



```

Amp=1;
Sig_f = 1; %rad/s or Hz as set in the block;
%% Analog Filter
K = 1;
tau = 0.1; %s
%% Digital Filter
T = 0.001; %s
ris=exp(-T/tau)

```

```
ris = 0.9900
```

```

%% Arduino UNO
ADC_bit =10 ;
ADC_V = 5; %V
PWM_bit = 8;
PWM_V = 5; %V
PWM_f = [31372.55, 3921.16, 980.4, 490.2, 245.1, 122.55, 30.64]; %Hz Digital #3
% PWM Low-Pass Filter
R = 2e3; %Ohm
C = 10e-6; %F;

```

## Ricerca parametri sperimentali

La frequenza massima del PWM per la scheda Arduino equivale alla grandezza "PWM\_f(1)", scelta per ottenere transitori brevi e, di conseguenza, una sinusoide in uscita più sottile.

Per essere certi di lavorare in condizioni ottimali, è necessario che il PWM sia abbastanza veloce in proporzione e dunque che la frequenza di taglio (RC\_f) sia almeno un ordine di grandezza minore di tale valore; nel caso specifico è stata ridotta di un fattore 20.

Considerando il valore di R costante e pari a 1000 ohm, è necessario adattare il valore di C al fine di ottenere una frequenza di taglio che rispetti la condizione enunciata in precedenza. Il valore richiesto è di circa 100nF, ottenibile tramite il collegamento di 2 condensatori da 47nF in parallelo (Ceq = 94nF).

```

PWM_fmax = PWM_f(1);
RC_f = PWM_fmax / 20

```

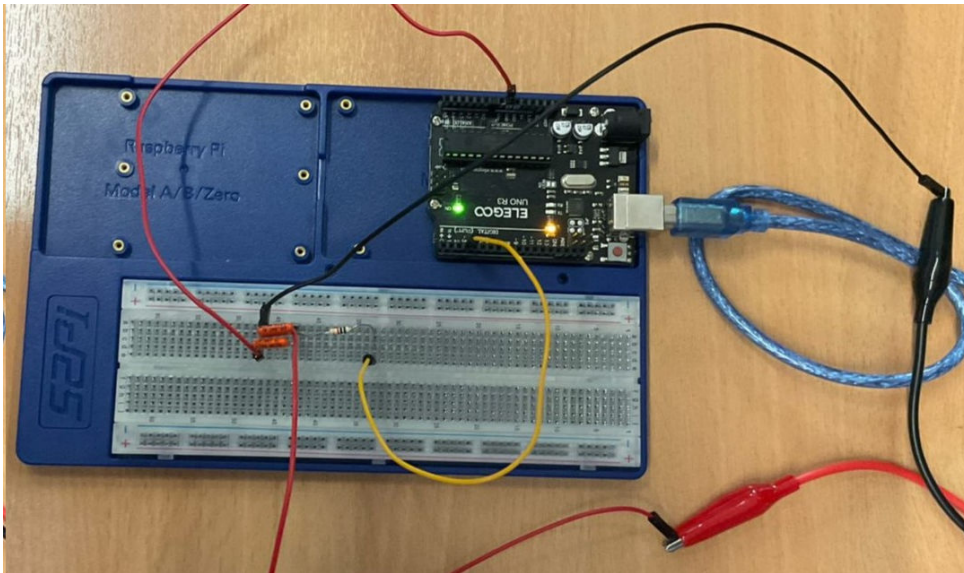
```
RC_f = 1.5686e+03
```

```

%RC_f = 1 / (2*pi*R*Ceq);
Rb=1e3; %Resistenza in laboratorio fissa
Ceq = 1/(RC_f * 2 * pi * Rb)

```

```
Ceq = 1.0146e-07
```



## Modellistica DAC

I parametri da cui dipende il circuito sono:

- frequenza di campionamento ( $\text{Camp\_f} = 1/T$ )
- frequenza del segnale ( $\text{Sig\_f}$ )
- frequenza del PWM ( $\text{PWM\_f}$ )
- frequenza di taglio del filtro analogico RC ( $\text{RC\_f}$ )

Le condizioni che devono rispettare sono:

```
PWM_f = max(PWM_f);      % Al fine di avere un transitorio il più breve possibile e
quindi una sinusoide in uscita più sottile
RC_f = 1/20 * PWM_f      % Affinchè il PWM risulti abbastanza veloce in proporzione
```

```
RC_f = 1.5686e+03
```

```
Sig_f = 1/100 * RC_f      % Almeno due decadi sotto per assicurarsi di non tagliare
il segnale
```

```
Sig_f = 15.6863
```

```
Camp_f = 50 * Sig_f      % Per rispettare il teorema del campionamento e oltre
```

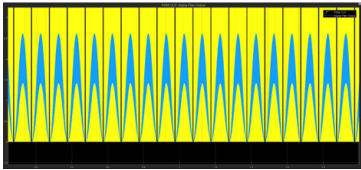
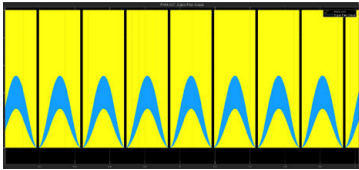
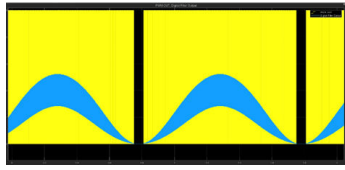
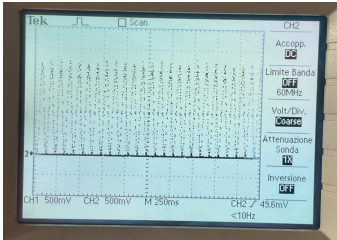
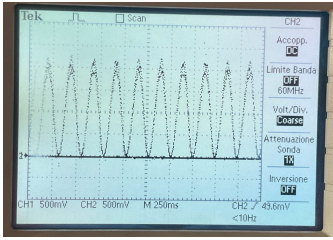
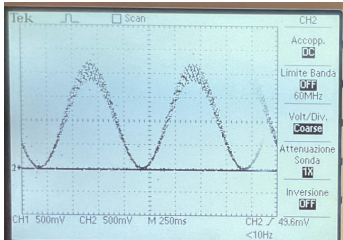
```
Camp_f = 784.3138
```

```
T = 1/Camp_f
```


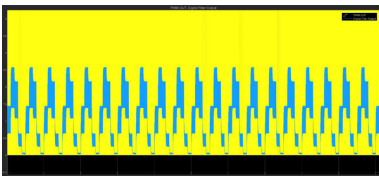
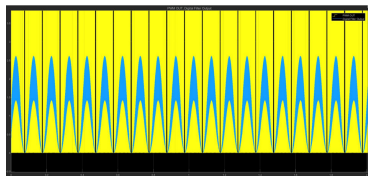
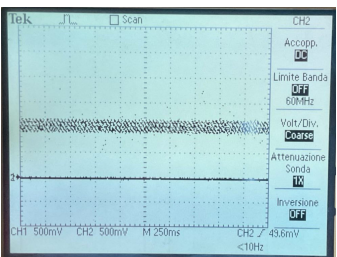
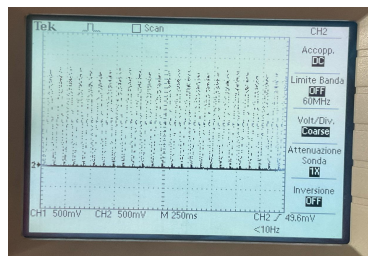
```
T = 0.0013
```

Si studiano i comportamenti del circuito al variare di tali parametri:

## Variazione frequenza del segnale ( $\text{Sig\_f}$ )

10Hz	4Hz	1Hz
	DIGITALE	
		
	ANALOGICO	
		

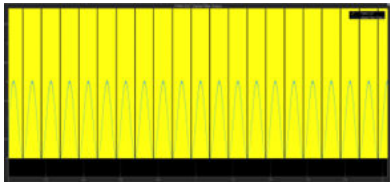
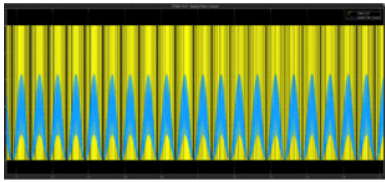
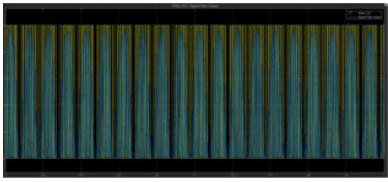
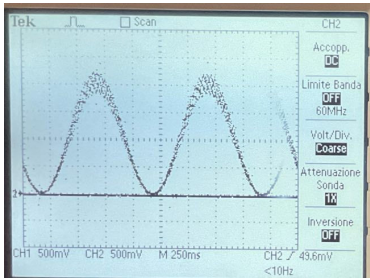
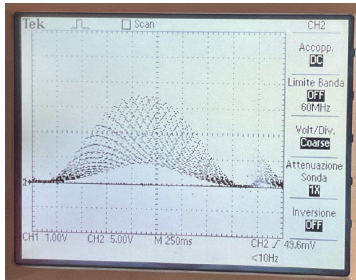
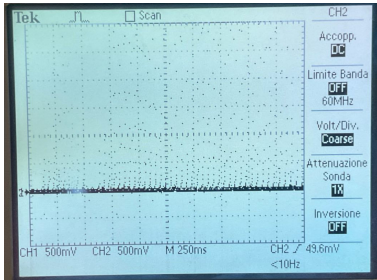
### Variazione frequenza di campionamento (Camp\_f)

10Hz	50Hz	1000Hz
	DIGITALE	
		
	ANALOGICO	
		

Con frequenza bassa si hanno pochi campioni, quindi la sinusoide è discretizzata in una sequenza di costanti: si nota che il segnale in uscita resta a regime per molto tempo.

### Variazione frequenza del PWM (PWM\_f)

31372.55Hz	490.2Hz	122.55Hz
	DIGITALE	

		
<b>ANALOGICO</b>		
		

Abbassando la frequenza del PWM si inspessisce la sinusoide in uscita perchè il periodo del PWM diventa dello stesso ordine di grandezza di TAU.


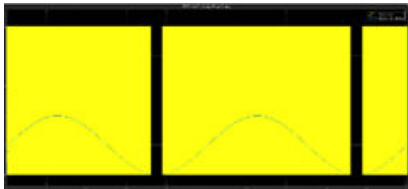
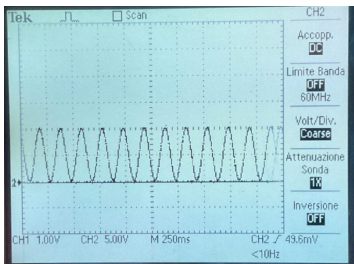
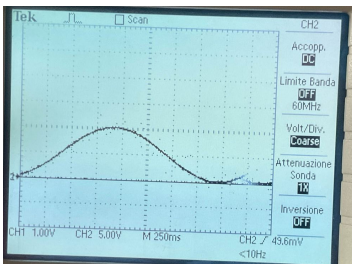
## Variando TAU filtro analogico

