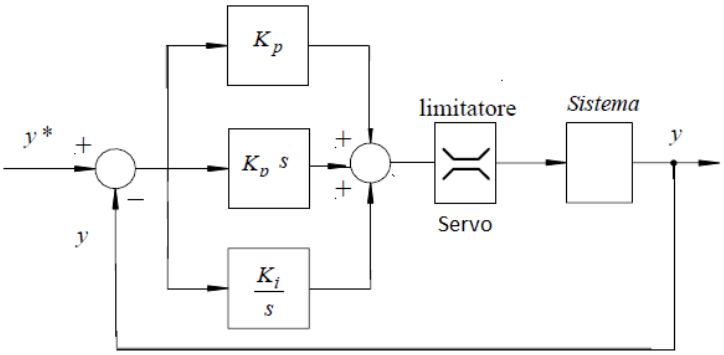


Fig. 7.1.4. – Il problema della saturazione del servo motore

Il metodo che viene di seguito proposto segue il principio di ricalcolare il valore dell'uscita della parte integrale in presenza di saturazione, in modo che il nuovo valore dia un'uscita molto vicina al limite stesso. E' conveniente evitare di resettare l'integrale istantaneamente, ma di effettuare il calcolo in modo dinamico, con una costante di tempo.

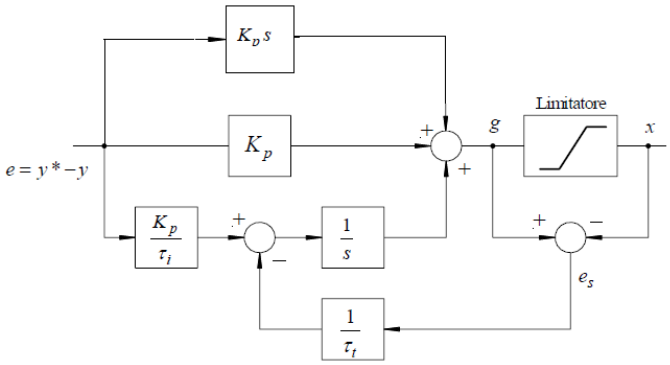


Fig. 7.1.4. – Regolatore PID con antiwindup basato su back-calculation and tracking

Gli schemi a blocchi sono riportati nel dominio delle trasformate di Laplace, naturalmente vale la stessa trattazione nel dominio discreto. Infine, per semplicità è stato scelto un regolatore proporzionale (P) per il controllo di velocità.

7.2. Implementazione e taratura del PID

7.2.1 Istruzione in linguaggio C

Come accennato nel paragrafo precedente l'implementazione nel PID è molto immediata e, dal punto di vista della potenza di calcolo, richiede pochi cicli macchina per la sua esecuzione.

```
double pidCal(int actualPosition, float Kp, float Kd, float Ki,
int* error, int* pre_error, double* integral)
{
    double outputPid = 0;
    double derivative = 0;
    *error= carPosition - actualPosition;
    *integral = *integral + (*error)*dt;
    derivative = (*error - *pre_error)/dt;
    outputPid = Kp*(*error) + Ki*(*integral) + Kd*derivative;

    if(outputPid > MAX)         /*Filtro di saturazione*/
    {

        *integral = *integral-KK(outputPid- MAX);   /*ANTI WIND UP, *integral = 1; */
        outputPid = MAX;
    }
    else if(outputPid < MIN)
    {
        outputPid = MIN;
        *integral = *integral-KK(outputPid-MIN);    /*ANTI WIND UP, *integral = -1; */
    }

    *pre_error = *error;            /*Aggiornamento errore*/
    return outputPid;
}
```

Come si può osservare è stato utilizzato un filtro di saturazione per evitare di forzare il servo a valori oltre i limiti imposti dal fine corsa. Quindi è stato introdotta anche una funzione di antiwindup per evitare la crescita incontrollata dell'integrale dell'errore. A questo punto per avere una PWM che andrà a comandare il servo motore si utilizza una semplice funzione di adattamento come quella riportata in figura 7.2.2.

```
int adattaControllo(double input)
{
    int risultato= (int)(centroServo + input*(10));
    return risultato;
}
```

