Robotic Sorter

Filippo Passarella, Ubaldo Tosi

24 dicembre 2024

Corso di studi di Ingegneria elettronica

0.1 Introduzione

0.2 Sensore di colore

Il PCB del sensore di colore è costituito dal sensore di colore vero e proprio (TAOS TCS3472), un led bianco che permette di illuminare gli oggetti per misurare il colore della luce riflessa, spegnibile nel caso in cui si voglia misurare il colore di uno schermo o comunque qualcosa che emette luce propria, un regolatore $5V \rightarrow 3.3V$ che ci permette di alimentare il sensore a 5V e di usare la I2C a 5V per comunicare con il sensore grazie a due transistori messi come in figura. Inoltre integra le resistenze di pull-up.

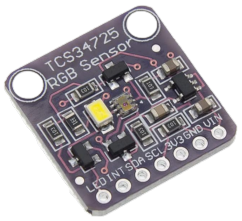


Figura 1: PCB Sensore

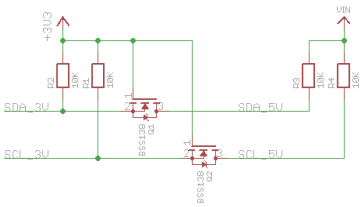
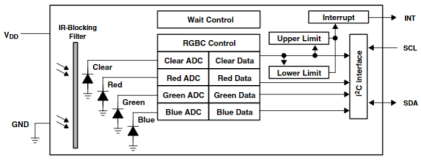
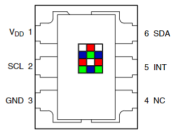


Figura 2: I2C

0.2.1 TAOS TCS3472



(a) Blocco funzionale



(b) Matrice fotodiodi sensore

Il sensore è costituito da una matrice 3x4 di fotodiodi con un filtro rosso, blu, verde e non filtrati (luce bianca), tutti i fotodiodi hanno un filtro ad infrarossi per aumentare l'accuratezza. La corrente generata da questi fotodiodi viene amplificata da un amplificatore trans-resistivo a guadagno variabile e campionata da un ADC integrativo a 16 bit per ogni colore. Il tempo di integrazione è programmabile. I dati poi vengono comunicati all'esterno attraverso l'interfaccia I2C (fino a 400 KHz). Il sensore può mandare un interrupt una volta che il valore del clear è sopra o sotto una certa threshold.

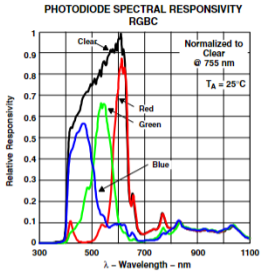


Figura 4: Responsività del sensore

Come si può vedere dall'immagine il sensore risponde molto bene al rosso e la responsività diminuisce andando verso il verde ed il blu questo come potremo vedere più avanti, unito al fatto che la luce del led tende al giallo porterà a problemi nella misura.

