Laboratorio di Segnali e Sistemi
- Esercitazione -9 -

DFT con arduino

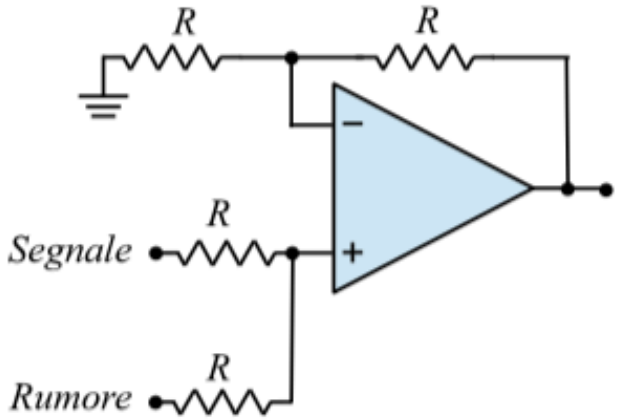
Claudio Luci

SAPIENZA
UNIVERSITÀ DI ROMA

last update : 070117

Scopo dell'esercitazione

Lo scopo di questa esercitazione è quello di effettuare con Arduino, con l'ausilio dell'applicazione Processing (vedi Appendice) un'analisi in frequenza di segnali, periodici o impulsivi, in assenza o in presenza di rumore.
Per i segnali periodici possiamo utilizzare il generatore di funzioni, per i segnali impulsivi utilizzeremo lo stesso Arduino. Potremo poi aggiungere del rumore (prodotto dal generatore gia' costruito) utilizzando un semplice sommatore non invertente, con amplificazione pari ad 1 (Fig 9.1).



Occorre un sommatore a tre ingressi per aggiungere un offset al rumore quando il segnale e' zero (studio dello spettro di frequenza del rumore).

- Analisi in frequenza: calcolo della trasformata di Fourier discreta
- Segnali periodici: sinusoidi a diverse frequenze, onda quadra
- Segnali impulsivi: "scarica" di un condensatore
- Inoltre vedremo come cambia lo spettro in frequenza con l'aggiunta del filtro passa-basso

Software

Per Arduino utilizzeremo lo sketch *adc_read_5*, scritto per lavorare in congiunzione con lo sketch di Processing*adc_arduino_5*.
L'interfaccia grafica dello sketch Processing ci consente di:

adc_read_5_2019, da scaricare dal mio sito

- Ordinare ad Arduino di avviare la presa dati. Il segnale verrà campionato 800 volte [1], ad una frequenza di ~10 kHz (~~ovvero la massima frequenza possibile con Arduino UNO~~) e i dati verranno inviati a Processing;

Arduino Due puo' arrivare a 200 kHz

- Calcolare la trasformata di Fourier discreta del campione (nonche' l'antitrasformata);
- Visualizzare il campione e il suo spettro in frequenza;
- Salvare su file le misure e i risultati del calcolo, per successive analisi off-line.

[1] E' il massimo numero di campioni possibile con la memoria disponibile sulla scheda Arduino UNO. La finestra temporale di campionamento e' quindi di circa ~~90 *ms*~~.

Nel nostro caso 80 ms

- abbiamo impostato analogRead() in modo da aver bisogno di circa 100 µs per essere completata; di conseguenza la massima frequenza di campionamento f_c è di circa 10 kHz
- Quindi la frequenza di Nyquist è di circa 5 kHz

N.B. In questa esercitazione dovrete usare processing (e arduino) come una black-box. Non c'e' nessun software da scrivere, come avete fatto nell'esercitazione precedente, ma ci sono tante misure da fare

Arduino e Processing

Per l'ultima esercitazione abbiamo bisogno di un software supplementare, l'applicazione Processing, che utilizzeremo congiuntamente ad Arduino.
Processing e' un'applicazione che consente di sviluppare contenuti interattivi. Eredita completamente la sintassi, i comandi e il paradigma di programmazione orientata agli oggetti dal linguaggio Java ma in piu' mette a disposizione numerose funzioni ad alto livello per gestire facilmente gli aspetti grafici e multimediali. Non e' difficile da usare per chi conosce gia' il linguaggio c.
E' distribuito sotto la licenza libera GNU General Public License, scaricabile gratuitamente dal sito ufficiale www.processing.org, e' supportato dai sistemi operativi Linux, Mac OS X e Microsoft Windows.
Processing puo' Interagire con le schede Arduino tramite USB, consentendcio quindi di analizzare sul PC i dati raccolti dal microcontrollore.

Processing fa (molto meglio) quello voi avete fatto "a mano" la volta scorsa nell'acquisire i vari punti di una data forma d'onda; in pratica e' un piccolo sistema di acquisizione dati completo (DAQ)

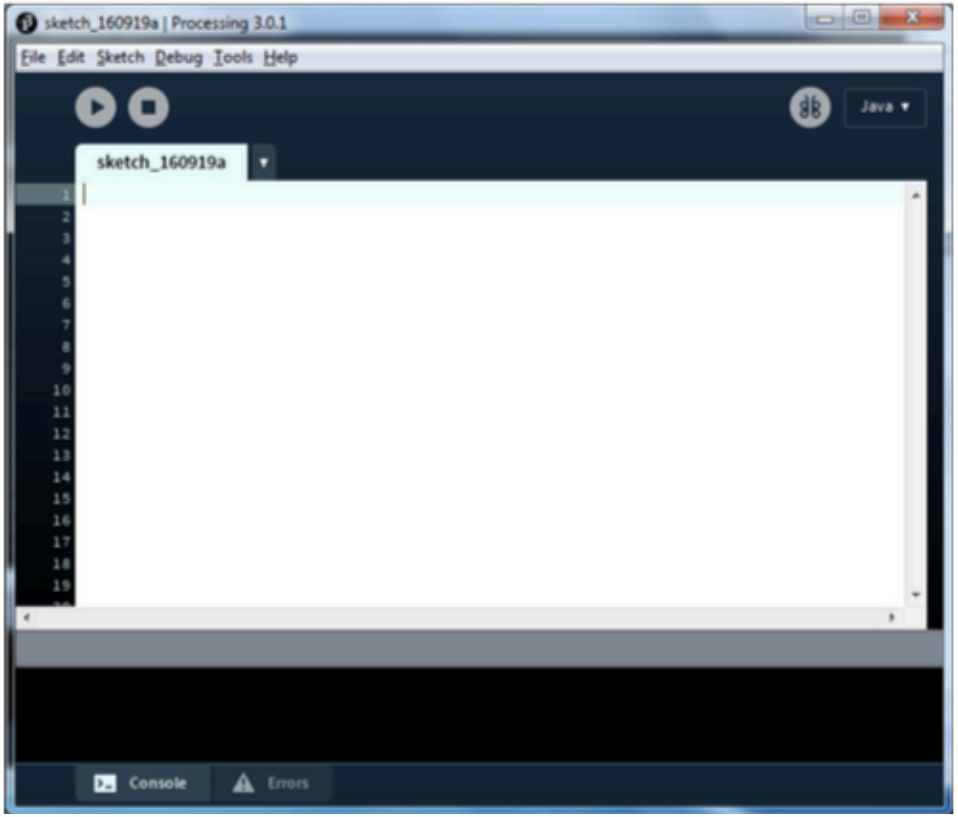
1) Dice a arduino (sketch adc_read_5_2019) di leggere i valori dell'adc ad una frequenza di 10 kHz
2) arduino scrive i valori (800 punti) in un buffer interno
3) alla fine del ciclo di lettura (~ 80 ms) arduino manda tutti i valori a processing via porta seriale
4) Processing calcola la DFT con l'algoritmo della Fast Fourier Transform
5) Fa il grafico della forma d'onda acquisita e della trasformata di Fourier discreta
6) Salva tutti questi valori su un file per permettere ulteriori analisi off-line

Avvio dell'applicazione processing

Al primo uso dell'applicazione e' necessario eseguire alcune operazioni preliminari:

- Avviate Processing[3].

  Viene creata automaticamente una cartella Processing nella vostra cartella Documenti e si apre una finestra con uno sketch vuoto (Fig. 9.6). Senza scrivere nulla salvate lo sketch vuoto con un nome qualunque (Menu File –> Salva con nome...) dentro questa cartella. Notare che, come Arduino, Processing salva ogni programma in una cartella con lo stesso nome.



[3]Se non avete il collegamento sul desktop potete avviare da C:/Programmi/processing-3.0.1/processing.

Chiudete Processing.

Avvio dell'applicazione processing

Nigro

- Prelevate dal sito del docente il file ControlP5.zip. Decomprimete e installate la cartella controlP5 nella cartella Documenti/Processing/Libraries. E' una libreria necessaria per il nostro programma.

Nigro (oppure dal mio sito)

- Prelevate dal sito del docente il file adc_arduino_5.zip. Decomprimete e installate la cartella adc_arduino_5 nella cartella Documenti/Processing. Essa contiene il programma che dovrete utilizzare nell'ultima esperienza.

Uso del programma adc_arduino_5

Dopo avere collegato la scheda Arduino Uno e avviato su di essa il programma *adc_read_5* avviate *adc_arduino_5* (cliccandoci sopra): si aprirà la finestra mostrata in Fig 9.7.

