Integrazione di Sistemi Embedded
Progetto Oscilloscopio

Antonio Tudisco 285433 Lorenzo Lagostina 288019

23 giugno 2022

Indice

1 STM32 - Acquisizione campioni

1.1 Spiegazione generale

Il campionamento dei dati da parte dell'ADC del uC viene avviata tramite un evento generato dal TIM2. I campioni vengono salvati in apposite variabili di appoggio tramite il DMA. La gestione del buffer avviene nella ISR del TIM2. Per poter raccogliere anche i dati precedenti al trigger, utilizziamo la prima metà del buffer come una FIFO, dove, finché non verrà individuato il trigger, verranno salvati i dati partendo dalla posizione 255 del buffer, traslando ogni volta i dati verso la posizione zero.

All'interno dell'ISR del ricevitore della USART vengono analizzati i byte in ingresso, verificando che rispettino il formato <*><COMANDO><#>. Il comando viene successivamente analizzato per ottenere il tipo di funzione da chiamare ed il valore dell'argomento. Una volta che sono stati raccolti 253 campioni viene avviata la trasmissione di questi in formato esadecimale al PC.

1.2 Struttura dati

La struttura dati è composta da:

- Un buffer di 513X2 interi su 8 bit, la dimensione di 513 campioni è stata scelta per poter posizionare sul grafico il trigger sia in posizione +128 che -128, partendo dal presupposto che il trigger è presente nella posizione 255, ossia il 256-esimo campione.
- Un vettore di interi su 32 bit di appoggio dove salvare gli ultimi campioni ottenuti. Il parallelismo a 32 bit è necessario per poter interagire con la DMA.
- Un intero su 16 bit che serve da indice per puntare alla prossima locazione del buffer in cui salvare i campio, una volta che il trigger è stato individuato, nella modalità AUTO questo indice viene riutilizzato prima dell'individuazione del trigger per contare i valori acquisiti, in modo da poter avviare la seconda parte del campionamento una volta raccolti 255 campioni.
- Tre variabili intere su 8 bit per salvare il livello di trigger, la modalità del trigger, e per salvare l'avvenuta ricezione del trigger.
- Un carattere per salvare i byte in ricezione dal PC
- Un array di 12 caratteri per salvare i comandi ricevuti ed un intero su 8 bit per contarne la lunghezza.

1.3 Flowchart e funzioni

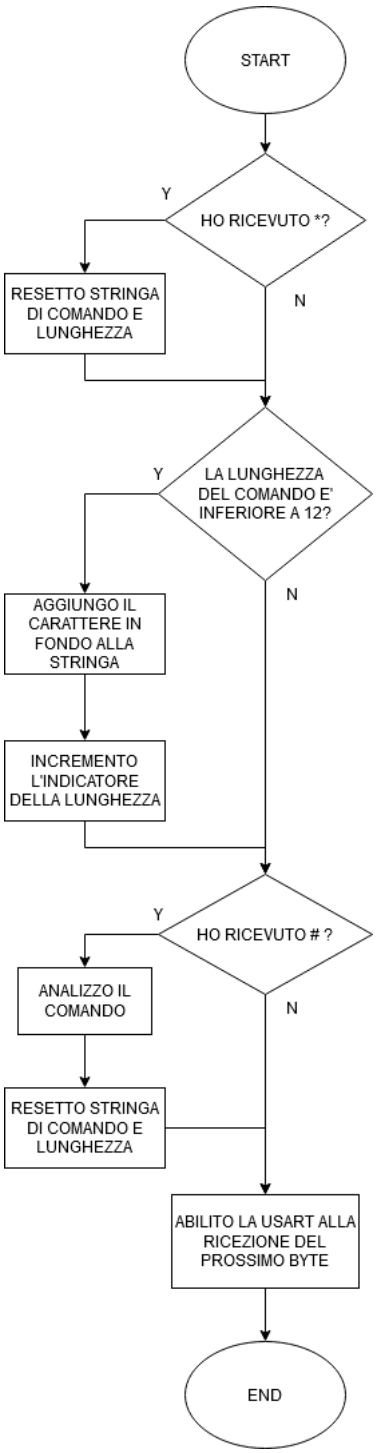

Figura 1: *HAL_UART_RxCpltCallback*

Nella ricezione viene prestata attenzione all'impedire che si possano inviare comandi più lunghi del previsto.

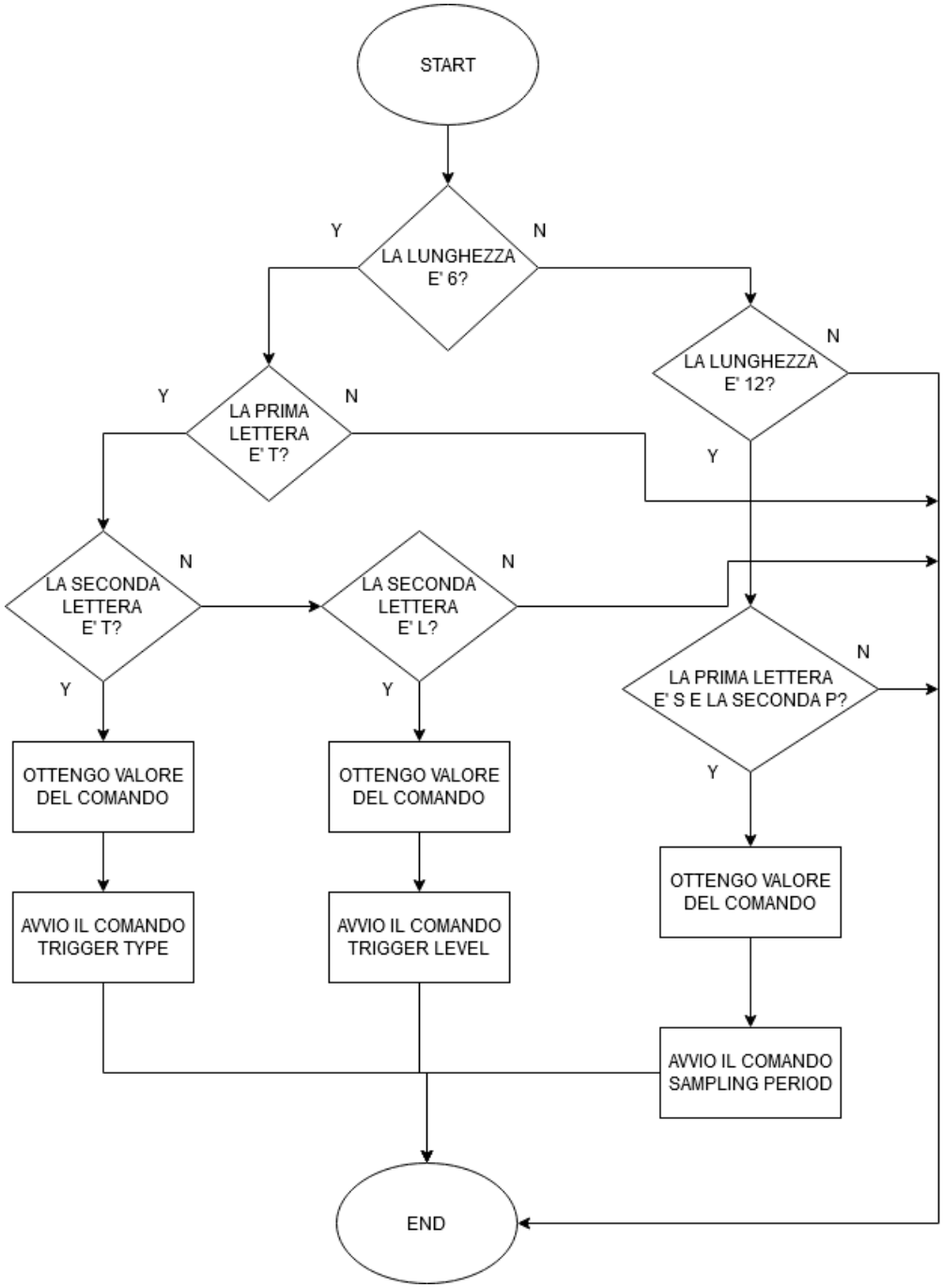

Figura 2: *analyseCommand*

Distinguo i diversi comandi, ricavo i valori associati ed avvio le funzioni relative.