I2C

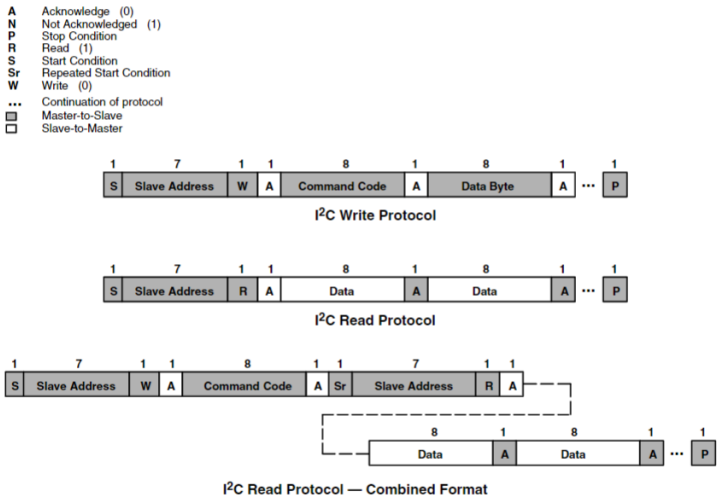


Figura 5: I2C

Come possiamo vedere la scrittura dei registri mediante I2C viene fatta mandando per prima cosa lo start bit, poi l'indirizzo della periferica e il write. Una volta ricevuto l'acknowledge si manda il Command code. Questo configura il registro di Command. Nel registro di Command possiamo stabilire se far leggere sempre lo stesso registro o scorrerli e da che indirizzo iniziare. Inoltre possiamo pulire gli interrupt. Per la lettura una volta settato correttamente il registro command basta mandare lo start, l'indirizzo e il read e lui

Table 4. Command Register

	7	6	5	4	3	2	1	0	
COMMAND	CMD	TYPE		ADDR/SF					--

FIELD	BITS	DESCRIPTION
CMD	7	Select Command Register. Must write as 1 when addressing COMMAND register.
TYPE	6:5	Selects type of transaction to follow in subsequent data transfers:
		FIELD VALUE / INTEGRATION TIME
		00 — Repeated byte protocol transaction
		01 — Auto-increment protocol transaction
		10 — Reserved — Do not use
		11 — Special function — See description below
		Byte protocol will repeatedly read the same register with each data access. Block protocol will provide auto-increment function to read successive bytes.
ADDR/SF	4:0	Address field/special function field. Depending on the transaction type, see above, this field either specifies a special function command or selects the specific control-status-data register for subsequent read and write transactions. The field values listed below only apply to special function commands:
		FIELD VALUE / READ VALUE
		00110 — Clear channel interrupt clear
		other — Reserved — Do not write
		The Clear channel interrupt clear special function clears any pending interrupt and is self-clearing.

Figura 6: Registro Command

continuerà dopo ogni ack a mandare il dato successivo fino a che non riceve lo stop bit.

Macchina a Stati sensore

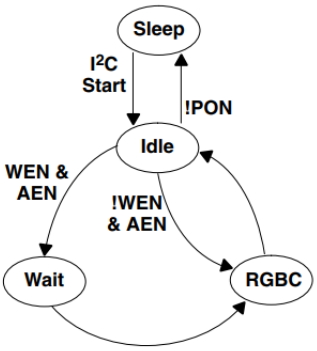


Figure 6. Simplified State Diagram

Figura 7: Macchina a stati

Il sensore per risparmiare energia rimane nello stato di sleep, una volta che riceve lo start dall'I2C va in idle ed in base a se è settato il bit di wait o meno fa subito la misura oppure aspetta un certo tempo. Una volta fatta la misura torna in idle e in caso di power on disabilitato torna in sleep. La macchina a stati dipende principalmente dai parametri del registro Enable (0x00).

Registri

Table 3. Register Address

ADDRESS	RESISTER NAME	R/W	REGISTER FUNCTION	RESET VALUE
—	COMMAND	W	Specifies register address	0x00
0x00	ENABLE	R/W	Enables states and interrupts	0x00
0x01	ATIME	R/W	RGBC time	0xFF
0x03	WTIME	R/W	Wait time	0xFF
0x04	AILTL	R/W	Clear interrupt low threshold low byte	0x00
0x05	AILTH	R/W	Clear interrupt low threshold high byte	0x00
0x06	AIHTL	R/W	Clear interrupt high threshold low byte	0x00
0x07	AIHTH	R/W	Clear interrupt high threshold high byte	0x00
0x0C	PERS	R/W	Interrupt persistence filter	0x00
0x0D	CONFIG	R/W	Configuration	0x00
0x0F	CONTROL	R/W	Control	0x00
0x12	ID	R	Device ID	ID
0x13	STATUS	R	Device status	0x00
0x14	CDATAL	R	Clear data low byte	0x00
0x15	CDATAH	R	Clear data high byte	0x00
0x16	RDATAL	R	Red data low byte	0x00
0x17	RDATAH	R	Red data high byte	0x00
0x18	GDATAL	R	Green data low byte	0x00
0x19	GDATAH	R	Green data high byte	0x00
0x1A	BDATAL	R	Blue data low byte	0x00
0x1B	BDATAH	R	Blue data high byte	0x00