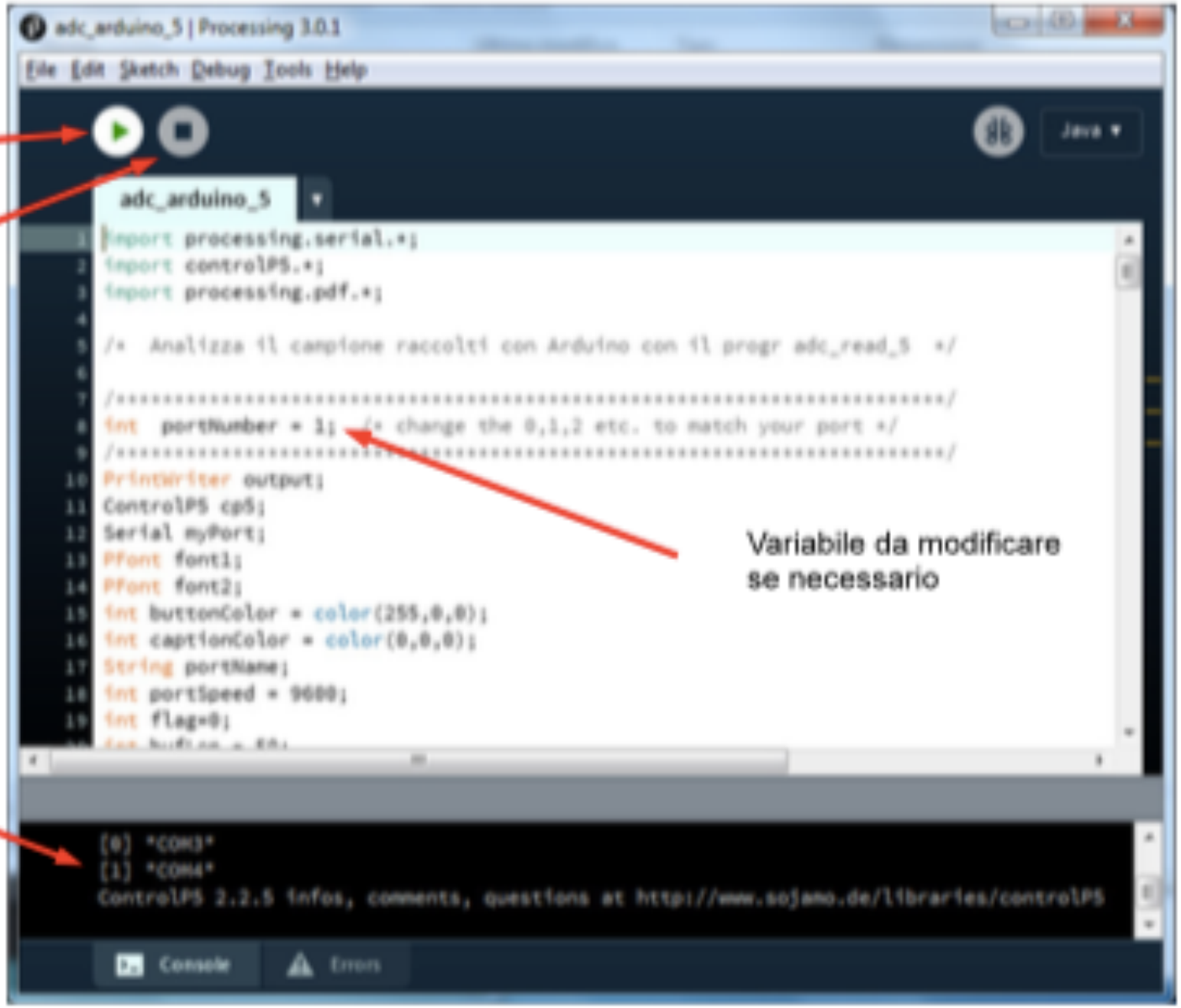
Questo programma e' pronto per essere utilizzato: premendo il bottone di avvio si apre la finestra grafica di interfaccia (Fig 9.8).

A questo punto dovete solo verificare che *adc_arduino_5* sia connesso alla stessa porta USB cui è connesso il microcontrollore: nella finestrella di controllo in basso (Fig 9.7) appare l'elenco di tutte le porte USB disponibili e per default il programma e' connesso alla porta 1 (nell'esempio corrisponde a COM4). Se necessario, modificare la variabile *portNumber* nel codice e salvare la modifica (Menu File -> Salva).

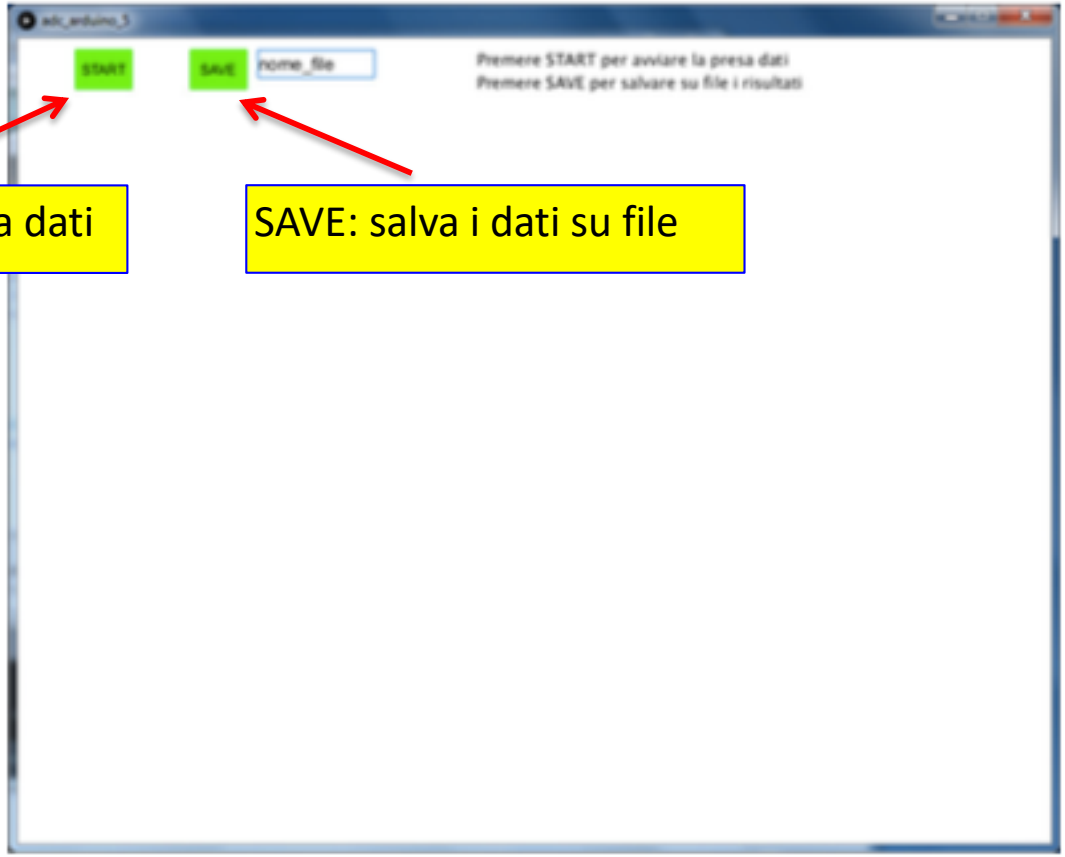
Ora potete avviare la presa dati con il bottone START. Viene inviato al microcontrollore il comando di inizio campionamento e i dati raccolti vengono restituiti a *adc_arduino_5* che produce due grafici, quello del campione e quello della trasformata[4]. Un esempio e' mostrato in Fig 9.9, dove e' stato campionata un'onda quadra con frequenza 500 Hz.

E' possibile salvare i dati su un file di testo inserendo un nome di file (per esempio *dati.txt*) e premendo il bottone SAVE. Il file verra' salvato nella cartella *Documenti/Processing/adc_arduino_5* e potra' essere successivamente analizzato con un qualunque altro programma (per esempio OpenOffice).