7.2.2 Tuning del PID

Prima di partire con il tuning occorre precisare il significato dei diversi parametri, si prenda come riferimento la *smart car* che insegue la traiettoria:

- Errore – rappresenta la differenza tra la posizione di riferimento e la posizione misurata
- Proporzionale - misura quanto la *smart car* è lontana dalla linea.
- Integrale - misura l'errore accumulato nel tempo. Il valore dell'integrale è crescente quando la *smart car* non è centrata sulla linea. Più a lungo la *smart car* rimane lontana dal centro maggiore sarà l'integrale.
- Derivata – misura la velocità con la quale la *smart car* si sta muovendo da destra verso sinistra o viceversa.

- Fattori K_P , K_I , K_D – sono dei valori costanti utilizzati per aumentare o diminuire rispettivamente il contributo della parte proporzionale, integrale, derivata.

Uno dei vantaggi connessi all'utilizzo dei regolatori PID consiste nella possibilità di effettuare la taratura dei parametri sulla base di semplici prove sperimentali, a prescindere dalla formulazione matematica, non sempre agevole, del sistema sotto controllo. Tra i numerosi metodi empirici per la sintonizzazione dei regolatori PID, è stato utilizzato il metodo di Ziegler e Nichols. Il metodo si articola nei seguenti passi:

1. Si chiude l'anello di controllo con il regolatore PID imponendo nulle le azioni integrale e derivativa: $K_I = 0$, $K_D = 0$.
2. Partendo da valori molto piccoli di K_P si effettua un semplice esperimento, si prova la *smart car* su pista e si osserva l'andamento.
3. Si aumenta progressivamente K_P ripetendo di volta in volta l'esperimento finché non si instaura nell'anello un'oscillazione permanente.
4. Detto $\underline{K_P}$ il valore del guadagno proporzionale corrispondente all'oscillazione permanente e T il periodo di tale oscillazione, si tarano i parametri di un regolatore PID in base alla seguente tabella:

	K_P	T_I	T_D
PID	$0.6\overline{K}_P$	$\frac{\overline{T}}{2}$	$\frac{\overline{T}}{8}$

L'utilizzo di questo metodo offre un valido punto di partenza per un eventuale aggiustamento dei parametri per via esperimentale. Infatti, come accennato prima, sono stati trovati tre valori validi per ogni coefficiente K_P , K_I , K_D che vengono utilizzati dall'algoritmo di controllo attraverso una logica "di switch", in base allo stato in cui si trova il veicolo.

7.3. L'algoritmo di controllo

7.3.1. La gestione a stati

Ricapitolando, la *smart car* elabora i dati in ingresso dalle due telecamere e riconosce lo stato in cui si trova: rettilineo, pre-curva, curva. Ad ogni stato corrispondono distinti parametri del PID che gestisce il controllo di posizione e differenti riferimenti di velocità.

Rettilineo
Le condizioni necessarie affinché il veicolo si trovi in questo stato sono:

- Le due telecamere riconoscono la linea nel proprio range di visualizzazione
- I centri linea identificati differenziano tra di loro per una distanza inferiore a 10 pixel.

In questo stato il veicolo segue il centro trovato dalla camera alta. Come accennato del paragrafo precedente è stato adottato un controllo di velocità a catena aperta. La velocità di riferimento è stata impostata a 2.6 m/s (80%). Inoltre il PID utilizzato è prevalentemente proporzionale/integrale. Questo per garantire una maggiore stabilità del veicolo anche ad elevate velocità.

Pre-curva
Le condizioni da verificarsi sono le seguenti:

- Le due telecamere riconoscono la linea nel proprio range di visualizzazione
- I centri linea identificati differenziano tra di loro per una distanza maggiore a 10 pixel.

Anche in questo stato, il veicolo viene guidato dalla camera alta. I parametri del PID rimangono invariati. Grazie al feedback di corrente viene stimata la velocità attuale, e si genera una rampa di frenatura. Il veicolo dovrà raggiungere una velocità pari 1,5 m/s (45%).