Figura 8: Registri sensore

Parametri scelti

Indirizzo	Nome	Valore
—-	Command	10100000
0x00	Enable	0x03
0x01	Timing	0xC0
0x0F	Control	0x03

Tabella 1: Indirizzi settati

Solo questi registri sono stati settati in quanto si è scelto di non utilizzare l'interrupt per decidere quando leggere il sensore, ma piuttosto di fare un polling alla pressione di un pulsante ed eseguire a quel punto 3 misure così da implementare un voter. Questo perché i livelli di luminosità registrati dal sensore erano simili tra con il cubetto sopra e senza e variavano più in base al colore del cubetto che dal fatto che fosse sopra o no.

Command

Come visibile nella figura 6 per fare la lettura e scrittura abbiamo deciso di usare registro di Enable come primo registro e di fare le letture sequenziali degli indirizzi.

Enable

Enable Register (0x00)

The Enable register is used primarily to power the TCS3472 device on and off, and enable functions and interrupts as shown in Table 5.

Table 5. Enable Register

	7	6	5	4	3	2	1	0	
ENABLE	Reserved			SAIEN	WEN	Reserved	AEN	PON	Address 0x00

FIELD	BITS	DESCRIPTION
Reserved	7:5	Reserved. Write as 0.
AIEN	4	RGBC interrupt enable. When asserted, permits RGBC interrupts to be generated.
WEN	3	Wait enable. This bit activates the wait feature. Writing a 1 activates the wait timer. Writing a 0 disables the wait timer.
Reserved	2	Reserved. Write as 0.
AEN	1	RGBC enable. This bit actives the two-channel ADC. Writing a 1 activates the RGBC. Writing a 0 disables the RGBC.
PON [1, 2]	0	Power ON. This bit activates the internal oscillator to permit the timers and ADC channels to operate. Writing a 1 activates the oscillator. Writing a 0 disables the oscillator.

Figura 9: Enable Register

Come scritto nella sezione 0.2.1 abbiamo ovviamente attivato il power on e gli ADC mentre non abbiamo attivato gli interrupt e il wait. Non è stato attivato il wait in quanto per la nostra applicazione abbiamo preferito dare priorità alla velocità di lettura che al power management tenendo anche conto che si fanno 3 letture del sensore ogni volta.

Timing e Control

RGBC Timing Register (0x01)

The RGBC timing register controls the internal integration time of the RGBC clear and IR channel ADCs in 2.4-ms increments. Max RGBC Count = (256 – ATIME) × 1024 up to a maximum of 65535.

Table 6. RGBC Timing Register

FIELD	BITS	DESCRIPTION			
ATIME	7:0	VALUE	INTEG_CYCLES	TIME	MAX COUNT
		0xFF	1	2.4 ms	1024
		0xF6	10	24 ms	10240
		0xD5	42	101 ms	43008
		0xC0	64	154 ms	65535
		0x00	256	700 ms	65535

Figura 10: Registro timing

Table 11. Control Register

	7	6	5	4	3	2	1	0	
CONTROL	Reserved						AGAIN		Address 0x0F

FIELD	BITS	DESCRIPTION	
Reserved	7:2	Reserved. Write bits as 0	
AGAIN	1:0	RGBC Gain Control.	
		FIELD VALUE	RGBC GAIN VALUE
		00	1× gain
		01	4× gain
		10	16× gain
		11	60× gain