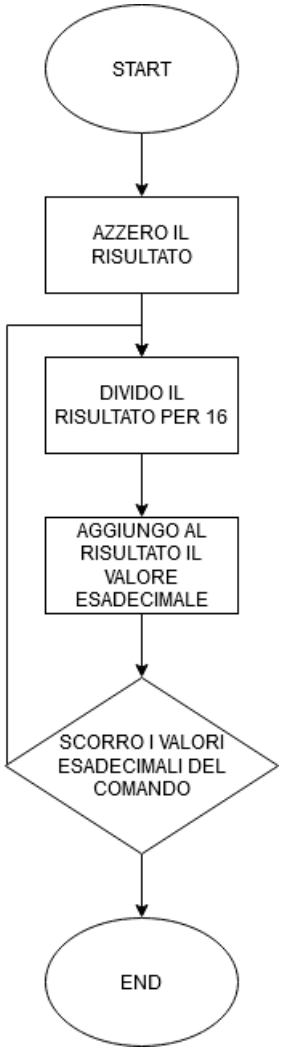

Figura 3: *obtainCommandValue*

Scorro i valori esadecimali della stringa di comando e li converto in decimale, per fare ciò alla funzione vengono passati gli indici estremali della parte di caratteri esadecimali all'interno della stringa di comando.

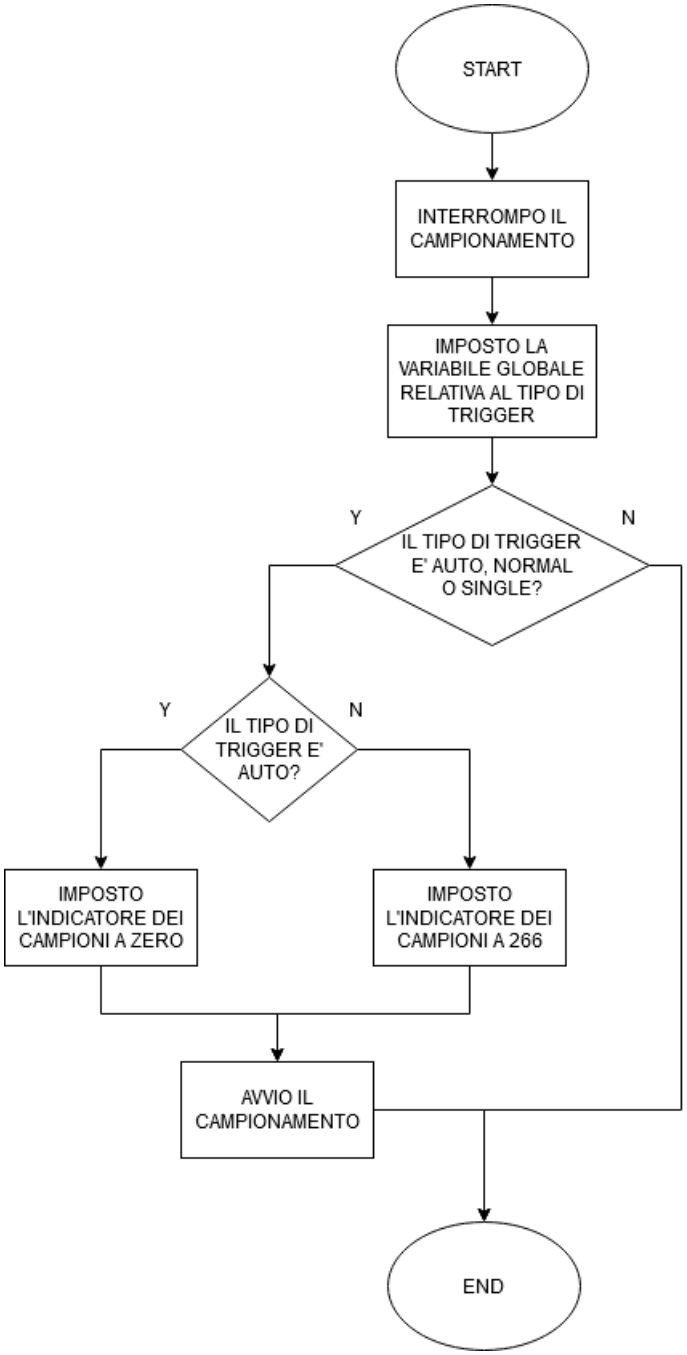

Figura 4: *triggerType*

Interrompo l'esecuzione del campionamento ed imposto il tipo di trigger, se la modalità è AUTO o NORMAL alla fine della funzione riprendo il campionamento.

