UNIVERSITÀ DEGLI STUDI DEL SANNIO

Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria
Elettronica per l'Automazione e le Telecomunicazioni

Corso di

Ingegneria del Software

Software Requirement Specification

"Recognizer_ Colors"

Docente :

Ch.mo prof. Massimiliano Di Penta

Studenti :

Nicola DI VICO 397/103

Raffaele LUISI 397/115

Rosa CAPOBIANCO 397/105

Anno Accademico 2015-2016

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 1

Introduzione

1.1 Obiettivo

La stesura di tale documento ha come obiettivo quello di illustrare i requisiti richiesti e l'architettura hardware e software di un dispositivo di riconoscimento di colori. Tale dispositivo è realizzato con l'idea di facilitare in qualche modo la percezione dei colori degli oggetti che ci circondano di coloro che abbiano difficoltà visive, in particolare non vedenti e/o daltonici.

1.2 Scopo

Il dispositivo ***Recognizer_ Colors*** deve essere in grado di identificare il colore dell'oggetto in prossimità del quale viene posto, riuscendo inoltre ad associare al colore identificato, una segnalazione vocale che metta a conoscenza all'utente del colore dell'oggetto in questione.

1.3 Struttura generale

Il documento consta di quattro capitoli: nel capitolo 2 viene effettuata una descrizione delle funzionalità del sistema, elencando anche quelli che sono i requisiti funzionali e non funzionali del sistema. Il capitolo 3 invece, descrive le funzionalità del dispositivo sotto forma di casi d'uso. Inoltre viene riportata una descrizione dell'architettura sia hardware che software del sistema. Infine, il capitolo 4 riporta le tecniche di testing utilizzate.

Capitolo 2

Requisiti del sistema

2.1 Vincoli

Di seguito sono riportati i vincoli del sistema:

• L'utente può mettere in funzione il dispositivo semplicemente pigiando il pulsante di cui è fornito.

• Il dispositivo è provvisto di una microSD per la memorizzazione dei file audio di una capienza pari a 1 GB.

• Il modulo che permette l'utilizzo della microSD non supporta memorie superiori a 2 GB.

• La distanza tra l'oggetto ed il dispositivo non deve essere superiore a 10 mm.

• Ambienti di sviluppo: *IDE* di Arduino.

2.2 Requisiti funzionali

Sono stati fissati i seguenti requisiti funzionali:

• Il dispositivo deve disporre di un pulsante che permetta l'esecuzione del sistema.

• Il dispositivo deve permettere di rilevare tramite modulo sensore di riconoscimento di colori, i valori di intensità luminosa di Rosso, Verde e Blu.

• Il dispositivo deve disporre di un modulo di memoria dei file audio necessario ad associare al colore rilevato la segnalazione acustica corrispondente.

• Il dispositivo deve disporre di un led che permetta di segnalare se il sistema è o meno in funzione.

2.3 Requisiti non funzionali

Sono stati fissati i seguenti requisiti non funzionali:

2.3.1 Documentazione

• Piano di Software Configuration Management, secondo lo standard IEEE 828.

• Analisi dettagliata dei requisiti.

• Source Code.

2.4 Interfaccia utente

Il dispositivo è dotato di un pulsante che permette, una volta avvicinato il sensore di riconoscimento di colori all'oggetto in questione, di rilevare il colore del suddetto oggetto, quindi inviare il segnale al sistema che sarà in grado di trasmettere il file audio corrispondente. Un led resta acceso in condizioni di standby del sistema mentre si spegne durante la segnalazione acustica.

2.5 Caratteristiche utente

Gli utenti destinatari del prodotto ***Recognizer_ Colors*** sono coloro i quali presentino difficoltà visive e non riescano a distinguere i colori. Potrebbero inoltre farne uso tutti colori i quali volessero riconoscere il colore dell'oggetto in riferimento.

2.5.1 Ambiente fisico di operatività

Il sensore può operare in qualsiasi ambiente fisico, però la corretta identificazione dei colori è garantita solo per oggetti con determinati vincoli di riflettività.