Curva
La condizione necessaria e sufficiente per questo stato è:

- La telecamera alta non vede più la linea nel proprio range di visualizzazione.

In questo stato, invece, il veicolo segue il centro trovato dalla camera bassa. Infatti quella alta,anche se acquisisce ad ogni ciclo di programma, i dati individuati risultano poco attendibili al fine di seguire la traiettoria. La camera bassa garantisce una elevata qualità dei dati acquisiti grazie alle scelte nel suo posizionamento. In questo stato la velocità è controllata ad anello aperto con un riferimento costante pari 1,5 m/s. I parametri del PID utilizzati sono prevalentemente proporzionali/derivati, per garantire elevate prestazioni allo sterzo in presenza di notevoli variazioni nel tempo del centro linea da seguire.

A titolo di esempio di seguito viene riportato una parte del main del programma che gestisce questi tre stati:

```
FORWARD(pwmMotorLeft,pwmMotorRight);
CAMERA();
trovaCentroBasso(&nonCiVedoBasso,&fineGaraSotto);
erroreBasso= carPosition-centraleBasso;

trovaCentroAlto(&nonCiVedoAlto, &fineGaraSopra);
erroreAlto = carPosition-centraleAlto;

aggiornaArray(centraleAlto,arrayMobileAlto,&vecchioAlto);
aggiornaArray(centraleBasso,arrayMobileBasso,&vecchioBasso);

/*rettilineo*/
if(nonCiVedoAlto==0 && abs(erroreAlto)<=SOGLIA_PRE_CURVA)
{
dritto(&Kp,&Kd,&Ki,&centroDaSeguire,centraleAlto,&pwmMotorLeft,
&pwmMotorRight,contatoreRettilineo,&contatoreAccelerazione);
}

/* Pre Curva*/
if(abs(erroreAlto)>=SOGLIA_PRE_CURVA && nonCiVedoAlto==0)
{
sterzata(&Kp,&Kd,&Ki,kpPulsante,kdPulsante,kiPulsante,&centroDaSeguire,
centraleBasso,fermata,frenata,&pwmMotorLeft,&pwmMotorRight,&contatoreRettilineo,
&contatoreAccelerazione);
differenziale(controlloServo,&pwmMotorLeft,&pwmMotorRight);
}

/*In Curva*/
if(nonCiVedoAlto==1)
{
sterzata(&Kp,&Kd,&Ki,kpPulsante,kdPulsante,kiPulsante,&centroDaSeguire,
centraleBasso,fermata,frenata,&pwmMotorLeft,&pwmMotorRight,&contatoreRettilineo,
&contatoreAccelerazione);
differenziale(controlloServo,&pwmMotorLeft,&pwmMotorRight);
}
```

E’ stato previsto un ulteriore stato di “emergenza” che si verifica quando la camera bassa perde la linea da seguire. In questo caso viene mantenuto in memoria l’ultimo punto individuato. Come risultato la *smart car* manterrà la stessa traiettoria. Se, dopo un certo un certo tempo (DELAY), nessuna delle due telecamere riesce ad individuare la linea, allora il programma si arresta e si bloccano i motori di trazione.

```
/*Se neanche con la telecamera bassa non vede niente tieni
il riferimento calcolato per ultimo */
if(nonCiVedoBasso==0 || nonCiVedoAlto ==0)
{
controlloServo=adattaControllo(pidCal(centroDaSeguire, Kp,Kd,Ki ,
&error,&pre_error, &integral));
}


/* Controllo se entrambe le telecamere non vedono più niente e
inizia il countdown per la fermata segnalandolo con il LED_1 */

if(nonCiVedoBasso==1 && nonCiVedoAlto==1)
{
fermata=fermata-1;
SIU.PGPDO[2].R &= 0x07000000;       /* Enable LED1*/
}

else
/* Non appena una delle telecamere vedono qualcosa
esco dalla modalità countdown e resettando il contatore
e spegnendo il LED_1 */
{
fermata= DELAY;
SIU.PGPDO[2].R |= 0x08000000;       /* Disable LED1*/
}

/*filtro saturazione servo*/
if(controlloServo>leftServo)
controlloServo=leftServo;
if(controlloServo<rightServo)
controlloServo=rightServo;

EMIOS_0.CH[4].CBDR.R = controlloServo;
}}
```