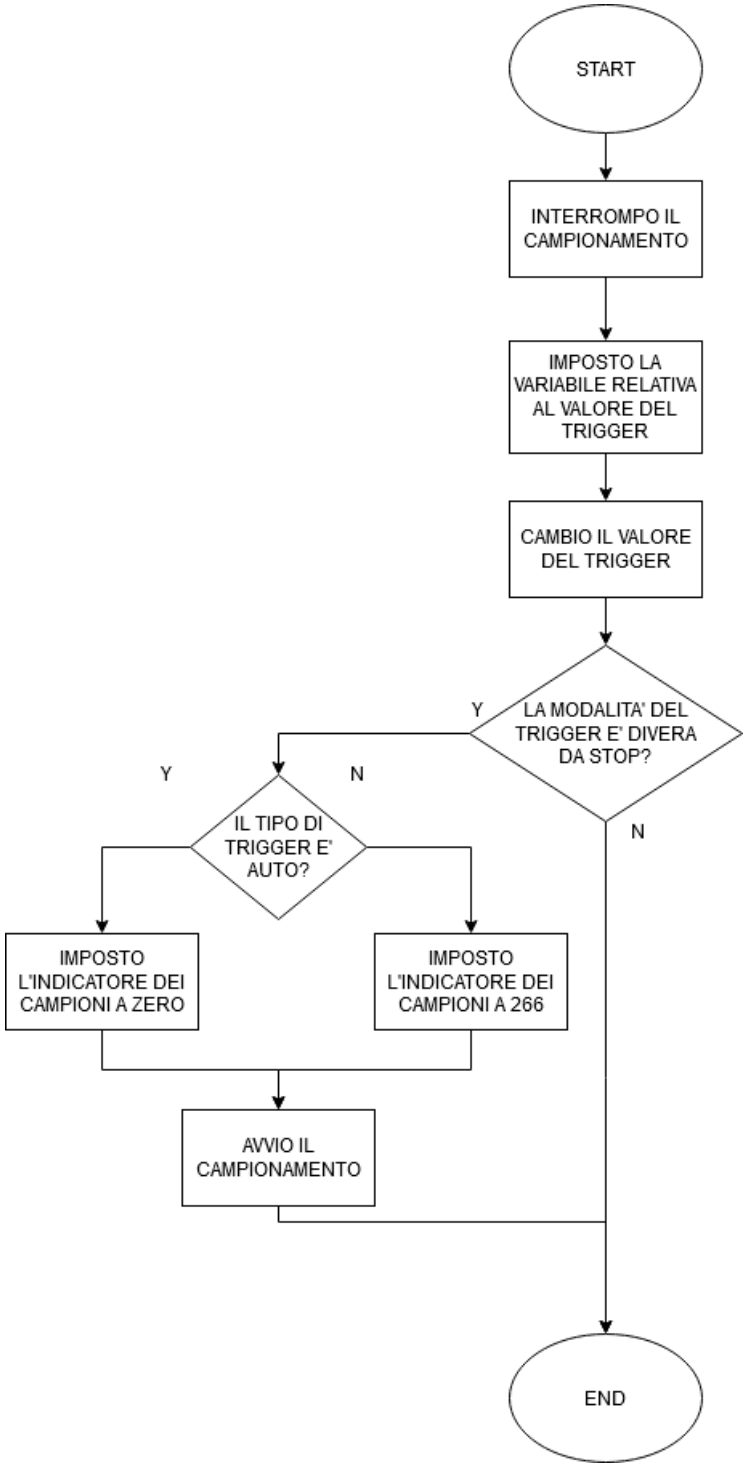

Figura 5: *triggerLevel*

Stesso comportamento della funzione precedente.

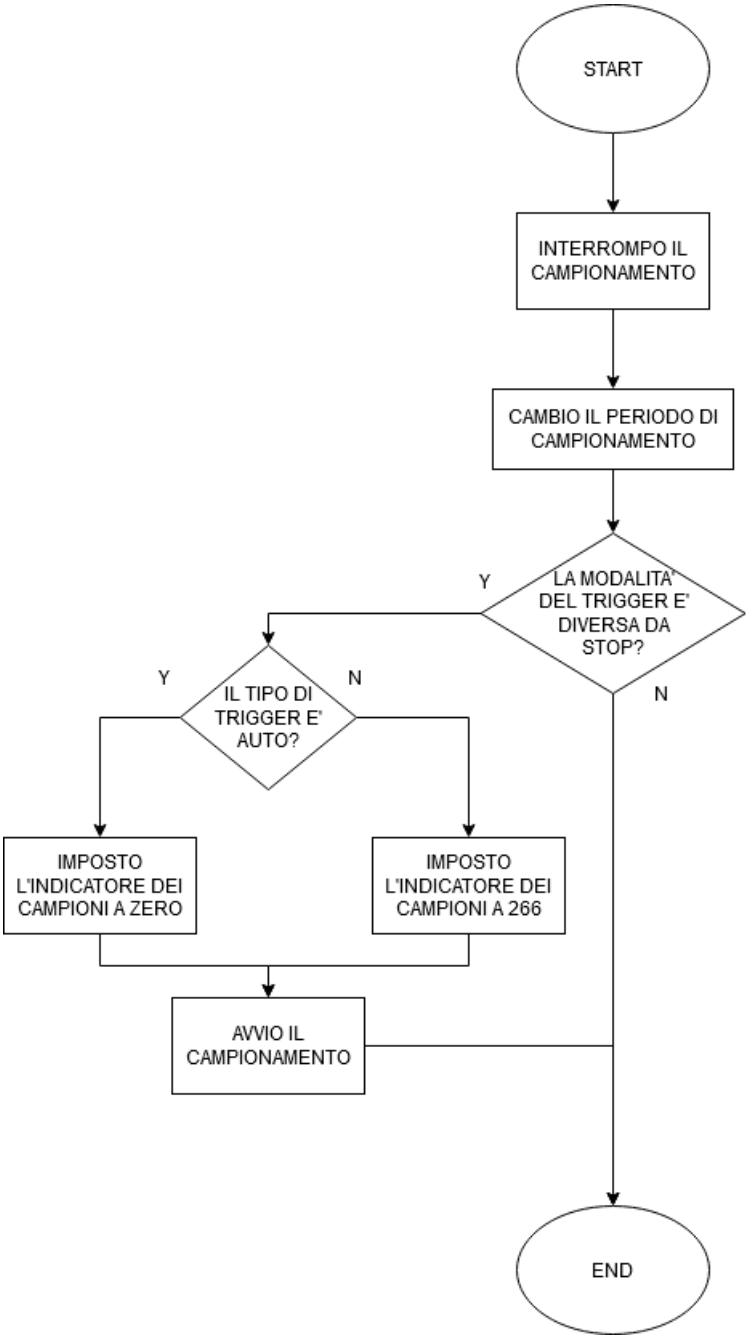

Figura 6: *samplingPeriod*

Stesso comportamento delle due funzioni precedenti. E' opportuno specificare che le queste tre funzioni non si occupano di cambiare le impostazioni del uC, bensì di chiamare le funzioni che lo fanno una volta aver creato le condizioni adeguate (i.e. campionamento interrotto)

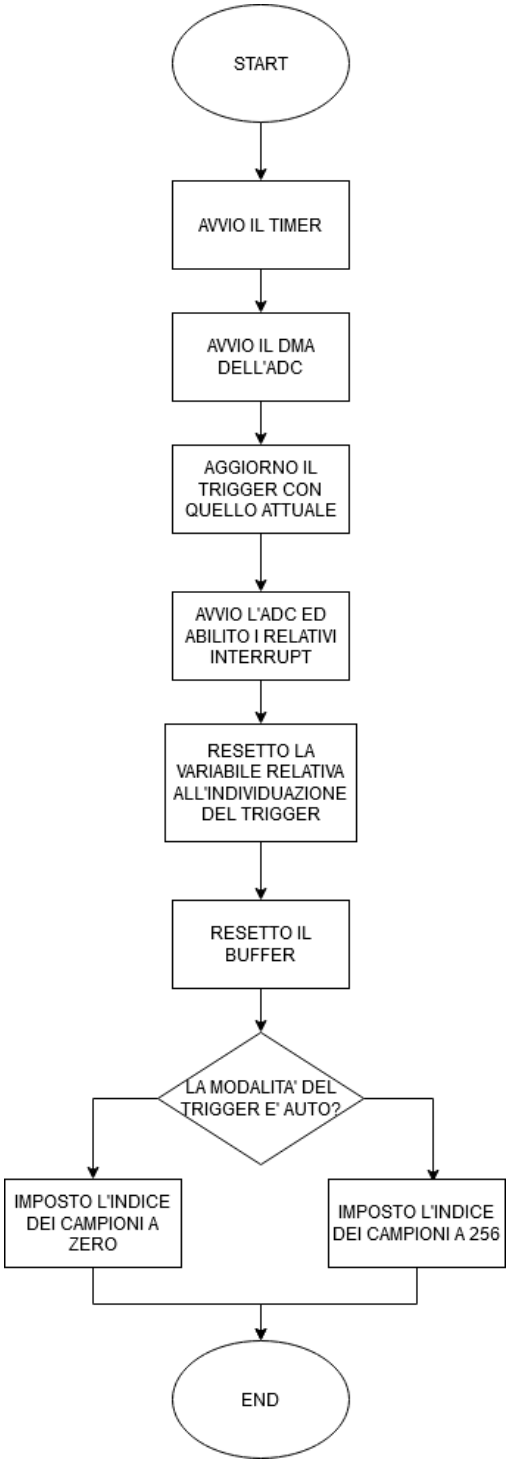

Figura 7: *startSampling*

Questa funzione racchiude tutti i metodi relativi alle impostazioni per l'avvio del campionamento.