Capitolo 3

Modellazione del sistema

3.1 Specifiche dei casi d'uso

I casi d'uso sono impiegati per descrivere il comportamento "esterno" del sistema da sviluppare, senza dover specificare come tale comportamento viene realizzato. Forniscono una descrizione dei modi in cui il sistema potrà essere utilizzato. Essi sono basati sulla metafora del dialogo tra utente (attore) e sistema.
L'attore rappresenta qualsiasi elemento esterno al sistema (persona o altro sistema) che usa il sistema per fare qualcosa. Nel nostro caso, l'attore è l'utente ed il sistema è Arduino.

Un caso d'uso è una collezione di scenari, di successo o di fallimento del conseguimento dell'obiettivo, correlati tra loro che descrivono come un attore usa un sistema per raggiungere un obiettivo. Di seguito è riportato il caso d'uso, con i relativi scenari, riferito al sistema realizzato:

♢ ***Caso d'uso: UC1*** - Controllo tramite *"pulsante"* del sistema.

♢ ***Attore:*** Utente, sensori.

♢ ***Input:*** Pressione del *"pulsante"* posto sul dispositivo.

♢ ***Precondizioni:*** Il dispositivo è dotato di un led che permette di segnalare se il sistema è o meno in funzione. Inoltre esso deve essere disposto in modo che il modulo di riconoscimento di colori sia sufficientemente vicino all'oggetto del quale si vuole riconoscere il colore.

♢ ***Output:***

1. (Scenario 1) Il dispositivo riconosce il colore, (1) l'altoparlante enuncia il colore rilevato.
2. (Scenario 2) Il dispositivo non riconosce il colore, (2) *"Colore non rilevato."*

♢ ***Postcondizioni:*** Il *"pulsante"* è rilasciato ed il led è acceso.

♢ ***Descrizione scenario:***

1. **Scenario 1**
   (a) L'utente avvicina il sensore di riconoscimento di colori all'oggetto.
   (b) L'utente preme il *"pulsante"* presente sul dispositivo.
   (c) Il sensore di riconoscimento di colori acquisisce e riconosce il colore dell'oggetto.
   (d) L'altoparlante emette il file audio corrispondente al colore rilevato (1) ed il led si spegne.
2. **Scenario 2**
   (a) L'utente avvicina il sensore di riconoscimento di colori all'oggetto.
   (b) L'utente preme il *"pulsante"* presente sul dispositivo.
   (c) Il sensore di riconoscimento di colori acquisisce e non riconosce il colore dell'oggetto.
   (d) L'altoparlante emette il messaggio (2).

3.2 Descrizione hardware del sistema

Il sistema è stato realizzato sfruttando una scheda elettronica *Arduino* munita di un modulo *TCS230* per il riconoscimento di colori e di un modulo *WTV020SD-16P* per realizzare la funzione vocale. L'idea è quella di utilizzare Arduino per far funzionare il modulo di riconoscimento di colori che grazie alla matrice (8x8) di fotodiodi di cui è predisposto, è in grado di riconoscere il colore dell'oggetto posto in prossimità del sensore. La scheda è inoltre collegata al modulo WTV020SD-16P in grado di leggere file audio da una microSD e trasmetterli con un semplice altoparlante, in modo tale da associare al colore acquisito dal modulo TCS230 la registrazione vocale di qualcuno che pronuncia il colore. Come microSD è stata utilizzata una Transcendt da 1GB e al fine di garantirne la leggibilità dei file audio, questi ultimi sono stati salvati in formato .ad4. In Figura 3.1 è riportato lo schematico del dispositivo, mentre in Tabella 3.1 i collegamenti.

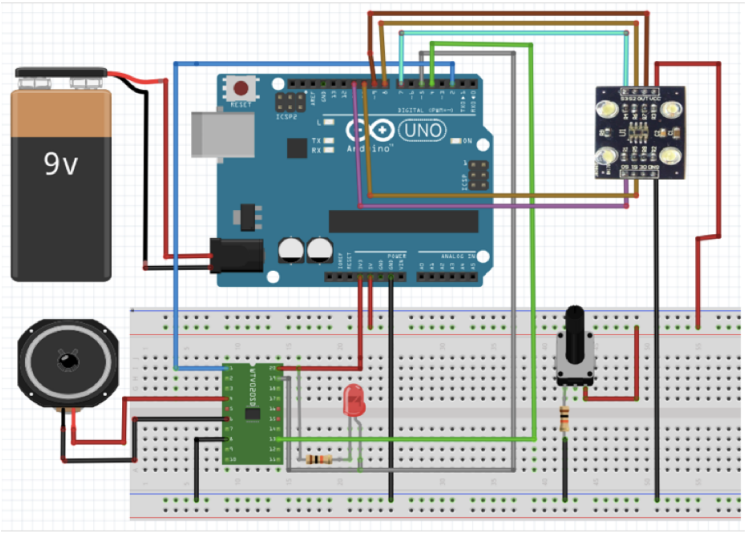

Figura 3.1: Schematico del dispositivo.