Finestra del programma adc_arduino_5



Interfaccia grafica di adc_arduino_5

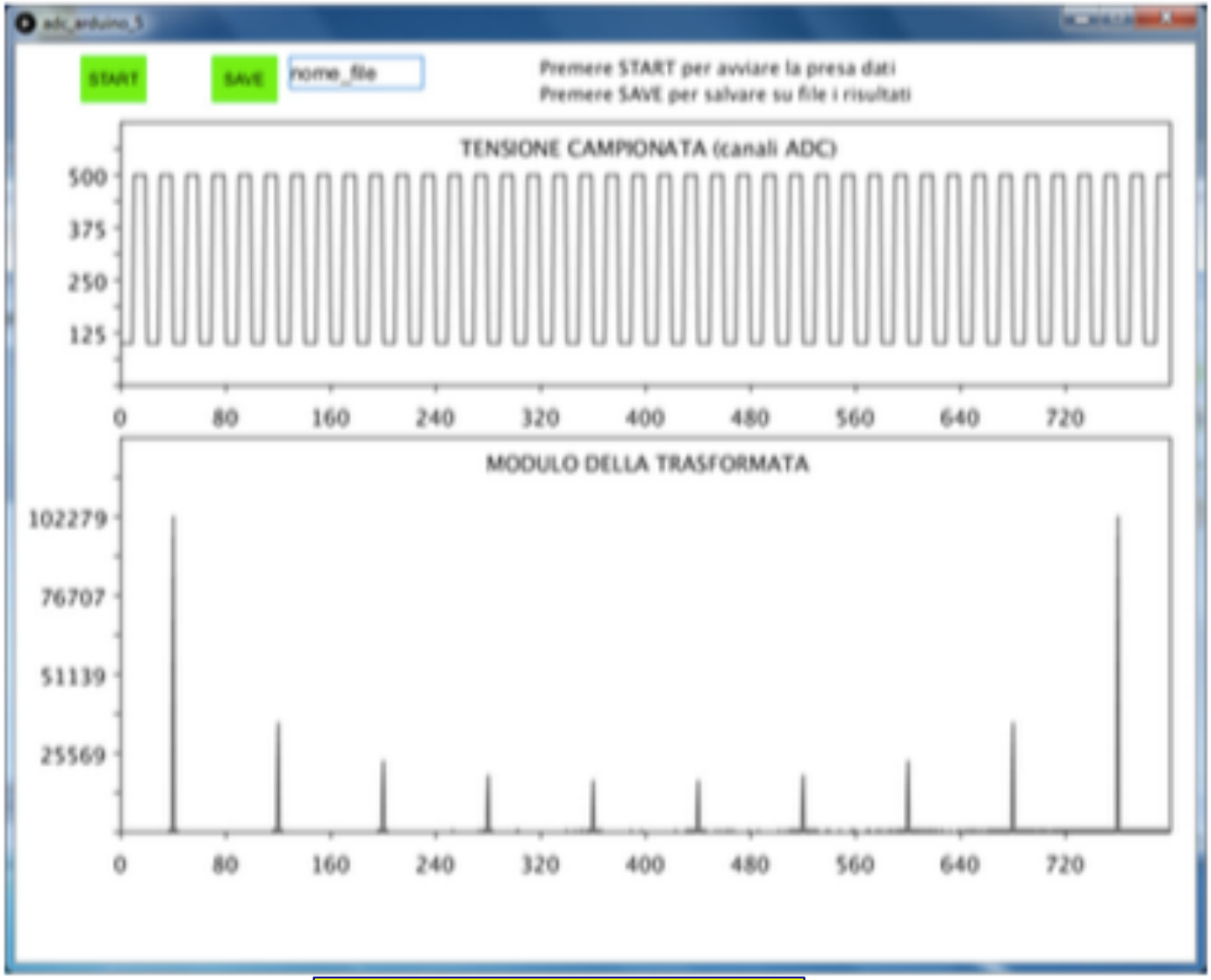


START: avvia la presa dati

SAVE: salva i dati su file

Il programma *adc_arduino_5* , quando e' in esecuzione, prende il controllo della porta USB designata e questo impedisce ad altri processi di utilizzarla. Quindi, se ad esempio e' necessario ricaricare il programma sul microcontrollore (tramite la finestra dell'applicazione Arduino), occorre arrestare *adc_arduino_5* : questo puo' essere ottenuto premendo il bottone di arresto del programma (vedi Fig 9.7).

Esempio di interfaccia grafica



Onda quadra a 500 Hz

adc_read_5_2019 (arduino)

```
int ledPin = 13;      //Collegato al led sulla scheda
int signalPin = 12;   //M. Raggi 13/07/2017

   digitalWrite(ledPin, HIGH);
```

E' acceso quando il programma di campionamento di arduino sta runnando

Va prelevato il segnale da mandare al filtro RC passa-basso

```
/* Legge ADC */
      int ii = 0;
      for (int i = 0; i < npoint; i++)
      {
       if(i==100){
          digitalWrite(signalPin,HIGH); // Apre un gate sul pin 12 digitale all'interno della finestra di lettura ADC
       }
       if(i==100+Delay)digitalWrite(signalPin,LOW);  //chiude il gate sul pin 12
      //crea un segnale di lunghezza sufficiente a caricare il condensatore del filtro RC 100KOhm 100nF
      // la tipica lunghezza e' di 110us per ogni lettura di analogread quindi con 20 letture circa 2ms

       int value  = analogRead(3);
       data[ii]   = highByte(value);
       data[ii+1] = lowByte(value);
       ii = ii +2;
        /* Serial.println(value); */
       }

/* Invia i dati a Processing */
//  return;
```

Il segnale analogico del generatore o dell'uscita del sommatore o del filtro Butterworth va mandato al pin analogico 3

Come prima cosa verificate che il pin analogico 3 di arduino funzioni. Per far questo usate il programma "voltmetro digitale": mandate una tensione continua in input e leggete il responso di arduino sulla seriale. Se non funziona, mandate il segnale su un altro pin, ma poi dovete modificare di conseguenza il programma adc_read_5_2017. Fatto cio' caricate il programma adc_read_5_2017 e non toccate piu' arduino.

Iniziamo le misure

Funzione sinusoidale

Possiamo utilizzare un segnale sinusoidale (con una frequenza inferiore alla frequenza di Nyquist), senza rumore, per verificare il buon funzionamento del sistema. Lo spettro in frequenza ottenuto sperimentalmente ci consente anche di calibrare con precisione la nostra scala di frequenza.
Possiamo poi studiare il fenomeno dell'*aliasing* utilizzando segnali sinusoidali con frequenza superiore a quella di Nyquist, verificando la validità della relativa formula teorica.

Onda quadra

Studiare lo spettro in frequenza di un'onda quadra (con una frequenza inferiore alla frequenza di Nyquist) e verificare che esso corrisponde a quanto atteso teoricamente, individuando i picchi delle armoniche e quelli dovuti all'*aliasing*

Segnale sinusoidale

- Misurate con Arduino e Processing, senza aggiungere il rumore, lo spettro di una sinusoide generata con il generatore di segnali:
  - Collegate il BNC del generatore alla breadboard
  - Utilizzate un filo per connettere il segnale dalla bradboard al ANALOG IN A3 di Arduino
  - Collegate il GND della beadboard a quello di Arduino

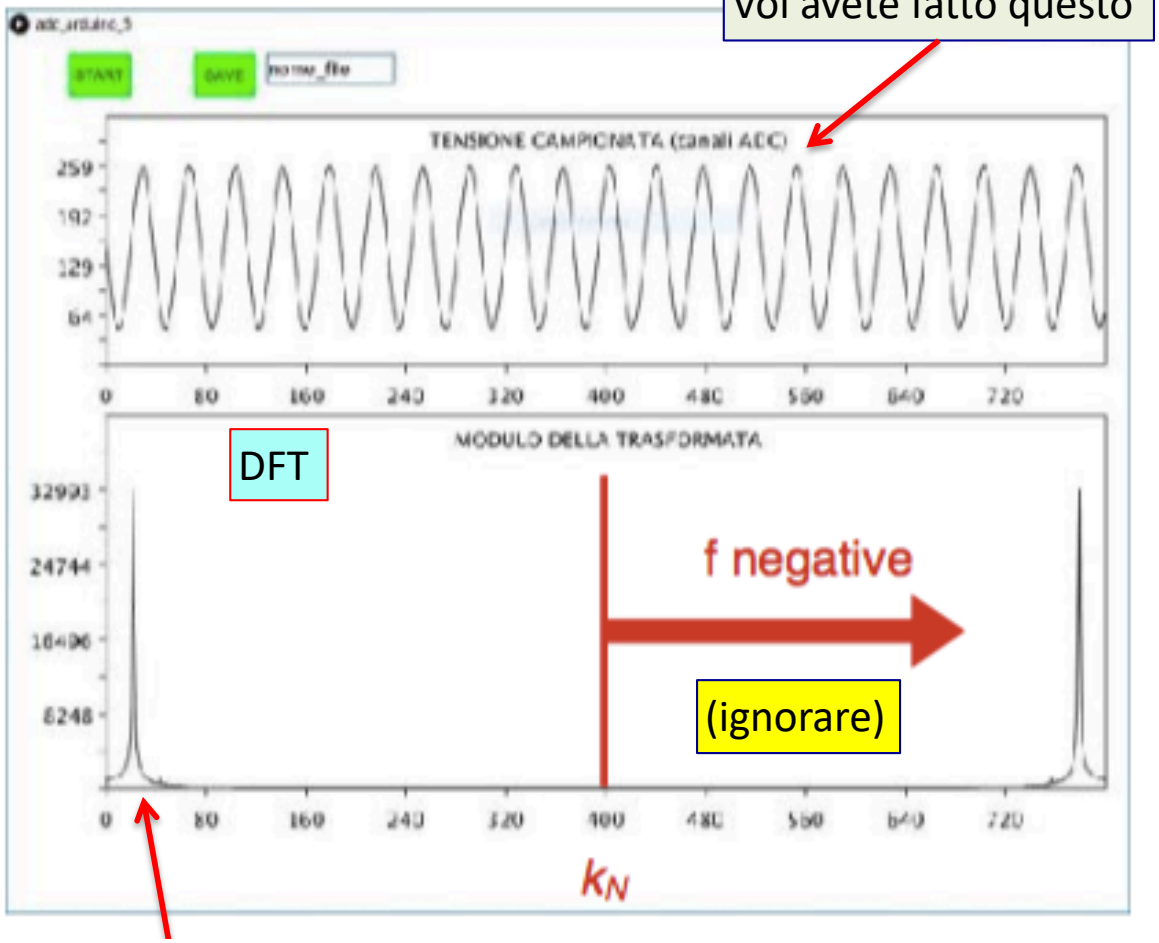
- **Inizialmente scegliete una frequenza inferiore a quella di Nyquist dell'ADC (10 kHz/2 = 5.0 kHz)**
- Studiate poi il fenomeno dell'aliasing utilizzando segnali sinusoidali con frequenza superiore a quella di Nyquist.
- Effettuare lo stesso studio con un'onda quadra.