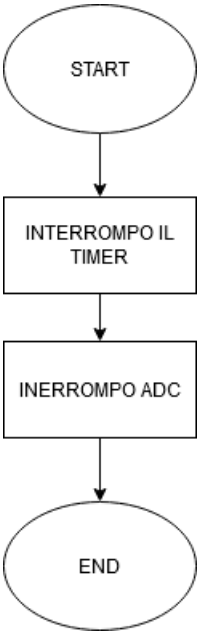

Figura 8: *stopSampling*

Similmente alla precedente per l'interruzione del campionamento.

Figura 9: *changeTrigger*

Abilito l'analog watchdog ed imposto il valore di threshold superiore associato.

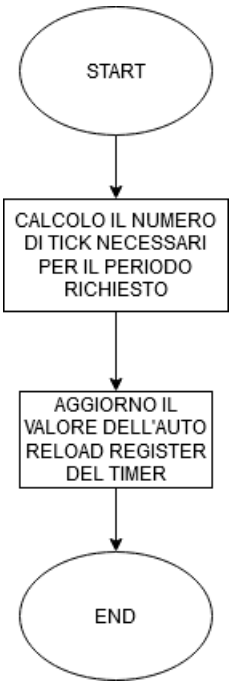

Figura 10: *changeSamplingPeriod*

Divido il periodo di campionamento richiesto per il valore in nanosecondi del periodo di clock, in questo modo ottengo il numero di tick da inserire nell'ARR.

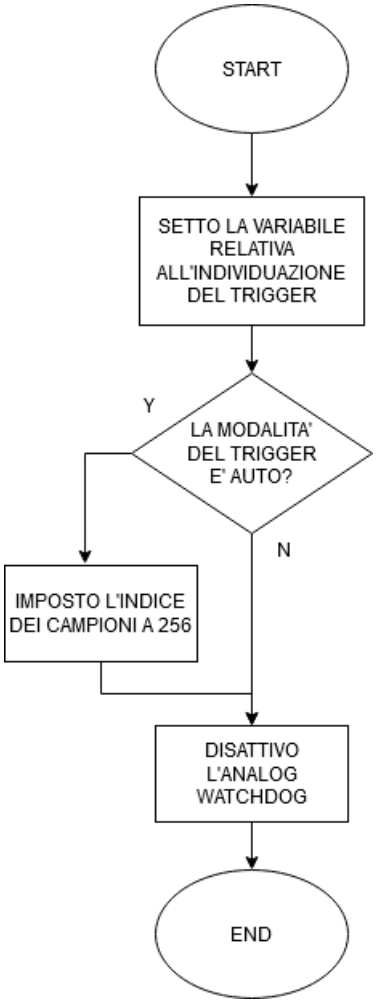

Figura 11: *HAL_ADC_LevelOutOfWindowCallback*

Segnalo l'avvenuta ricezione del trigger e nel caso della modalità AUTO imposto l'indice dei campioni a 256, in modo da continuare il campionamento a prescindere dal numero di valori acquisiti.

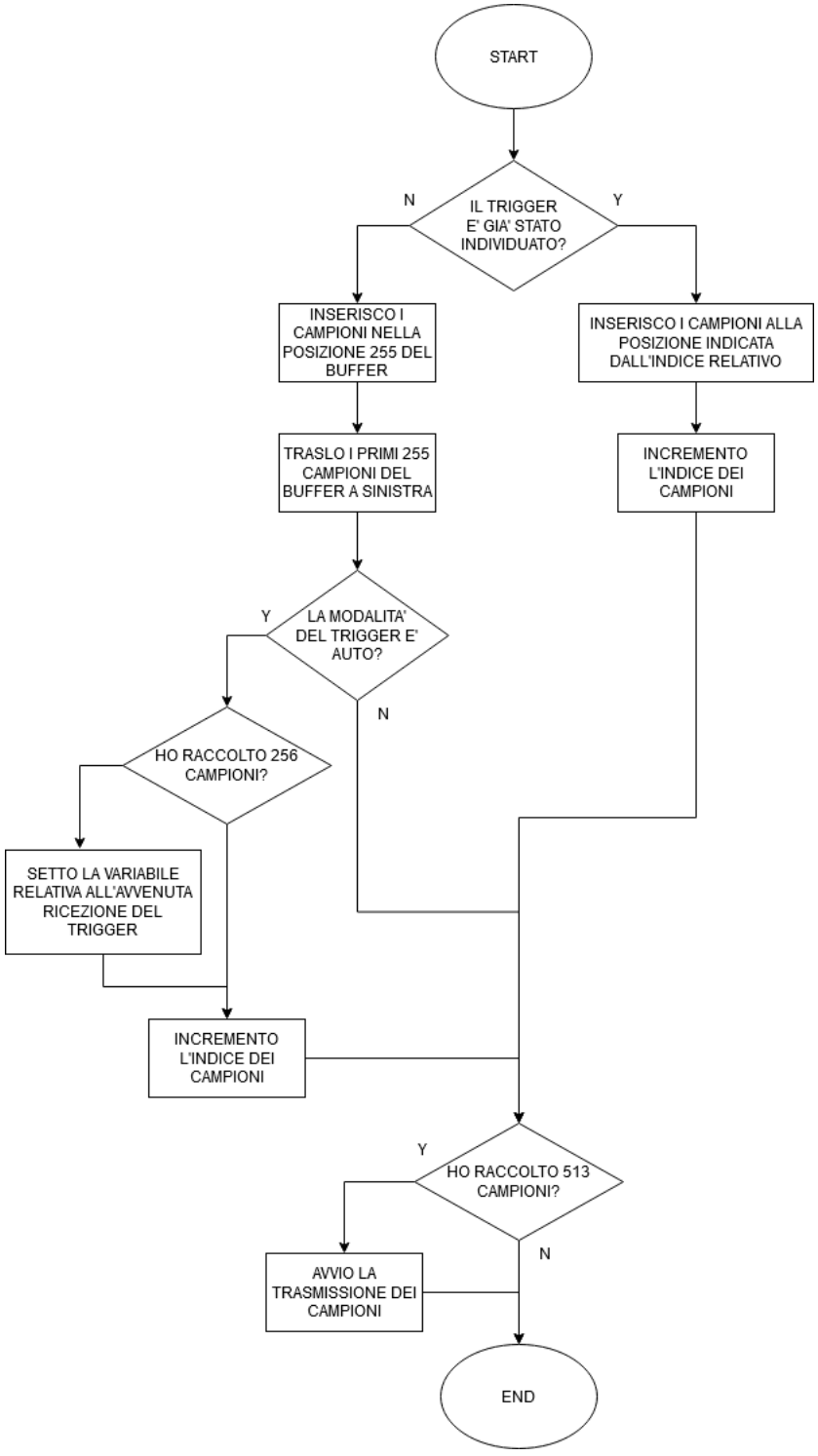

Figura 12: *HAL_TIM_OC_DelayElapsedCallback*

Se non vi è ancora stato il trigger, traslo la prima metà del buffer e salvo l'ultimo campione acquisito nella posizione del trigger (i.e. 256), nel caso la modalità corrente sia AUTO e siano stati acquisiti 256 campioni, verrà segnalata la presenza del trigger.