Pin Arduino digitali	*Modulo TCS230*	*Modulo WTV020sd-16p*	*Button*
2		Reset	
3		Clock	
4		Data	
5		Busy	
6			Button
7	S3		
8	Out		
9	S2		
10	S1		
11	S0		

Tabella 3.1: Tabella dei collegamenti.

3.2.1 Modulo TCS230

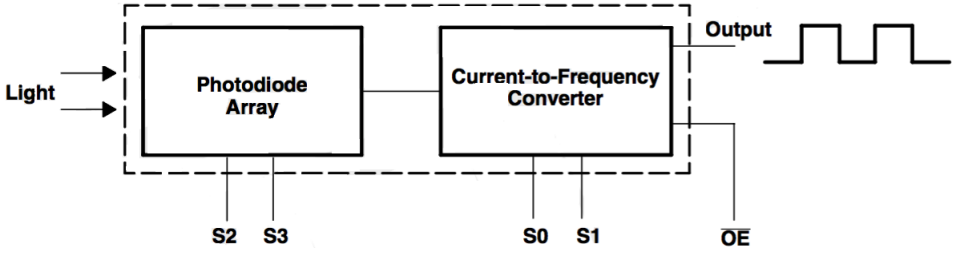

Figura 3.2: Block Diagram modulo TCS230.

Il modulo *TCS230* è un convertitore colore luce – frequenza. Quest'ultimo è composto da 16 fotodiodi con filtri rossi, 16 fotodiodi con filtri blu, 16 fotodiodi con filtri verdi e 16 non filtrati. Tutti i fotodiodi dello stesso colore sono posti in parallelo. Quando un oggetto è posto frontalmente al modulo, ad una distanza non superiore a 10 mm, questo viene illuminato dai 4 led bianchi e la luce riflessa andrà a colpire i 64 fotodiodi, ottenendo in uscita un'onda quadra (Duty cycle 50%) con una frequenza direttamente proporzionale all'intensità della luce riflessa. La frequenza a fondoscala in uscita, può essere regolata su uno dei tre valori preimpostati disponibili tramite due piedini dell'ingresso di controllo (S0 e S1) come riportato in Tabella 3.2

. Gli ingressi digitali e l'uscita digitale permettono di interfacciarlo direttamente ad Arduino o ad altri circuiti logici. I piedini S2 e S3 sono usati per selezionare quale gruppo di fotodiodi (rosso, verde, blu, chiaro) sono attivi.

S0	S1	OUTPUT FREQUENCY SCALING (f_o)
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	PHOTODIODE TYPE
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Tabella 3.2: Tabella delle opzioni.

3.2.2 Modulo WTV020SD-16p

Il modulo *WTV020SD-16p* ci permette di riprodurre dei brani salvati su una micro-SD, e grazie all'interfaccia con Arduino, è possibile scegliere quale brano eseguire ed in quale momento. Con questo modulo possiamo lavorare in due modi:

• autonomamente (con l'ausilio di un piccolo circuito);

• mediante Arduino (con l'ausilio di un'opportuna libreria).

Può essere collegato direttamente ad un piccolo altoparlante o ad un amplificatore. Tale dispositivo non legge direttamente file .mp3 ma file .ad4, quindi prima di caricarli sulla microSD è necessario convertirli con un tool opportuno. La scelta della scheda microSD è molto critica, infatti accetta schede con capacità massima di 2 GB e non tutte funzionano. Lo schema circuitale è riportato in Figura 3.3.