Figura 11: Registro control

I valori dei registri di timing e control sono stati scelti in tandem in quanto sono i valori che influenzano la lettura e la sua accuratezza. Il registro di timing modifica il tempo di integrazione degli ADC e quindi influi-

sce sulla sensibilità e sulla risoluzione della misura mentre quello di control determina il guadagno dell'amplificatore. Inizialmente per il timing ci siamo imposti un tempo massimo per le 3 misure di 1,5 secondi. Sapendo che $Timing = (0xFF - 0xATIME) * 2.4ms$ Possiamo calcolare il $ATIME_{min} = 0x2A$. Inizialmente abbiamo provato con il massimo guadagno (x60) e abbiamo aumentato il tempo di integrazione fino a raggiungere dei valori con un buon valore massimo e minimo con tutti i colori evitando la saturazione ovvero $ATIME_1 = 0xC0$ ovvero Timing = 154ms così da avere un buon compromesso tra il tempo di integrazione e la risoluzione e sensibilità dei valori misurati. Successivamente abbiamo fatto una prova usando il massimo tempo di integrazione considerato accettabile ($ATIME_2 = 0x2A$ ovvero Timing = 510ms) ed aumentato il guadagno fino ad arrivare ad un valore né troppo basso né in saturazione per ogni colore, ovvero (x16).

Registro	Configurazione 1	Configurazione 2
Timing	0xC0	0x2A
Control	0x03	0x02

Tabella 2: Configurazioni

Registro	Configurazione 1	Configurazione 2
Red_L	01	6A
Red_H	FF	D6
$Green_L$	31	E4
$Green_H$	2F	26
$Blue_L$	62	97
$Blue_H$	25	1F

Tabella 3: Cubo rosso

Registro	Configurazione 1	Configurazione 2
Red_L	2D	E4
Red_H	49	43
$Green_L$	03	2B
$Green_H$	80	79
$Blue_L$	77	F4
$Blue_H$	6C	66

Tabella 4: Cubo blu

Vedendo che la differenza tra le due configurazioni era trascurabile si è op-

Registro	Configurazione 1	Configurazione 2
Red_L	79	48
Red_H	5A	55
$Green_L$	CD	89
$Green_H$	80	7C
$Blue_L$	9A	6C
$Blue_H$	3D	38

Tabella 5: Cubo verde

tato per l'uso della prima così da risparmiare un secondo nella misura. Un problema riscontrato è stato quello della misura sul cubo blu in quanto il colore dal valore più alto è risultato il verde ed il blu solo il secondo più alto. Provando a cambiare colore con un cartoncino più scuro il colore con il valore più alto risultava invece il rosso a causa del grande assorbimento di luce del cartoncino blu scuro. Questo è dovuto in parte al fatto che la luce del led in dotazione tende un po' verso il giallo (componente rossa più accentuata) e alla maggior responsività alla luce rossa e verde rispetto alla blu.

0.2.2 Lettura

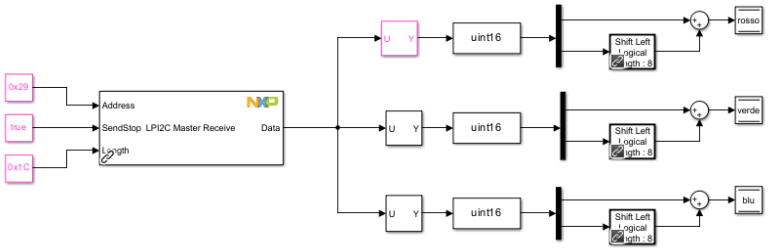


Figura 12: lettura sensore

Si è deciso di leggere tutti i blocchi ogni volta che fosse stata necessaria la lettura per evitare dopo la scrittura di dover riscrivere il registro command con il nuovo indirizzo di partenza visto che il guadagno in termini di tempo era trascurabile. Una volta eseguita la lettura di esegue la somma pesata dei valori h e l dei registri di ogni colore per avere il valore effettivo a 16 bit.