$C_{new} = 4.7e-6;$   
 $\tau_{2} = R_b \cdot C_{new}$

$\tau_{2} = 0.0047$


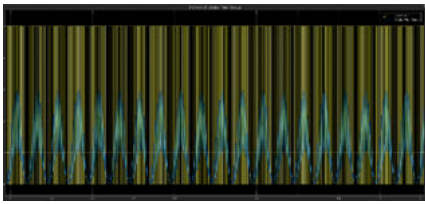
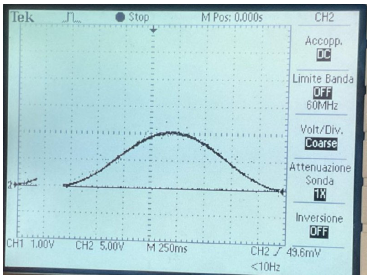
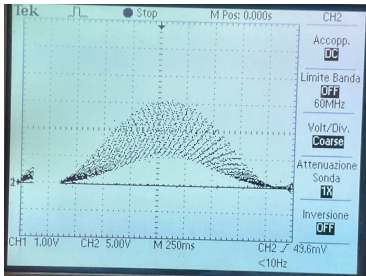
Ripetiamo lo studio al variare dei parametri sul circuito con la nuova TAU

## Variazione frequenza sel segnale (Sig\_f)

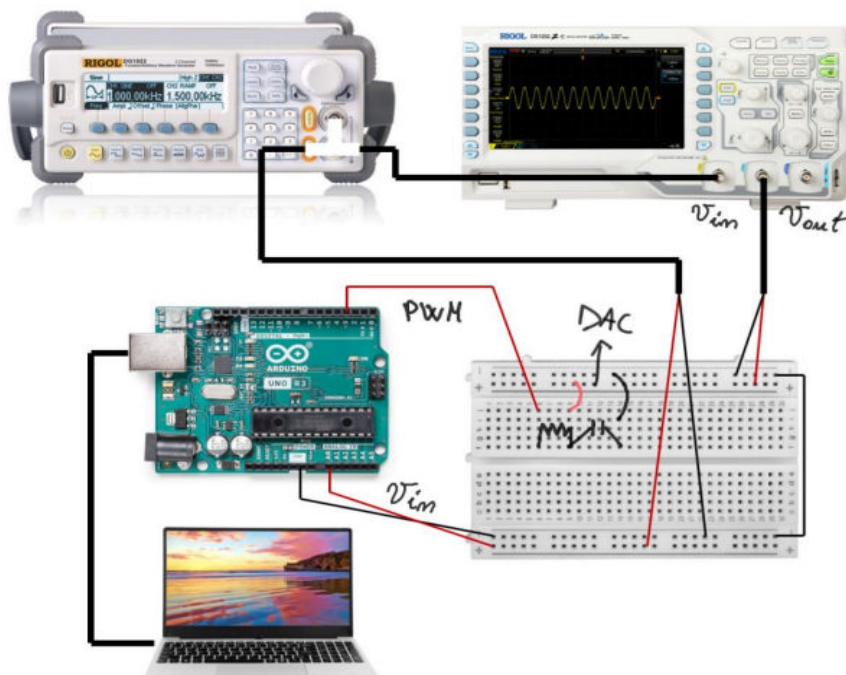
10Hz	1Hz
DIGITALE	
	
ANALOGICO	
	



## Variazione frequenza del PWM (PWM\_f)

3921.16Hz	122.55Hz
ANALOGICO	
	
DIGITALE	
	

## Arduino come ADC - DAC

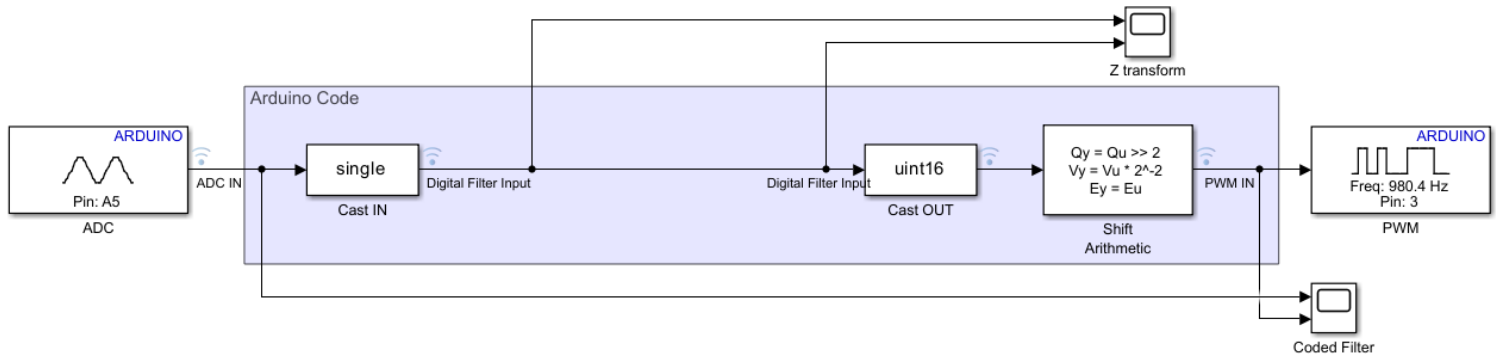


A differenza del modello DAC, ora il segnale viene acquisito dal generatore d'onda.

Il circuito equivalente rimane lo stesso della modellistica DAC, la differenza consiste nell'implementare ora sia l'ADC che il DAC.

Inizialmente si prova il modello senza filtro digitale, aggiungendolo in un secondo momento.