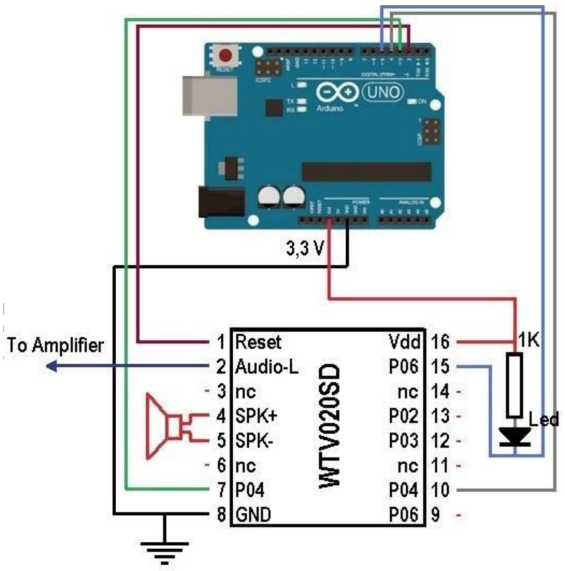

Figura 3.3: Schema circuitale modulo WTV020SD-16p.

3.3 Flow Chart del codice Arduino "Color_Sensor.ino"

Nel seguente paragrafo viene effettuata una descrizione di quello che è lo sviluppo del software del dispositivo. Come ambiente di sviluppo è stato utilizzato un IDE dedicato di cui Arduino è fornito e che permette di utilizzare come linguaggio di programmazione C/C++. Si è dunque realizzato uno Sketch rinominato *Color_Sensor* in cui sono state implementate le funzioni necessarie al dispositivo per effettuare le operazioni richieste. In particolare lo sketch prevede una prima fase di *define*, seguita da una fase di *setup* in cui, i pin utilizzati, sono stati impostati come input/output ed inizializzata la seriale per la comunicazione con Arduino. Quindi, è stata realizzata una funzione *readColor()*, la cui descrizione viene riportata in seguito, inserita nel metodo *loop()* e richiamata solo in seguito alla commutazione del pin *button*. In Figura 3.4 è riportato il flow chart relativo al codice Arduino:

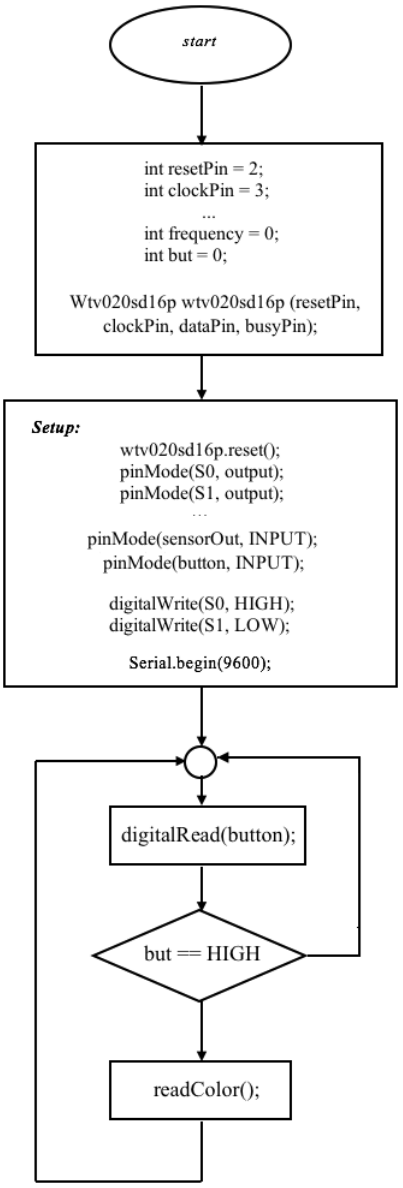

Figura 3.4: Flow chart "Color_Sensor.ino".

Funzione *readColor()* - tale funzione possiamo suddividerla essenzialmente in due parti:

• una prima parte in cui viene effettuato un setting dei fotodiodi, utilizzando i pin

corrispondenti e prelevata per ciascun insieme di fotodiodi, tramite la funzione *pulseIn(X,Y)*, le rispettive uscite;

• nella seconda parte vengono settati gli intervalli di valori di rosso, blu e verde al fine di identificare il colore dell'oggetto in questione.

Per la comunicazione con il modulo WTV020SD-16P è stata utilizzata un'apposita libreria reperibile al link http://www.adrirobot.it/WTV020-SD/programmi/Wtv020sd16p.zip.