Lo start viene dato da processing

Sinusoide a 200 Hz

- Caratteristiche Arduino
  - 1024 conteggi, 0-3.3 V
  - 800 campionamenti a circa 10 kSPS

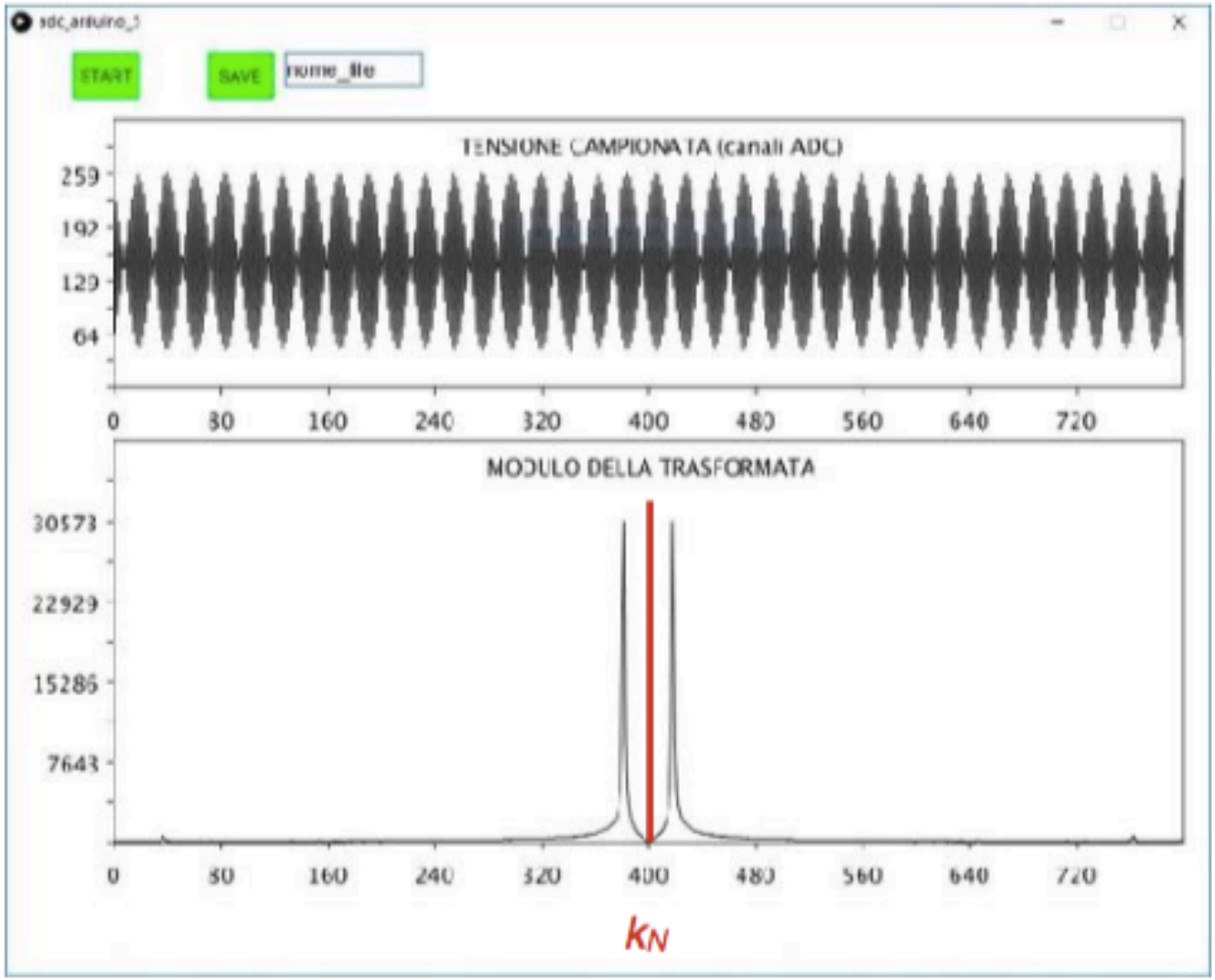
    Sample Per Second
- Pertanto
  - Volt / count = 3.3 / 1024 = 3.22 mV / count
  - N = 800
  - f_s = 10 kHz
  - T = N / f_s = 80 ms
  - $\Delta t = 1 / f_s$ = 100 µs
  - $\Delta f = f_s / N$ = 12.5 Hz
  - f_N = 5 kHz
  - k_N = 400



- $k_{segnale}$ = 200 Hz / 12.5 Hz = 16

f=K × 12.5 Hz

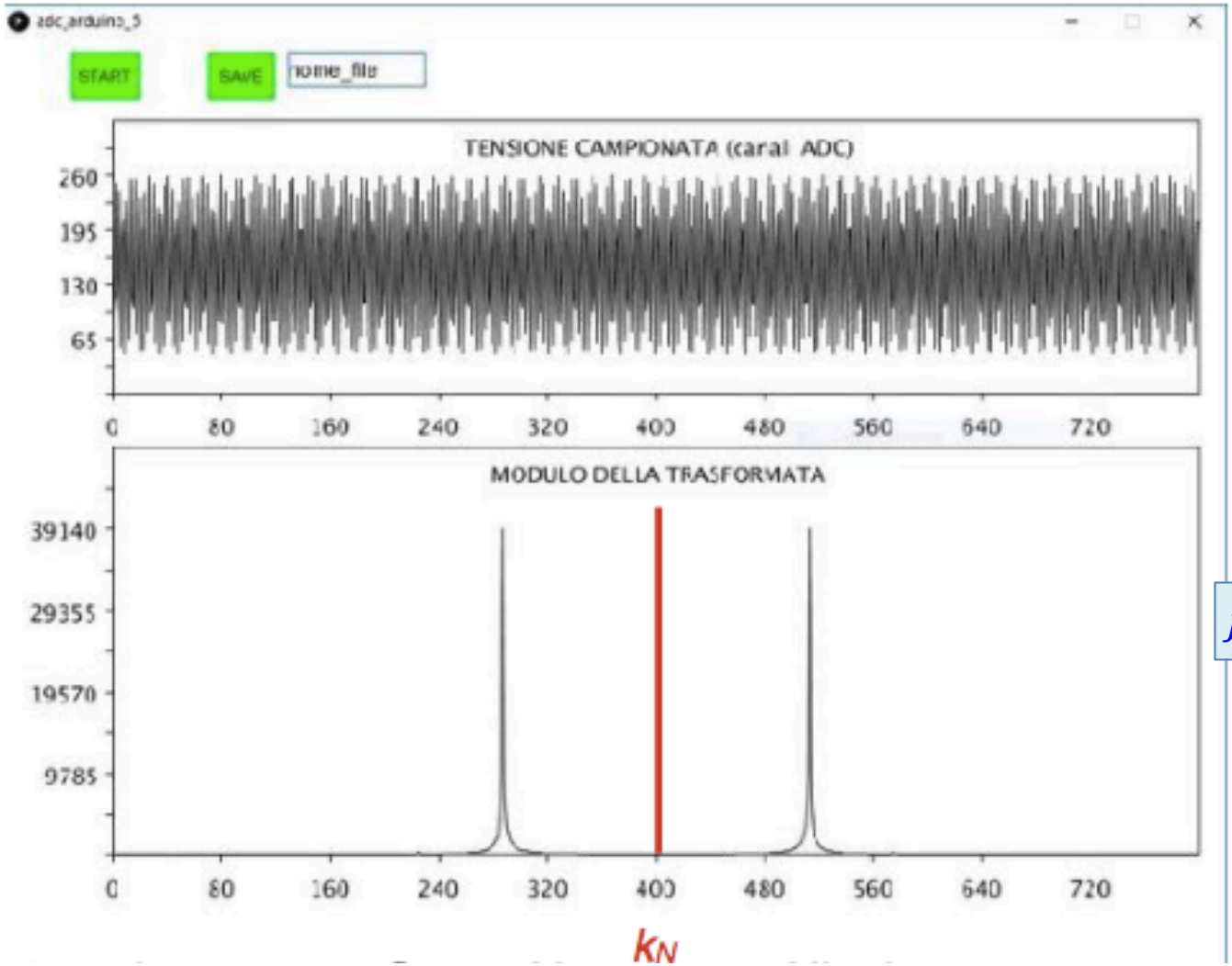
Sinusoide a 4.2 kHz



- $k_{segnale}$ = 4200 / 12.5 Hz = 336

Fare questa misura per 3-4 valori di frequenze minori della frequenza di Nyquist e per 3-4 valori al di sopra