0.2.3 Macchina a stati

Come possiamo vedere nella figura 15 all'accensione viene configurato il sensore e messo colore a 4 per mettere il braccio ad una posizione di default

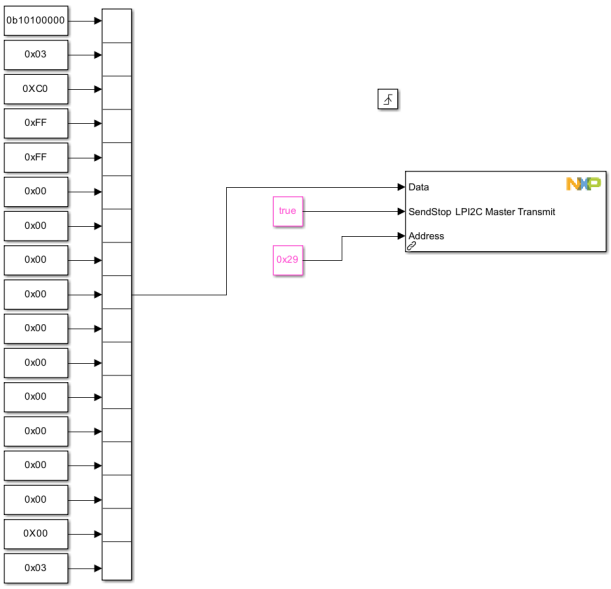


Figura 13: scrittura sensore

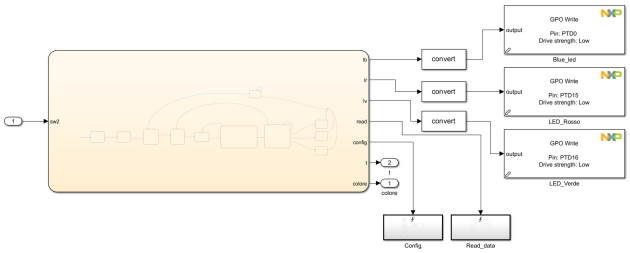


Figura 14: Matlab sensing

Successivamente si entra in uno stato di wait in cui si resettano tutti i contatori. Premendo il pulsante sw2 si passa allo stato di reset del voter. Successivamente si eseguono 3 letture con conseguente decisione del colore per ogni misura. A quel punto il voter decide il colore del cubo se vi sono 2 colori uguali oppure rifà le misure se sono tutti e 3 diversi. Per finire incrementa un contatore ogni 200ms per fare l'attuazione.

Voter

Per stabilire con accuratezza il colore del cubo è stato necessario implementare un voter in quanto durante i test la misura a volte risultava diversa per cui si è deciso di applicare una ridondanza nelle misure a scapito della velocità di misura. Perle ragioni esposte nella sezione 0.2.1 nel blocco color detection è