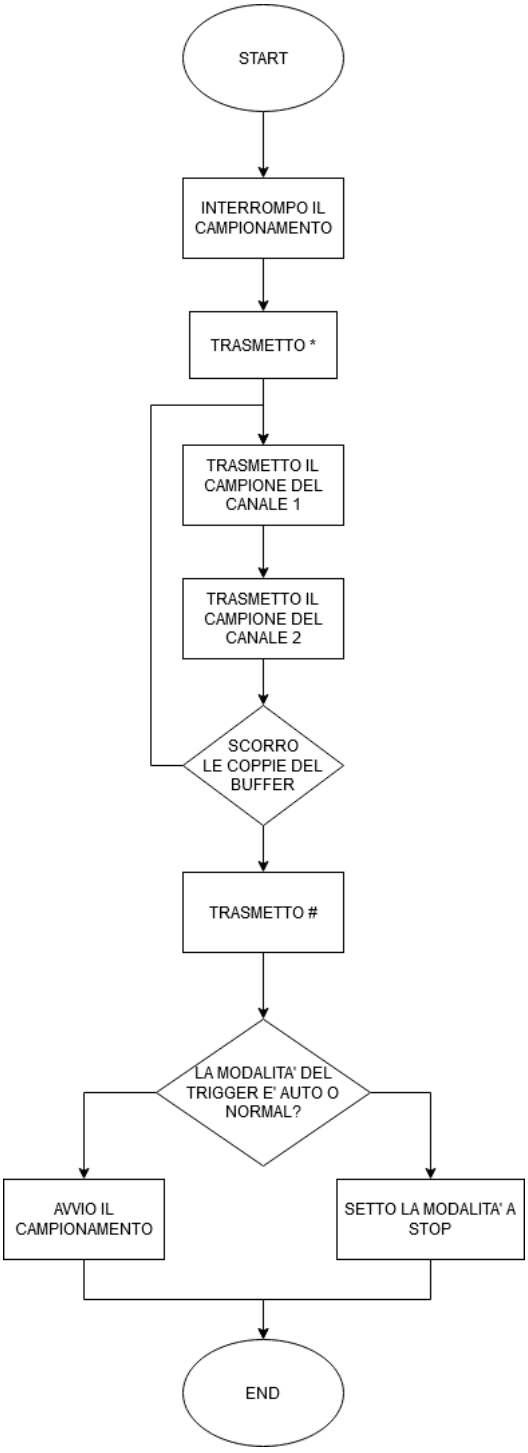

Figura 13: *samplesTransmission*

Trasmetto prima il <*>, poi le 513 coppie di campioni ed infine il <#>.

2 Python - Interfaccia Grafica

2.1 Spiegazione generale

L'obiettivo di questa parte è la realizzazione di un' interfaccia grafica che possa essere utile all'utente e che possa, previa selezione, inviare dei comandi al microcontrollore e sia anche in grado di ricevere dei dati e mostrare il risultato in modo grafico. Innanzitutto, era necessario suddividere il problema in sezioni più piccole, partendo dal distinguere i problemi relativi alla seriale dalla parte di interfaccia.

2.2 Seriale

La seriale deve fare 2 operazioni, leggere i dati inviati dal microcontrollore e deve anche poter inviare i comandi relativi alle impostazioni scelte dall'utente.

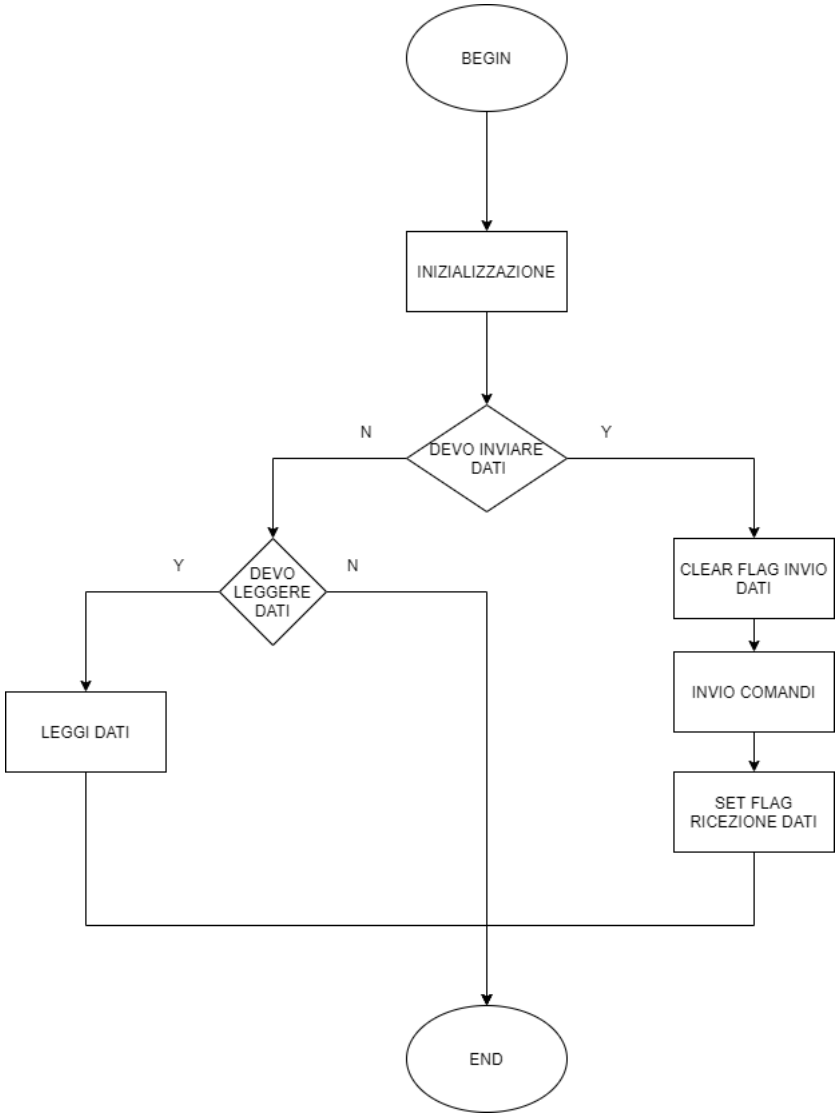

Figura 14: *Comportamento della seriale*

Costantemente deve essere controllato lo stato in cui deve lavorare la seriale, per capire se sia necessario fermarsi, leggere i dati o inviare i comandi. Ogni qual volta è necessario modificare le impostazioni, viene inviato un segnale di STOP al microcontrollore Per quanto concerne la lettura abbiamo pensato di leggere 1 byte per volta, e ogni qualvolta il messaggio venga ricevuto correttamente, devono essere memorizzati i valori e inviati alla parte relativa al grafico. Il problema principale si pone relativamente al salvataggio dei dati, per risolvere ciò abbiamo pensato di implementare questo algoritmo.