Segnale a 6 kHz

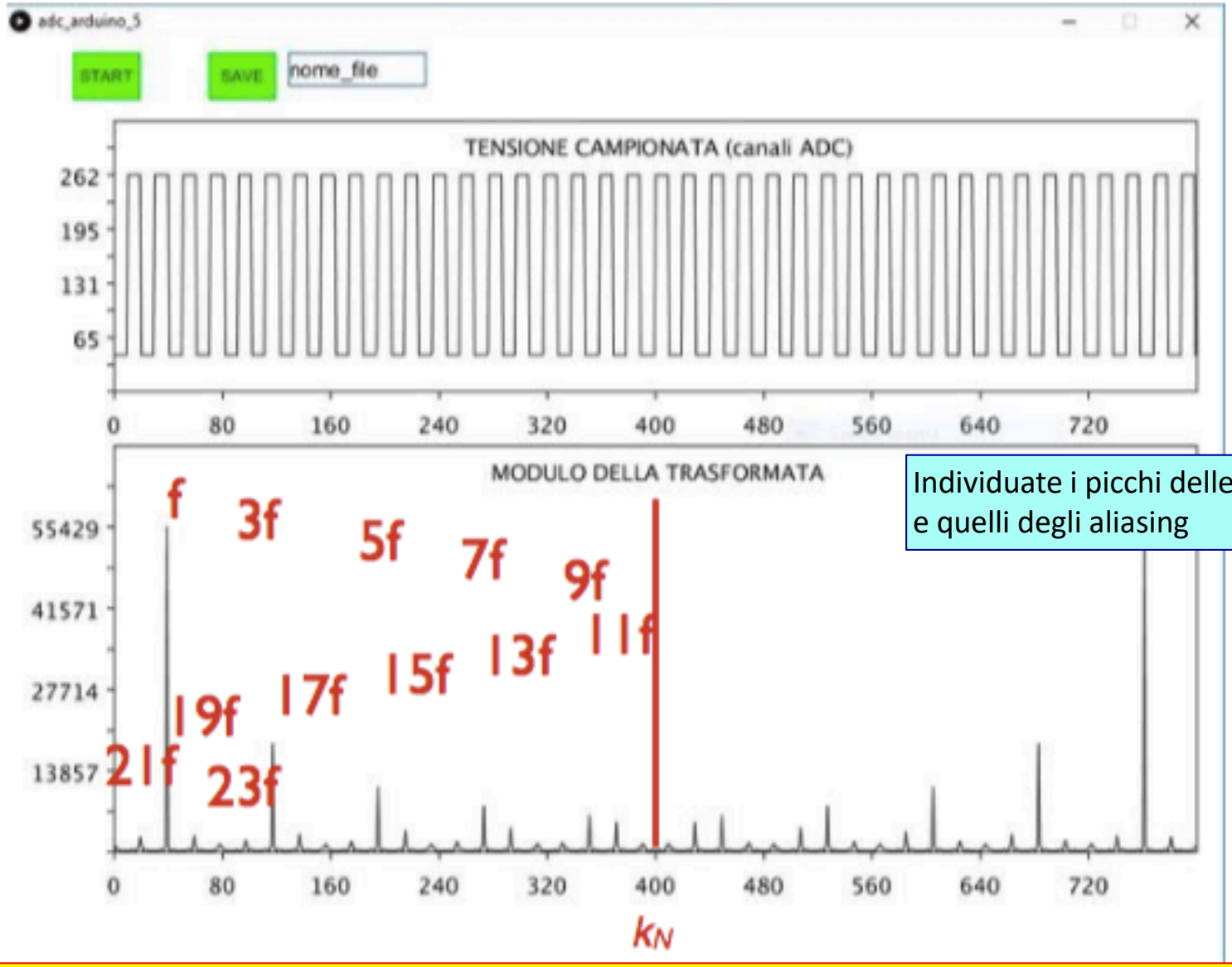


$f_a = N \times f_N - f_s$

N=2

- $k_{segnale}$ = 6000 / 12.5 Hz = 480 Sopra Nyquist → Aliasing
- $k_{apparente}$ = 400 - 80 = 320 (si tratta come un rimbalzo su k_N)

Onda quadra a 450 Hz



Individuate i picchi delle armoniche e quelli degli aliasing

Provate a far passare il segnale attraverso il filtro Butterworth prima di mandarlo ad arduino. Se taglia tutte le armoniche (f_t=1 kHz), provate con un'onda quadra di 150 Hz (con e senza filtro)

Misura del rumore

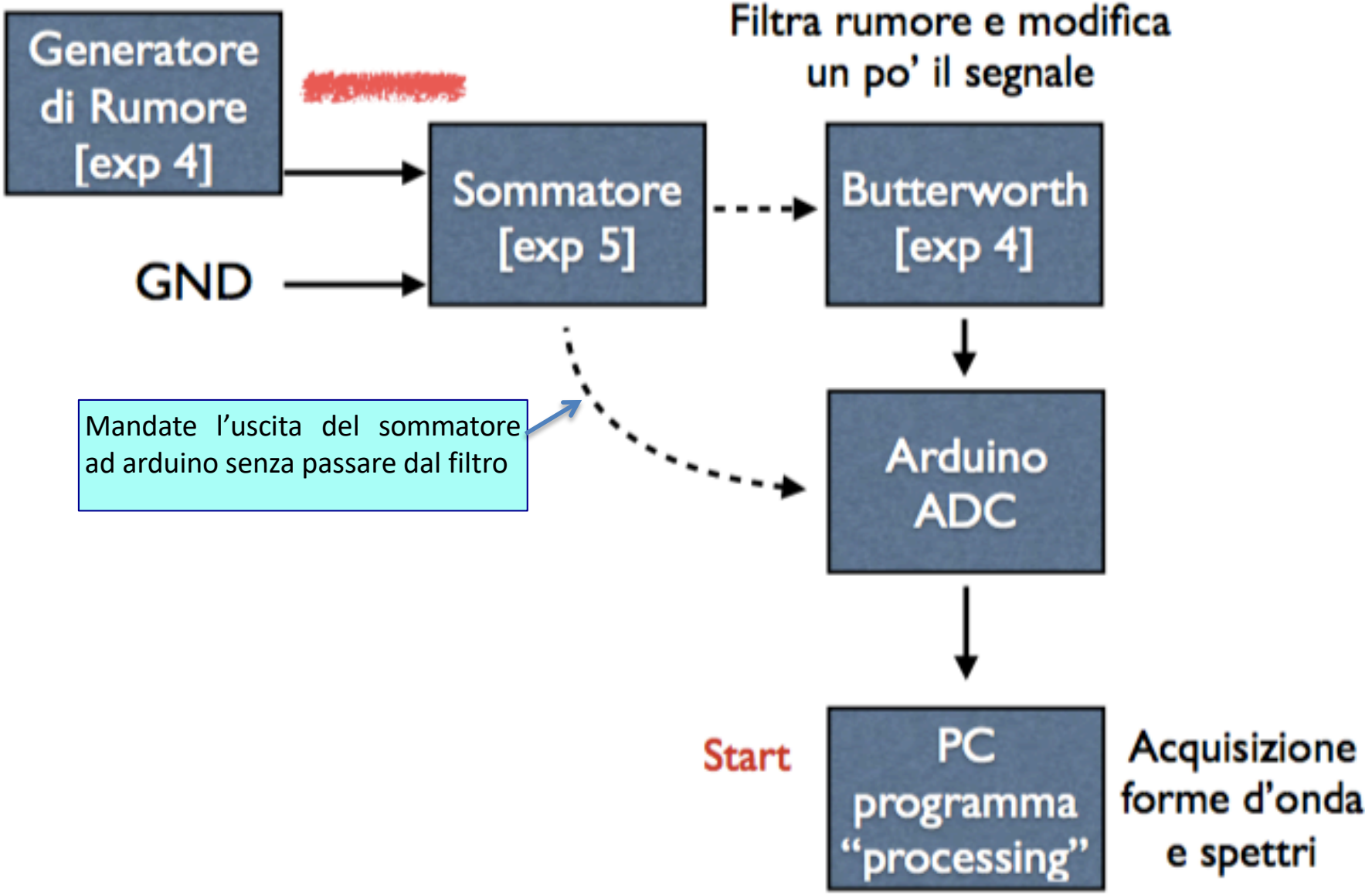
Rumore

E' interessante studiare l'andamento in frequenza del rumore da solo. Data la sua natura casuale è sensato prendere più campioni $(5 \div 10)$ per poi ricavare lo spettro di rumore S facendo la media dei moduli quadri della trasformata:

$$S_k = \sqrt{\frac{\sum_{m=0}^{M-1} |X_k|^2}{M}} \tag{9.1}$$

dove m e' il numero di segnali acquisti. Si dovrebbe verificare che questo spettro è sostanzialmente piatto. Si può procedere salvando ogni campione su un file diverso e poi, off-line, fare la media (con un programma in c, oppure con Open Office). Collegare i segnale al filtro e confrontare i risultati ottenuti.

Misura del rumore



Misura del rumore

- Studiate l'andamento in frequenza del rumore da solo staccando il segnale dal sommatore e mettendolo a massa. (in realta' ancora non abbiamo collegato il segnale)
- Data la sua natura casuale è sensato prendere piu' forme d'onda (M = 5 ÷ 10) per poi ricavare lo spettro facendo la media dei moduli quadri della trasformata.

$$S_k = \sqrt{\frac{\sum_{m=0}^{M-1} |X_k|^2}{M}}$$

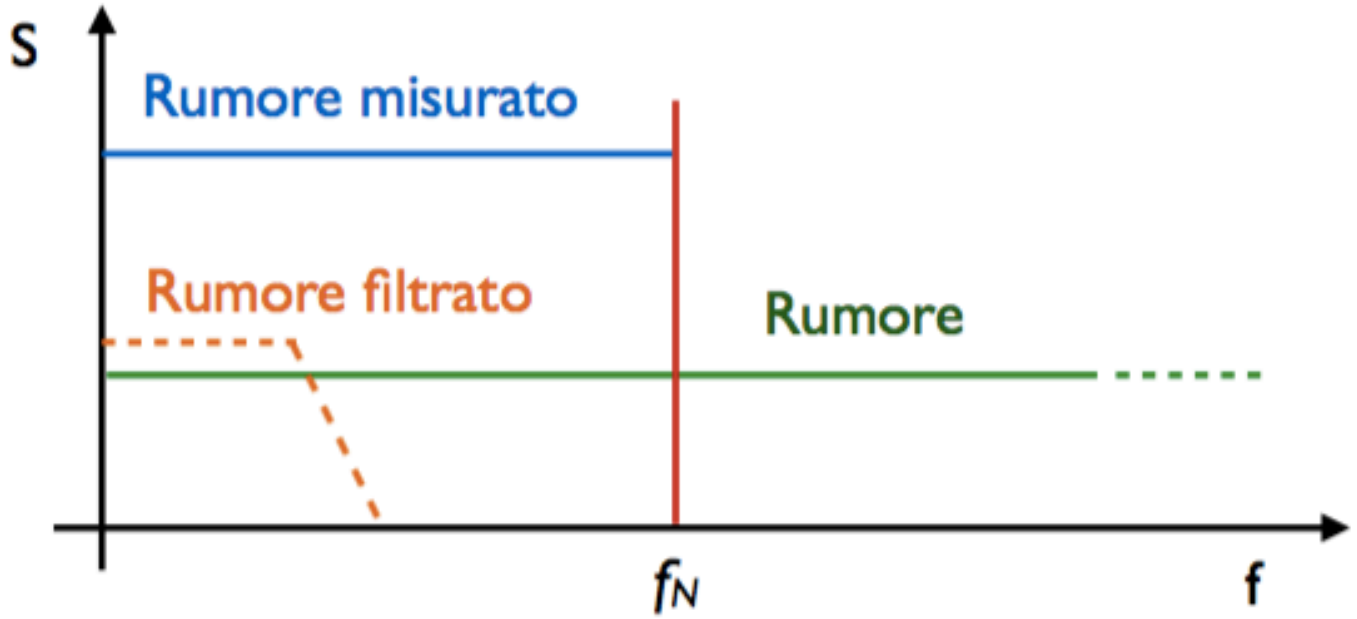
Spettro di potenza del rumore. K va da 1 a 800

  - Con Processing salvare i dati in formato txt [i, (V(i) Re(i), Im,(i), Mod(i)]
  - Salvare ogni campione *m* su un file diverso e poi, off-line, fare la media (con un programma in c, oppure con Open Office).