3.4 Sequence Diagram

Il *Sequence Diagram* è un diagramma di interazione: evidenzia come una funzionalità è realizzata tramite la collaborazione di un insieme di oggetti. E' particolarmente adatto per la modellazione di sistemi real-time e per la descrizione del flusso di eventi degli use case. Enfatizza la sequenza temporale di scambio di messaggi tra oggetti, utilizzata per raggiungere un obiettivo. L'idea chiave è che le interazioni tra gli oggetti avvengano seguendo un ordine ben preciso e tale sequenza avvenga, nel tempo, dall'inizio alla fine. E' costituito da attori, da rettangoli aventi un nome, da messaggi rappresentati da linee continue recanti una freccia alla loro fine e dal tempo rappresentato come progressione verticale. Quando un oggetto invia un messaggio, quest'ultimo parte dalla *lifeline* (linea della vita) dello stesso ed arriva nella *lifeline* dell'oggetto ricevente. Tale linea della vita identifica una sorta di "periodo temporale di attivazione dell'oggetto" e può essere continua (quando ci si trova davanti alla rappresentazione di iterazioni da parte di attori) o spezzata (quando un oggetto non è sempre attivo, ma viene attivato o si attiva solo in base a determinate chiamate). Di seguito si riporta il Sequence Diagram relativo al caso d'uso sopra descritto.

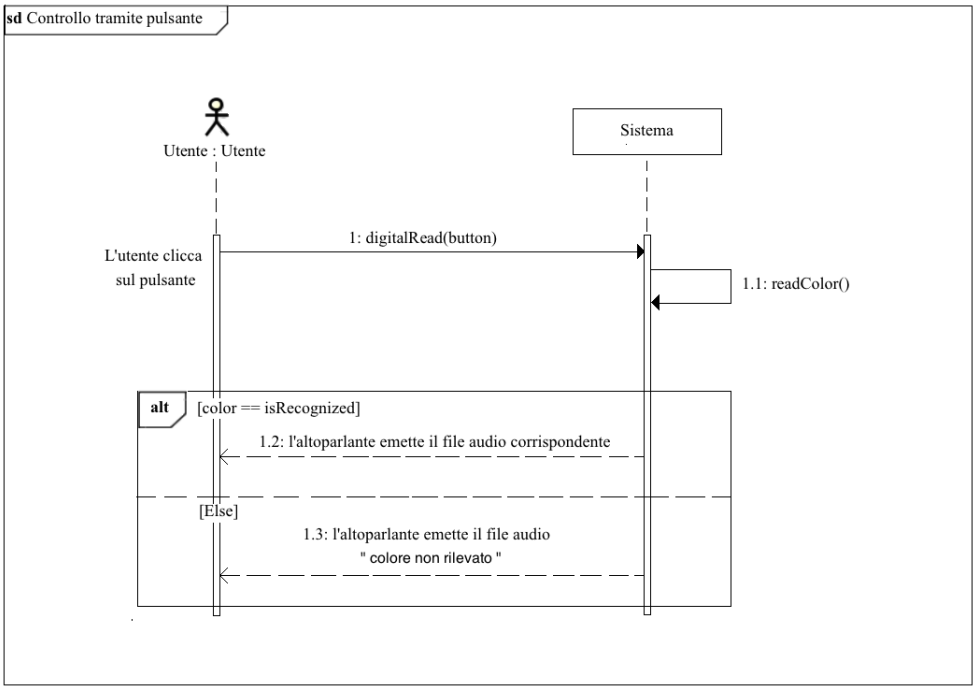

Figura 3.5: Sequence Diagram.

3.5 Deployment Diagram

Un *"Deployment Diagram"* viene sfruttato per modellare l'hardware utilizzato per l'implementazione del sistema e i collegamenti tra i diversi pezzi hardware. L'UML fornisce dei simboli che hanno il compito di favorire la creazione di un quadro chiaro di come il settaggio e la composizione finale dell'hardware dovrà essere. Un cubo rappresenta un nodo che corrisponde ad un elemento fisico del sistema. Un nodo ha un suo nome ed è possibile anche usare uno stereotipo per indicare il tipo di risorsa che esso rappresenta. Se un nodo fa parte di un package, allora il nome del package deve precedere il nome del nodo. Una linea che unisce due cubi rappresenta una connessione tra i due nodi. E' possibile usare uno stereotipo anche per fornire informazioni sulla connessione. Essi possono anche comprendere ulteriori componenti del

sistema al proprio interno, quali ad esempio le componenti software utilizzate. In Figura 3.6 viene riportato il Deployment Diagram relativo all'architettura hardware del progetto:

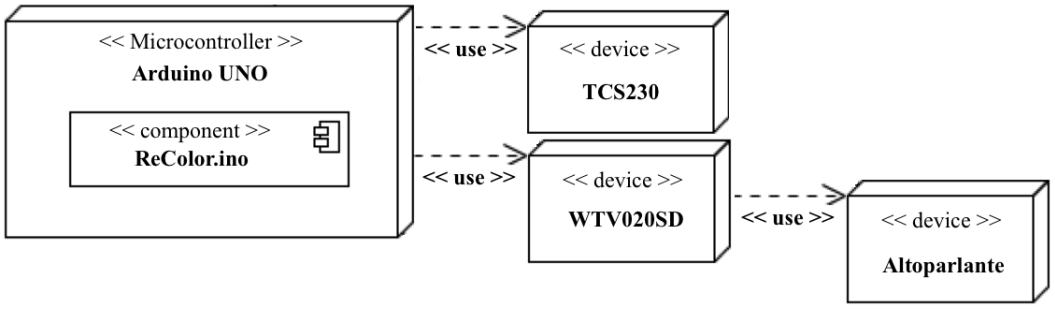

Figura 3.6: Deployment Diagram.

Per lavorare correttamente Arduino, stereotipato come "Microcontrollore", ha bisogno di usufruire di diversi device esterni, ovvero il trasduttore TCS230 ed il modulo ausiliario WTV020SD. Quest'ultimo è interconnesso all'altoparlante.

Capitolo 4

Testing

4.1 Black-box Testing (Testing funzionale)

Il testing effettuato sul dispositivo è un testing di tipo funzionale, cioè basato sulla definizione delle specifiche dello stesso. Sulla base dei requisiti del sistema, descritti in precedenza, sono stati considerati specifici input per poi analizzare la risposta del sistema. La strategia che è stata utilizzata per la definizione dei casi di test è quella del *Category-Partition*, consentendo di identificare relazioni esistenti tra le classi di equivalenza e dunque ridurre i costi in termini di numero di casi di test.

Nello specifico identifichiamo come input del dispositivo:

• L'interruttore che permette l'esecuzione delle operazioni volte al corretto funzionamento del dispositivo (***Interruttore BT***).

• I diversi colori per cui il dispositivo è abilitato a funzionare, in particolare la lista di colori è la seguente: rosso, blu, bianco, nero, giallo, verde, grigio, marrone. Tale lista risulta poi rappresentare il range di valori di input ammissibili (***Colore CL***).

Nello stesso listato sono stati inseriti tra parentesi i parametri del Category-Partition utili per presentare i test frame. Per ogni parametro sono stati dunque definiti i possibili valori:

• Interruttore **BT**:

1. *ON*;
2. *OFF*.

• Colore **CL**:

1. *Rosa* [***error***];
2. *Rosso* [***if interruttore is ON***];
3. *Giallo* [***if interruttore is ON***];
4. *Verde* [***if interruttore is ON***];

Definite quindi le specifiche di test per i parametri considerati, si riportano i *Test frame* considerati:

BT1 - CL1 BT = ON, CL = *'Rosa'*;

BT1 - CL2 BT = ON, CL = *'Rosso'*;

BT1 - CL3 BT = ON, CL = *'Giallo'*;

BT1 - CL4 BT = ON, CL = *'Verde'*;

BT2 BT = OFF.

Si riportano in forma tabellare i casi di test:

TC1 – Test frame BT1 –CL 1	
Input	*Pressione interruttore di esecuzione del dispositivo*
Output	*Messaggio di segnalazione audio: "colore non rilevato"*
Esito	***Positivo***

TC2 – Test frame BT1 –CL 2	
Input	*Pressione interruttore di esecuzione del dispositivo*
Output	*Messaggio di segnalazione audio: "Rosso"*
Esito	***Positivo***

TC3 – Test frame BT1 –CL 3	
Input	*Pressione interruttore di esecuzione del dispositivo*
Output	*Messaggio di segnalazione audio: "Giallo"*
Esito	***Positivo***

TC4 – Test frame BT1 –CL 2	
Input	*Pressione interruttore di esecuzione del dispositivo*
Output	*Messaggio di segnalazione audio: "Verde"*
Esito	***Positivo***

Tabella 4.1: Casi di test.