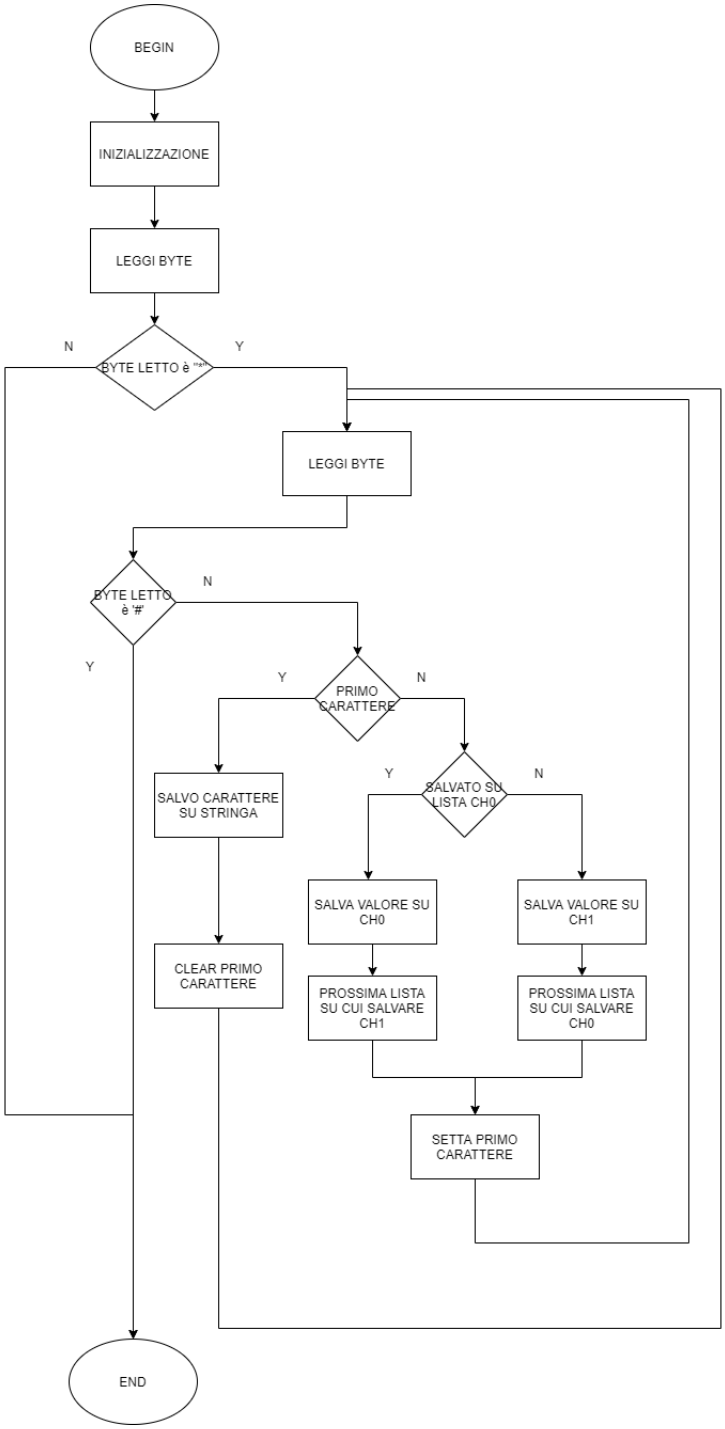

Figura 15: *Lettura dei byte ricevuti dal uC*

L'algoritmo è pensato in modo tale che i byte che arrivano via seriale contengono il valore di due caratteri, questi due caratteri sono 2 diverse cifre esadecimali, pertanto deve essere effettuata una trasformazione ogni volta da stringa a intero. Per quanto concerne la parte di invio comandi, questi sono inviati secondo le specifiche richieste dalla traccia.

Sempre relativamente alla parte seriale si è pensato di aggiungere la possibilità all'utente di scegliere la seriale che si ha intenzione di utilizzare, per fare ciò, si crea dapprima una finestra con all'interno una COMBOBOX, ovvero un menu a tendina in cui sono presenti tra le opzioni le varie seriali che si possono selezionare. Qualora la seriale presenti qualche errore di apertura questo viene segnalato mediante messaggio di errore. Le varie scelte di seriale sono mantenute in una lista. Una volta che la seriale scelta è aperta, la finestra viene automaticamente chiusa.

2.3 Grafico

Per quanto concerne il grafico, questo deve ricevere due liste, trasformarle in array selezionando l'opportuno intervallo scelto, e questi array devono successivamente essere mostrati graficamente. Per fare ciò abbiamo deciso di trattare questo come un oggetto in cui sono definite due funzioni, la funzione di inizializzazione(il costruttore) in cui vengono passati il frame su cui deve essere istanziato il grafico, e le due coordinate x e y per definire in che punto questo debba essere istanziato. L'altra funzione necessaria è quella relativa all'assegnare i valori dei due array per poterli rappresentare in via grafica. Per quanto concerne le strutture dati utilizzate sono degli array che mantengono i vari valori di y. Per cercare di rendere la parte relativa al grafico comprensibile abbiamo pensato di mettere trasformare la coordinata y in modo tale che i valori siano compresi nell'intervallo 0-3.3 V. I valori su x sono compresi nell'intervallo 0 e 255. Un ulteriore argomento passato è il trigger position, che serve per selezionare opportunamente quali range di valori prendere dal buffer ricevuto(da 128 – triggerPosition a 384 – triggerPosition).

2.4 Entry

Per quanto concerne l'interfaccia grafica, sono presenti delle entry, ovvero degli spazi in cui l'utente può inserire dei valori all'interno. Questi oggetti sono stati definiti comprensivi di entry, label (ovvero una sorta di leggenda, serve per far capire all'utente a cosa sia relativa l'entry) e una entry in modalità readonly, in cui viene segnalato l'eventuale errore. Questi oggetti sono comuni per la parte relativa al Trigger Level, la parte relativa al Trigger Position e anche per il Sampling Period. Oltre alla parte relativa al costruttore, sono presenti ulteriori metodi. Sono stati definiti dei metodi di enable e disable dell'entry, necessari al fine permettere o impedire all'utente di modificare i valori inseriti, un metodo che permette di disporre gli oggetti nel frame, una volta passate le coordinate, ed infine è presente un metodo chiamato getValue che ritorna il valore inserito. Come detto precedentemente questo oggetto è comune sia per il Sampling Period, sia per il Trigger Level che il Trigger Position, questi oggetti però appartengono a tipologie diverse. Per quanto concerne il Sampling Period, viene aggiunta a destra dell'entry un menu a tendina in cui viene data la possibilità di scegliere l'unità di misura, e qualora non venisse scelta, l'unità di misura di default è il nanosecondo. Per realizzare questa Combobox è stato necessario utilizzare una lista per fornire le varie scelte e un ulteriore dizionario contenente come chiave l'unità di misura e come valore il fattore moltiplicativo da utilizzare per avere il numero in nanosecondi.