➢ Aggiungete un offset al rumore per fare in modo che sia sempre positivo

- Effettuate questa misura con e senza filtro passa basso. Ricordatevi nel confronto che il filtro ha un guadagno. 1.6

Cosa aspettarsi

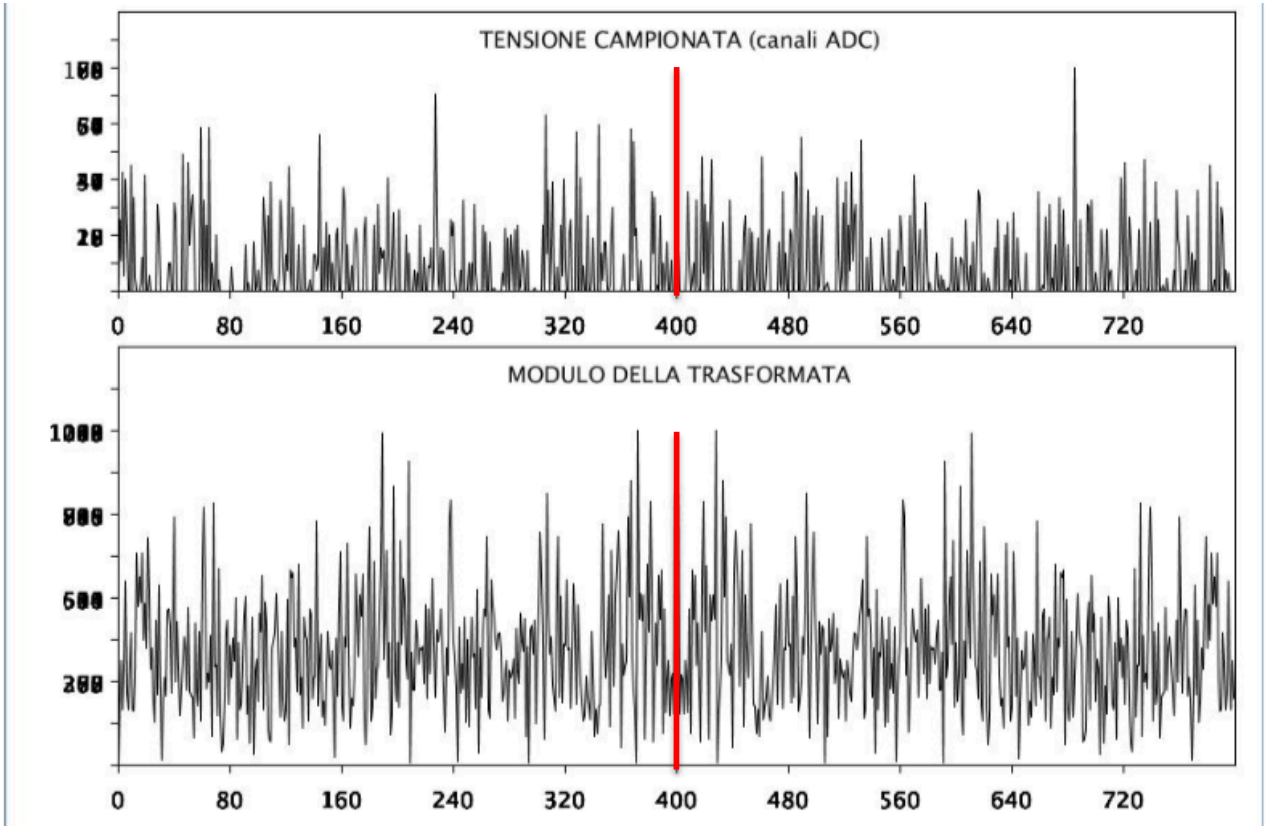


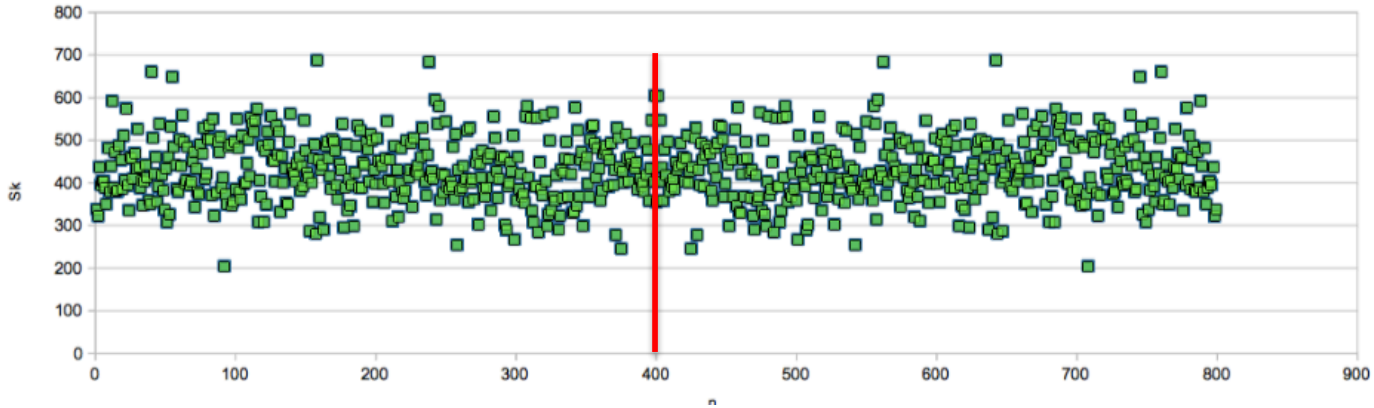
- Perché il rumore misurato ha un livello più alto di quello originale?
- Se avete tempo convertite i grafici da indice k a frequenza, in questo modo riconoscerete facilmente il taglio del Butterworth.

Misura del rumore (relazione 2016)