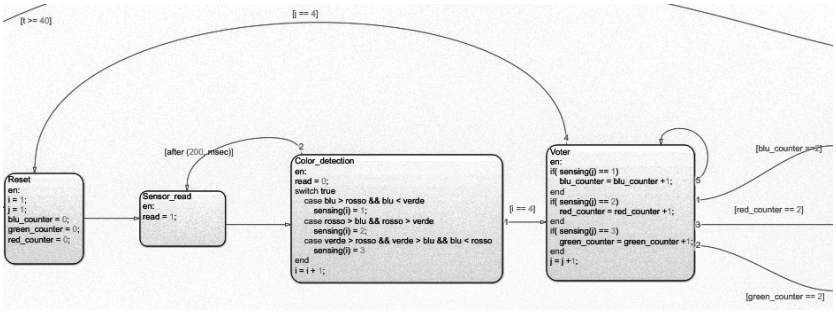


Figura 15: Macchina a stati

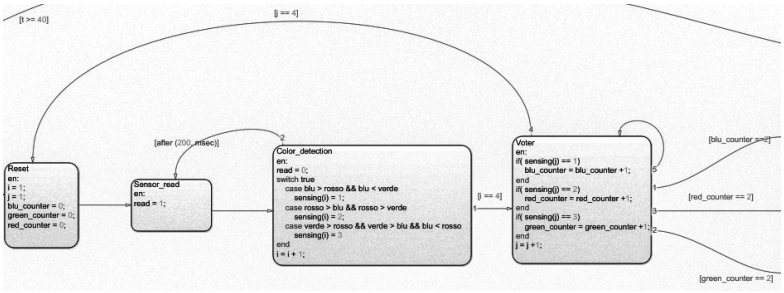


Figura 16: Voter

stato necessario per il riconoscimento del blu mettere che blu fosse compreso tra il verde e il rosso.

0.2.4 Test

Purtroppo non è stato possibile fare il PIL dei blocchi contenenti la comunicazione via I2C in quanto Matlab dava errore in caso di presenza dei blocchi di send e receive via I2C. Per eseguire i test della parte di sensore inizialmente

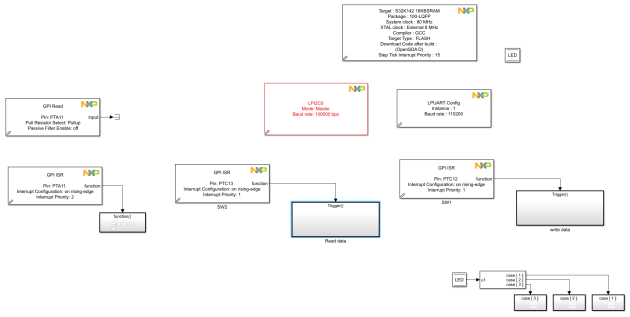


Figura 17: Test I2C

è stato fatto un file Matlab che si occupava solo di comunicare con il sensore

e mandare i dati ricevuti via uart al pc così da poter valutare i parametri migliori per il sensore. Una volta individuati è stata fatta la macchina a stati e testata prima in MIL e poi usando la uart e il led rgb per il debug. Una

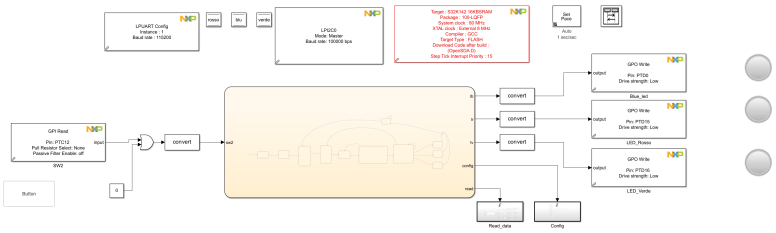


Figura 18: Test macchina a stati

volta controllato che tutto funzionasse a dovere è stato unito alla parte di attuazione ed è stato testato il tutto prima in modalità MIL e poi usando la uart per il debug. Una volta risolti eventuali problemi abbiamo attaccato il braccio e testato il comportamento fisico ottimizzando i path.

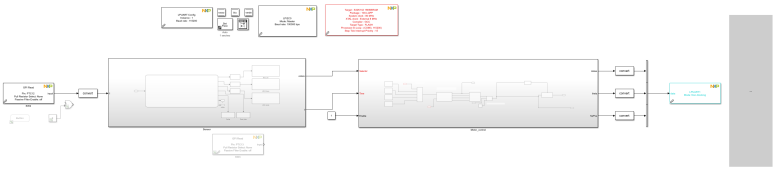


Figura 19: MIL completo

0.2.5 Miglioramenti

1. Trovare cartoncini blu tali per cui il valore del blu il maggiore rispetto agli altri
2. Spegnere il led del sensore quando non si sta facendo la misura per ridurre il consumo di energia

0.3 Controllo motori

0.4 Conclusioni