Per esempio, il valore nel caso del millisecondo è 10^6.Anche in questo caso abbiamo utilizzato la funzione che permettesse di abilitare o disabilitare l'oggetto. Inoltre, per quanto concerne il valore di uscita, deve dapprima essere verificata la correttezza del valore (se sia un valore float valido e sia dentro il range di valori possibili) e poi può essere salvato il valore e anche utilizzato. A differenza del Sampling Period, gli oggetti relativi al Trigger Level e al Trigger Position, non presentano il menu a tendina, ma presentano una Label in cui è scritta l'unità di misura, ovvero "LSB". Anche in questo caso deve essere verificato che sia un numero valido (un intero compreso nel range scelto).

2.5 Trigger mode

Per quanto concerne questo oggetto, viene data la possibilità all'utente di scegliere la modalità desiderata mediante menu a tendina. Qualora non venisse selezionata alcuna scelta, l'impostazione di default sarà la modalità STOP. Anche in questo caso sono presenti una label e i metodi di enable e disable per permettere o impedire all'utente di modificare le impostazioni.

2.6 Bottone run

Per distinguere la modalità di RUN dalla modalità di STOP, abbiamo preferito usare un bottone che quando cliccato cambierà il tipo lo stato dell'interfaccia, permettendo o impedendo di modificare le impostazioni dell'oscilloscopio.

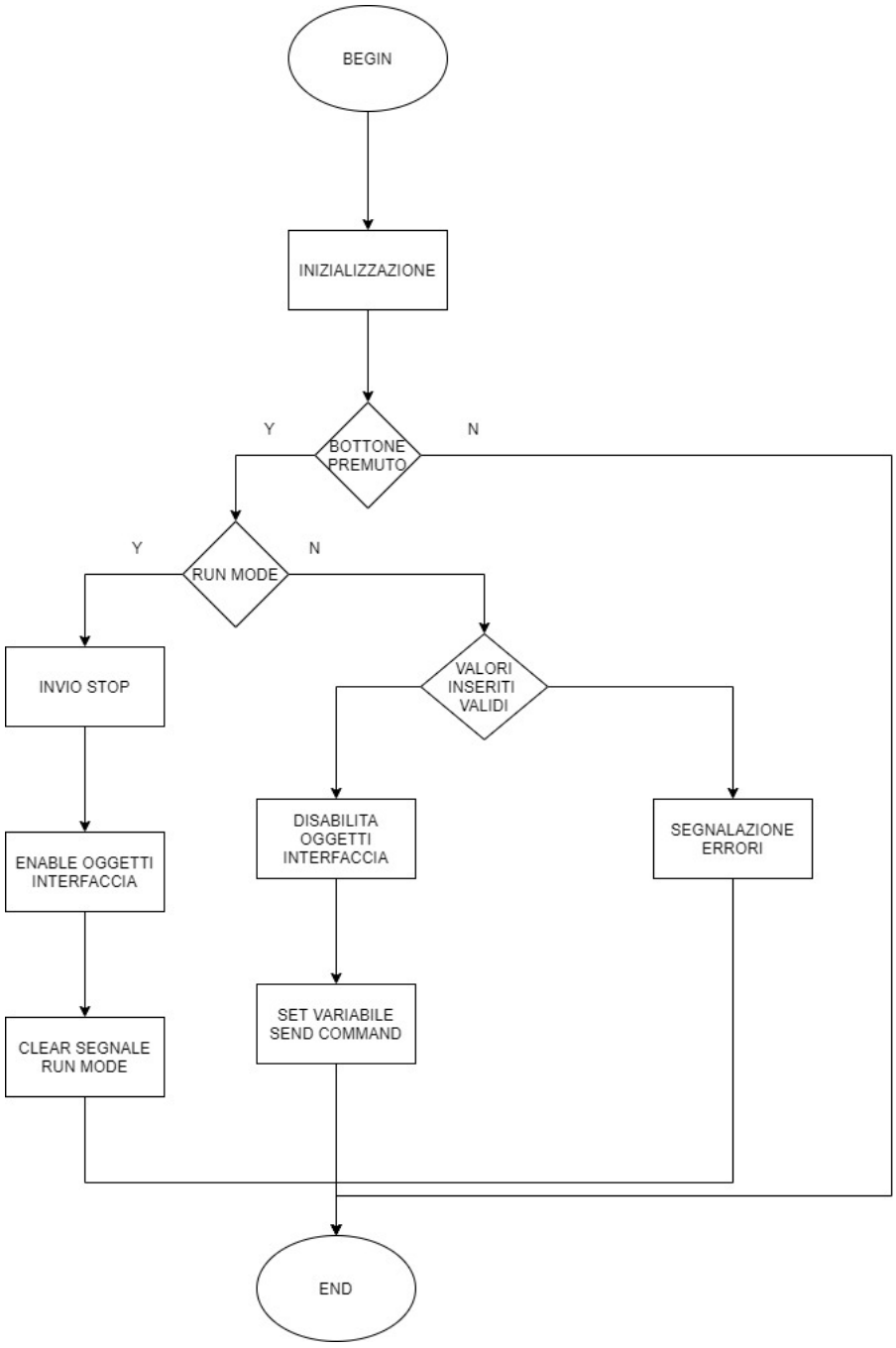

Figura 16: *Algoritmo relativo al bottone di run*

2.7 Disposizione

Una volta descritti i vari oggetti abbiamo deciso che questi dovessero essere posizionati nel seguente modo.