Senza filtro

Spettro piatto



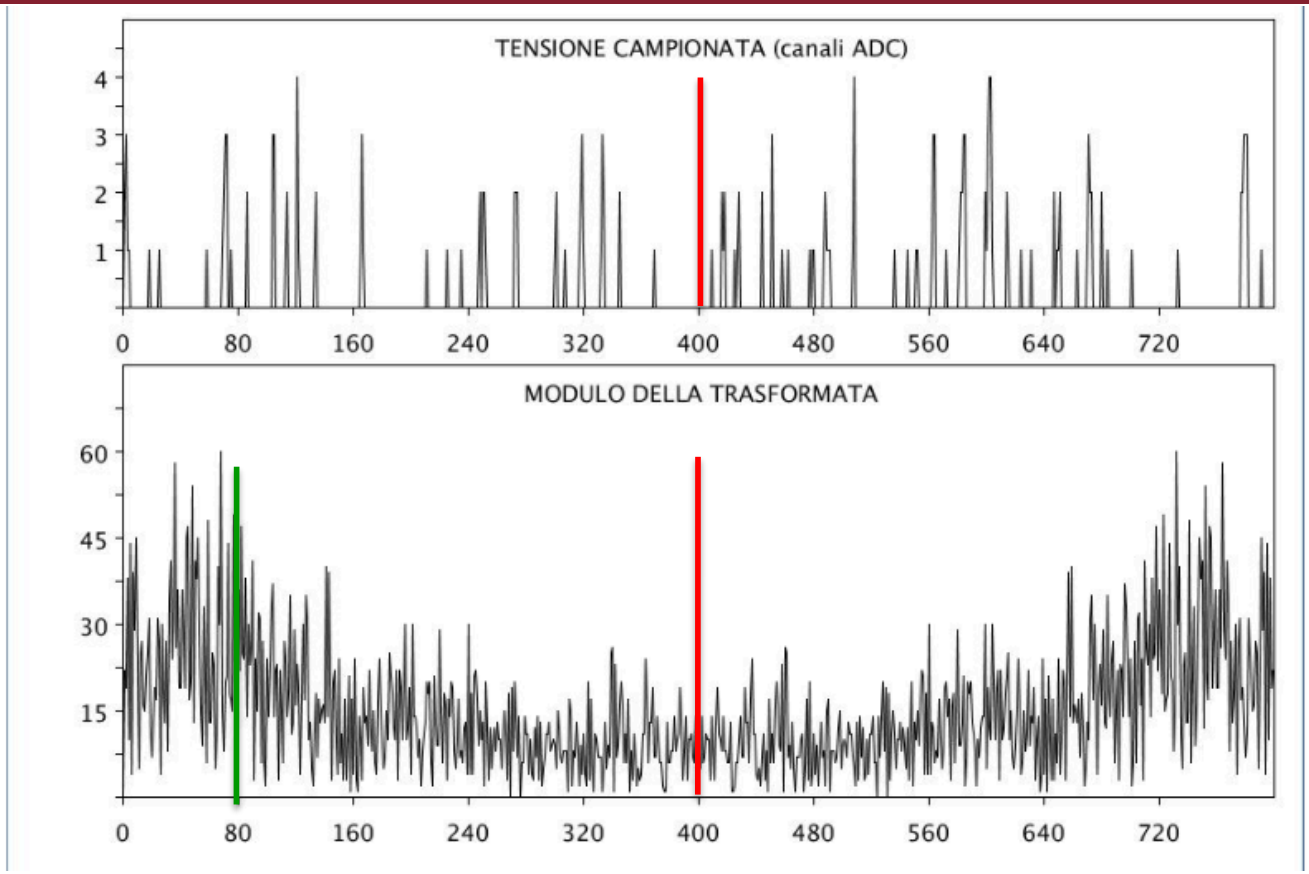


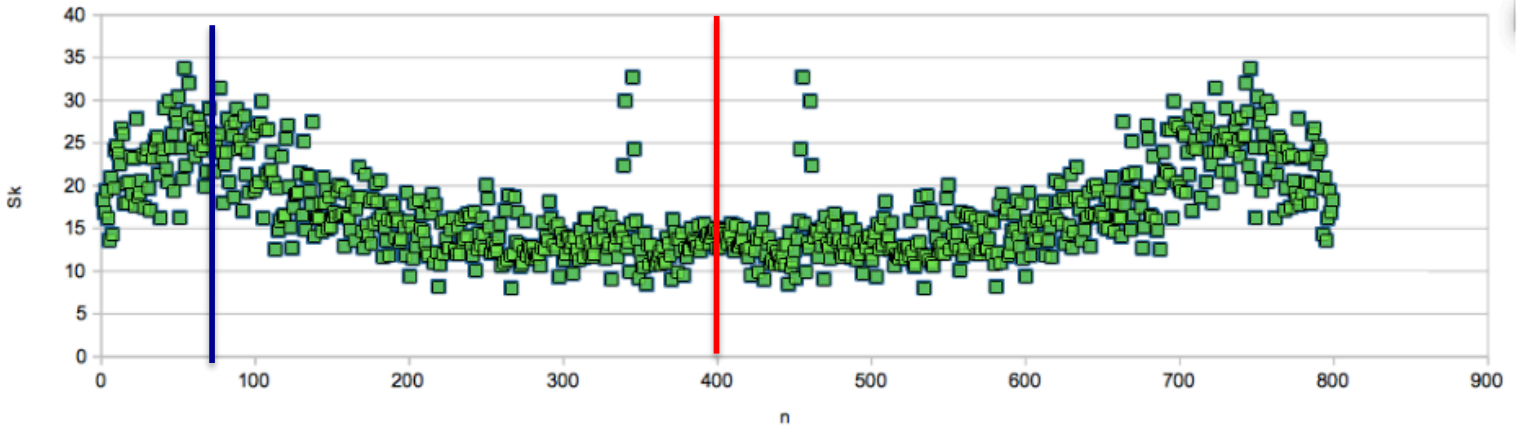
Misura del rumore (relazione 2016)

Con filtro

Spettro piccato
a basse frequenze

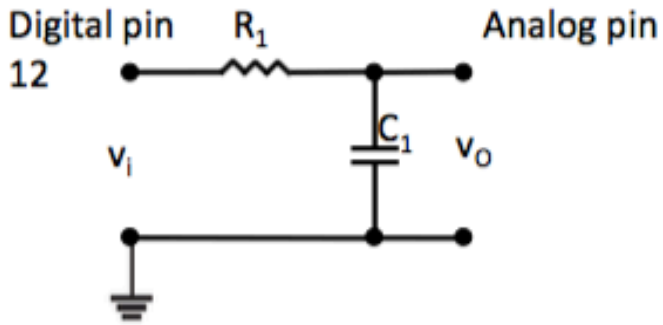
k_T=1000/12.5=80





Segnale impulsivo

Impulso singolo (da rifare) (parte diversa rispetto a due anni fa)



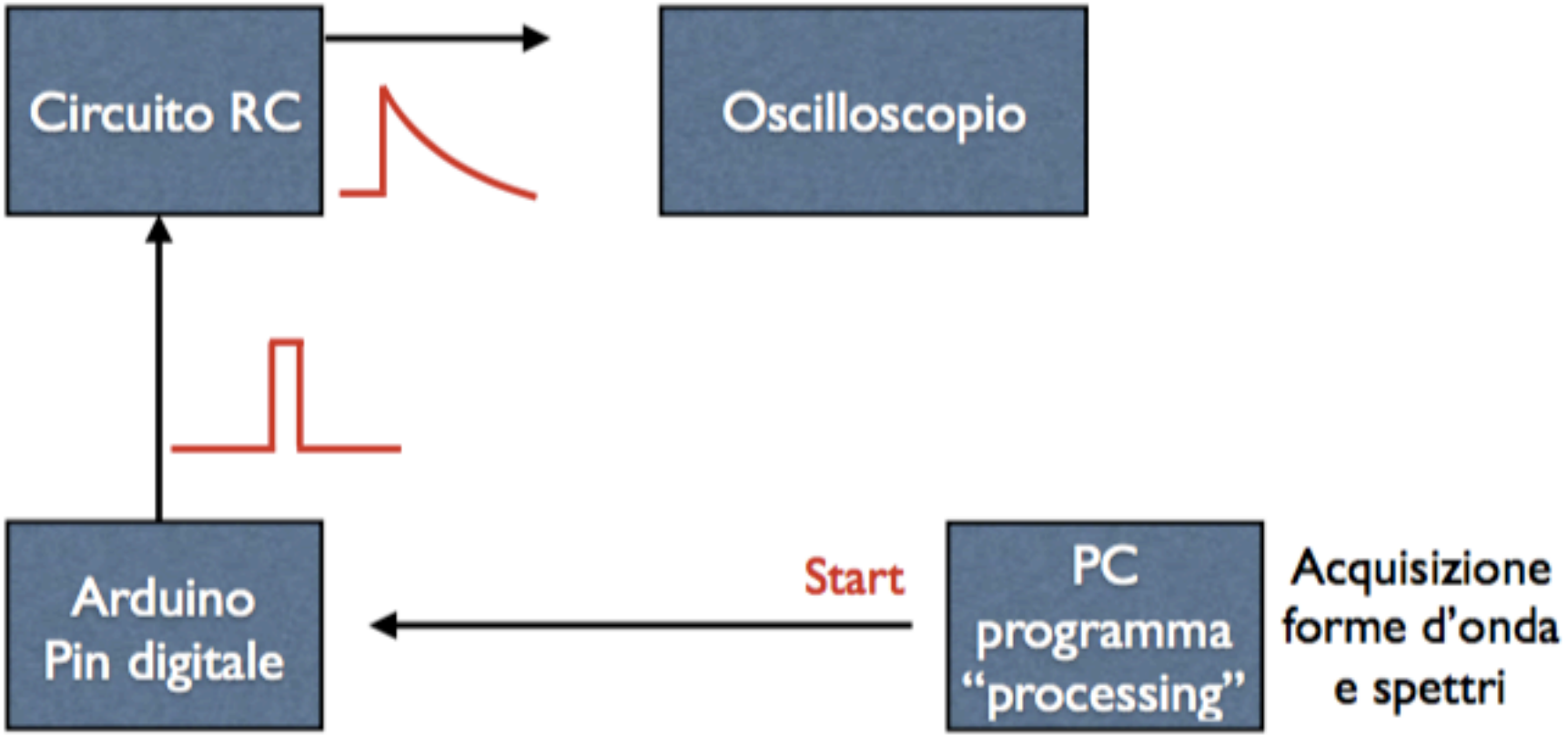
R=100 kΩ
C=100 nF
tau=10 ms

Figura 9.2: *Circuito RC a cui connettere Arduino*

Come detto, utilizzeremo Arduino per produrre un segnale impulsivo digitale che invieremo ad un filtro RC con costante di tempo $\tau \sim 10$ ms mostrato in Fig. 9.2. Nel programma *adc_read_5_2019* viene generato un segnale impulsivo di forma quadrata della durata di $\sim 2.2ms$.

Dopo aver visualmente verificato che il programma funzioni possiamo acquisire dati utilizzando i programmi *adc_read_5_2019* e *adc_arduino_5*. Osserviamo prima il segnale prodotto da Arduino collegando il pin di uscita digitale 12 ad uno degli ingressi analogici (pin 3 di default). Successivamente colleghiamo l'uscita del circuito RC ad Arduino ed osserviamo la risposta del circuito ed acquisiamo una forma d'onda di riferimento.

Creazione di un segnale impulsivo



- Controllare il segnale in uscita dal pin digitale di Arduino.
- Controllare il segnale in uscita dal circuito RC (100 kΩ x 100 nF = 10 ms)
- Se i segnali sono brutti probabilmente non avete connesso i GND.

Forma attesa del segnale impulsivo

- Da Arduino esce un'onda quadra molto stretta, circa ΔT = 2 ms, con ritardo di circa 10 ms rispetto allo start dell'acquisizione ADC.
- Possiamo rappresentarlo come la sovrapposizione di 2 onde quadre e applicare Laplace:

$$f(t) = A[\theta(t) - \theta(t - \Delta t)] \qquad f(s) = A\left[\frac{1}{s} - \frac{1}{s}e^{-s\Delta t}\right]$$

- Moltiplicando per la funzione di trasferimento del passa basso.