7.3.2. Interfaccia Hardware-software

Per completare la trattazione occorre parlare del software messo a disposizione dalla Freescale Semiconductors. Il primo è RAppID, uno strumento di sviluppo grafico per la famiglia di microcontrollori con l’architettura MPC5xxx che permette all'utente di configurare facilmente e rapidamente il microcontrollore e generare la documentazione completa. RAppID non solo genera codice C per l'inizializzazione dei registri, ma fornisce anche una funzione di inizializzazione del sistema che imposta il controllore ad eseguire una sequenza ordinata di istruzioni in fase di avvio. E’ da sottolineare che RAppID offre un metodo veloce per configurare il software solo dei microcontrollori costruiti secondo l’architettura MPC5xxx. E’ da considerarsi quindi un software altamente dedicato. L’altro strumento a disposizione è CodeWarrior (versione 5.9.0), un software di tipo IDE (integrated development environment) e permette la creazione di programmi compatibili con molti sistemi embedded. Il principale vantaggio è la elevata flessibilità per la programmazione, infatti permette di scrivere un software in linguaggio C anche a basso livello. Oltre a quella usata sono disponibili diverse versioni di CodeWarrior compatibili con una vasta gamma di architetture hardware e linguaggi di programmazione.

Conclusioni

Con questo lavoro si è voluto offrire uno strumento in più agli studenti che intraprendono la competizione *Freescale Cup*, in particolare l'obiettivo prefissato era quello di illustrare in modo semplice ma completo le potenzialità dei componenti hardware e introdurre il lettore al primo approccio nella progettazione di un algoritmo di controllo per la gestione di una *smart car*. Inoltre si è voluto fornire degli esempi chiari e mirati allo scopo "line following" attraverso il software messo a disposizione da *Freescale (Code Warrior),* implementando così un controllo di base del veicolo. Sono stati illustrati diversi metodi per migliorare le prestazioni del sistema. Il presente documento si pone quindi ad un livello più alto rispetto alla letteratura disponibile in rete e ai manuali. Si sottolinea l'importanza del controllo di velocità che in questo documento è stato descritto in maniera minima per mancanza di un sistema di sensing dedicato (encoder), ma fondamentale per avere una gestione a 360° del circuito di gara. Per ulteriori chiarimenti e informazioni tecniche più dettagliate si rimanda alla letteratura di riferimento.

Bibliografia

1. Freescale University Programs (https://community.freescale.com/community/uvp)

2. T. N. Blalock and R. C. Jaeger, *Microelectronic Circuit Design,* McGraw-Hill, 3rd ed., 2008

3. Freescale Semiconductor, *MC33931 Data Sheet*, Rev. 3.0, June 2012

4. Freescale Semiconductor*, MPC5604B Application Note: Using Parallax TSL1401-DB Linescan Camera Module for line detection*, Rev. 0, January 2011

5. Freescale Semiconductor, *MPC5604B/C Microcontroller Reference Manual*, Rev. 8.1, May 2012

6. Taos, *TSL1401CL 128x1 linear sensor array with hold,* July 2011

7. M. Zigliotto, *Dispense dal corso di Fondamenti di macchine ed azionamenti elettrici*, 2012

8. M. Bertocco and A. Sona, *Introduzione alle misure elettroniche*, Lulu, 2nd ed., 2010

9. Futaba (http://www.futabarc.com)

10. Standard Motors (http://www.standartmotor.net)

11. F. R. Ramirez, M. Trujillo, R. Mendoza, *Linescan Camera Module for line detection*

12. CodeWarrior Development Studio IDE 5.9 User's Guide - july 2010

13. Steve Mihalik, *Getting Started with MPC560xB, Freescale Cup,* January 24, 2011