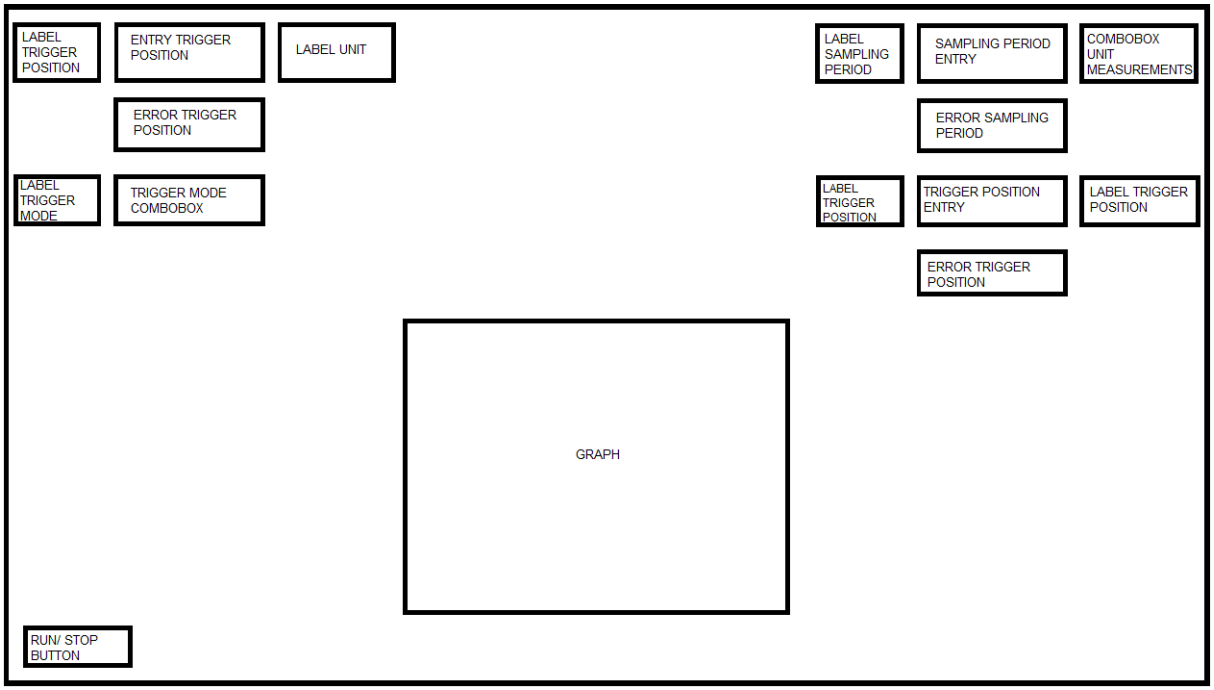

Figura 17: *Disposizione degli oggetti sull'interfaccia*

3 Risultati

3.1 Test

Il primo test è stato eseguito campionando due onde quadre a una frequenza di 80Hz e tensione e duty cycle differenti in modalità NORMAL; per poter catturare in una metà del buffer l'intero segnale è stato usato un periodo di campionamento di 48828ns, equivalente a 20.48kHz.

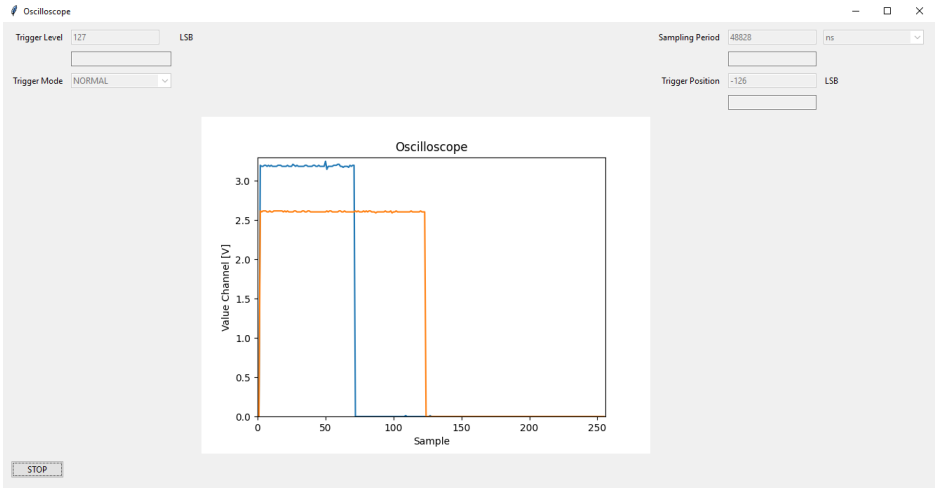

Figura 18: *Test su onde quadre con DC e tensioni diverse*

Per testare il comportamento dell'oscilloscopio alla minima frequenza di campionamento, è stata campionata, in modalità AUTO, la tensione in uscita da un potenziometro, la quale è stata fatta variare durante l'acquisizione dei dati tra 0V e 3.3V.

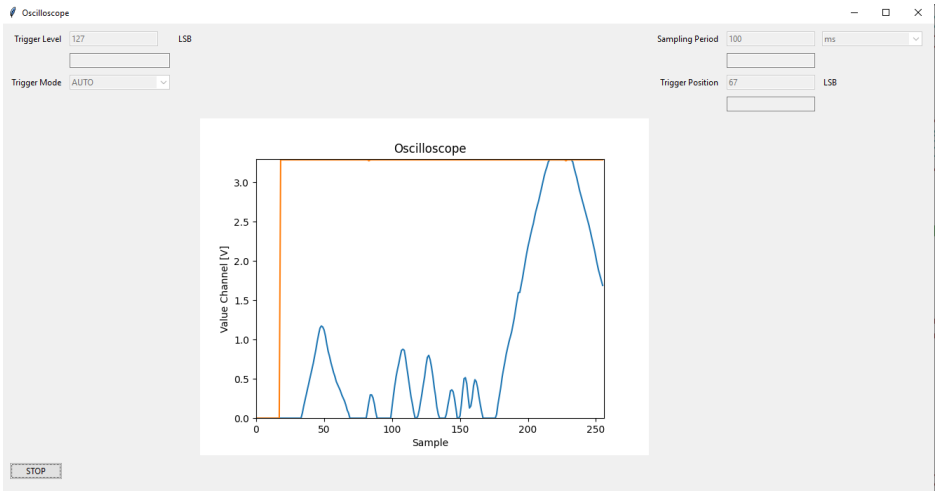

Figura 19: *Test su potenziometro azionato a mano*

Al fine di verificare il corretto funzionamento alla massima risoluzione possibile, è stato osservato il comportamento di un circuito di debouncing mentre viene rilasciato il pulsante. Per poter catturare questo segnale è stata usata la modalità SINGLE RUN.

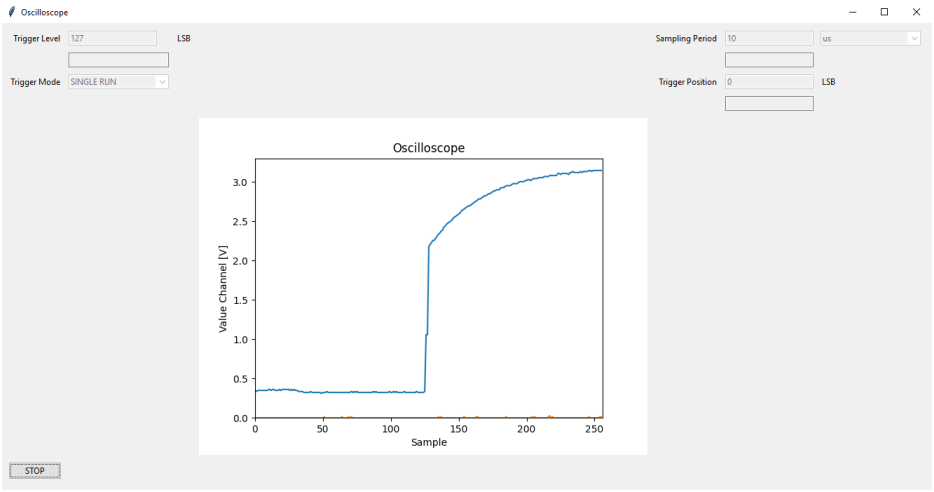

Figura 20: *Test durante il rilascio di un pulsante in un circuito di debouncing*

3.2 Spazio allocato nel uC

Memory Regions | Memory Details

Region	Start address	End address	Size	Free	Used	Usage (%)
RAM	0x20000000	0x20018000	96 KB	93,01 KB	2,99 KB	3,12%
FLASH	0x08000000	0x08080000	512 KB	488,46 KB	23,54 KB	4,60%

Figura 21: *Spazio allocato nella memoria del uC*