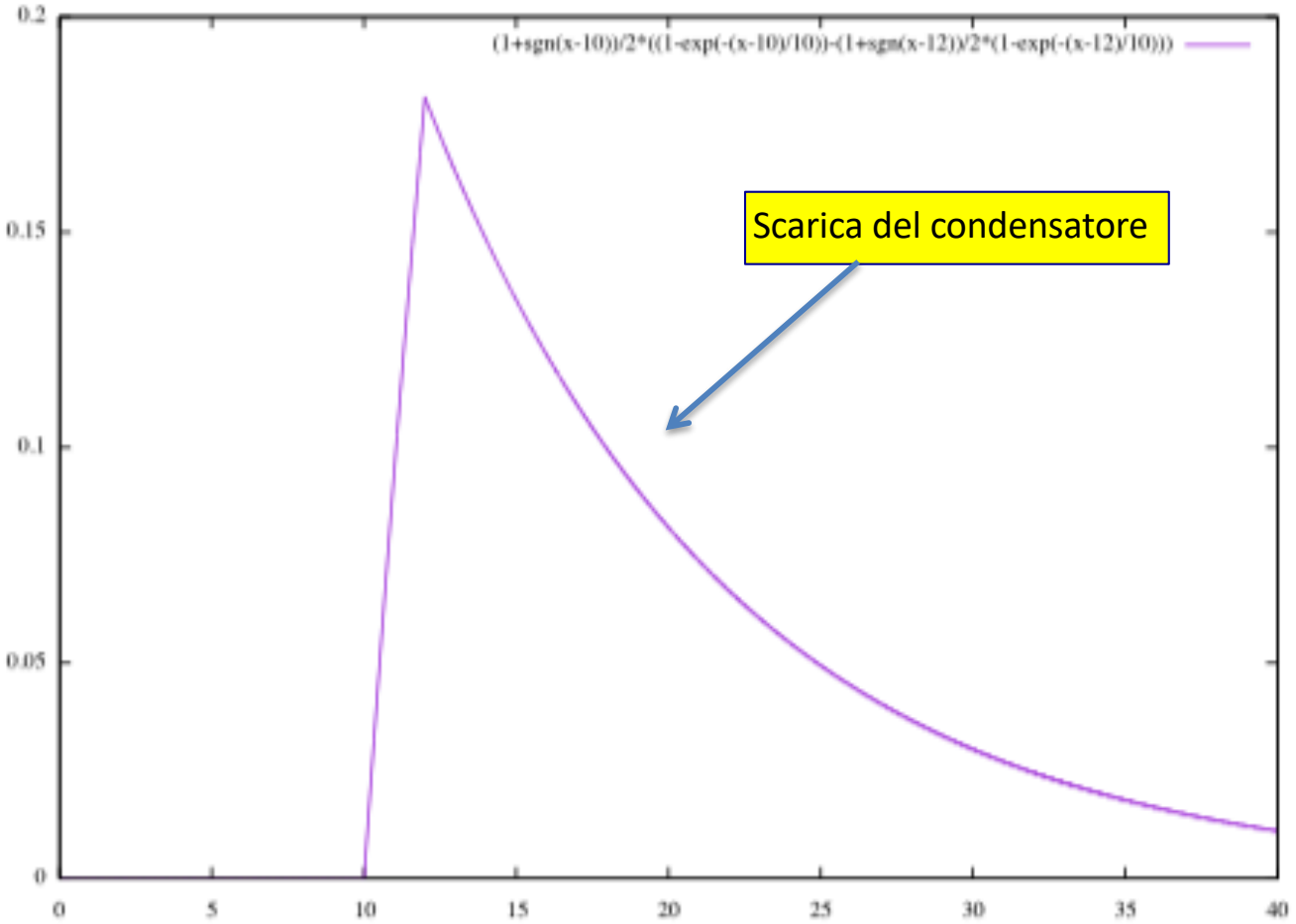
$$P(s) = f(s)\frac{1}{1 + s\tau}$$

$$P(t) = A\theta(t)[1 - e^{-t/\tau}] - A\theta(t - \Delta t)[1 - e^{-(t-\Delta t)/\tau}]$$

- Quindi in pratica l'impulso è:

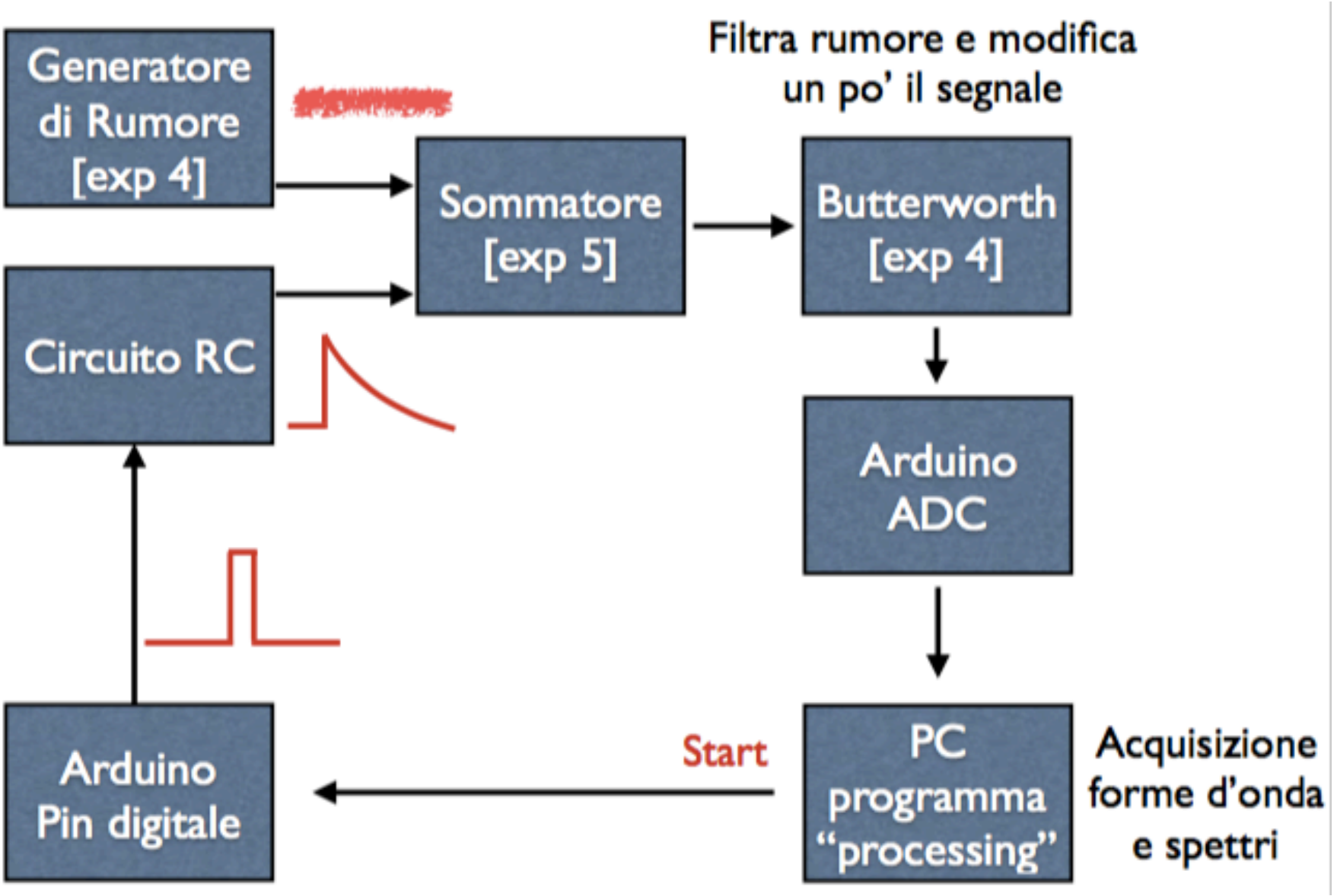
$$P(t) = \begin{cases} A[1 - e^{-t/\tau}], & \text{if } t < \Delta t \\ A[e^{\Delta t/\tau} - 1]e^{-t/\tau}, & \text{if } t >= \Delta t \end{cases}$$

segnale impulsivo simulato



- Il massimo vale A $[1-e^{-t/\Delta t}] = 0.18$ A

Circuito completo



Misure segnale + rumore

Possiamo ora simulare una situazione reale aggiungendo al segnale impulsivo una certa quantità di rumore utilizzando il sommatore di Fig. 9.1 (eventualmente aggiustando i pesi relativi) e studiare lo spettro in frequenza che ne risulta.
Nelle situazioni reali quello che si cerca di fare è di ridurre il più possibile il rumore cercando di alterare il meno possibile la forma del segnale, trovando il giusto compromesso tra queste due esigenze, mediante filtri, che vengono progettati studiando appunto lo spettro in frequenza del rumore e quello del segnale.
Possiamo cercare di fare lo stesso inserendo nel circuito il filtro passa basso VCVS già costruito scegliendone opportunamente la frequenza di taglio.

Osservando il segnale all'oscillografo possiamo valutare il risultato che otteniamo. Colleghiamo ora il circuito ad Arduino e osserviamo il risultato della misura.

Misure segnale + rumore

- Analogamente a quanto fatto per il rumore, studiate lo spettro in frequenza del segnale da solo staccando il rumore dal sommatore e mettendolo a massa.
  - Effettuate questa misura con e senza filtro passa basso.
- Attaccate segnale e rumore al sommatore.
  - Controllate l'uscita del sommatore con l'oscilloscopio.
  - Acquisite le forme d'onda nel dominio tempo con Arduino.
  - Effettuate questa misura con e senza filtro passa basso.

SAPIENZA
UNIVERSITÀ DI ROMA

Fine esercitazione 9