## ADC senza Filtro Digitale



Utilizzando il seguente modello Siimulink viene riprodotta una funzione "echo".

Teoricamente il segnale di uscita dovrebbe corrispondere al segnale d'ingresso, nella pratica invece si nota un ritardo di fase tra i due segnali.

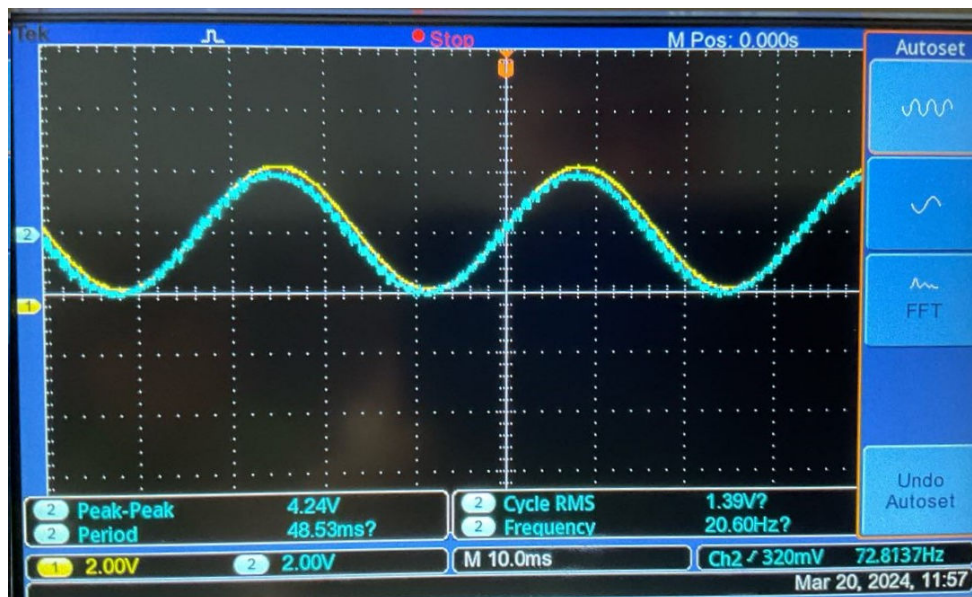
Questo perchè quando si utilizza Arduino per implementare un convertitore digitale-analogico (DAC), è fondamentale considerare il tempo necessario per eseguire le operazioni richieste.

Se il periodo di campionamento (T) viene ridotto troppo, Arduino potrebbe non essere in grado di completare tutte le operazioni in tempo a causa delle sue limitate capacità di elaborazione.

Quindi, c'è un limite minimo al periodo di campionamento (T) determinato dalla velocità di elaborazione di Arduino.

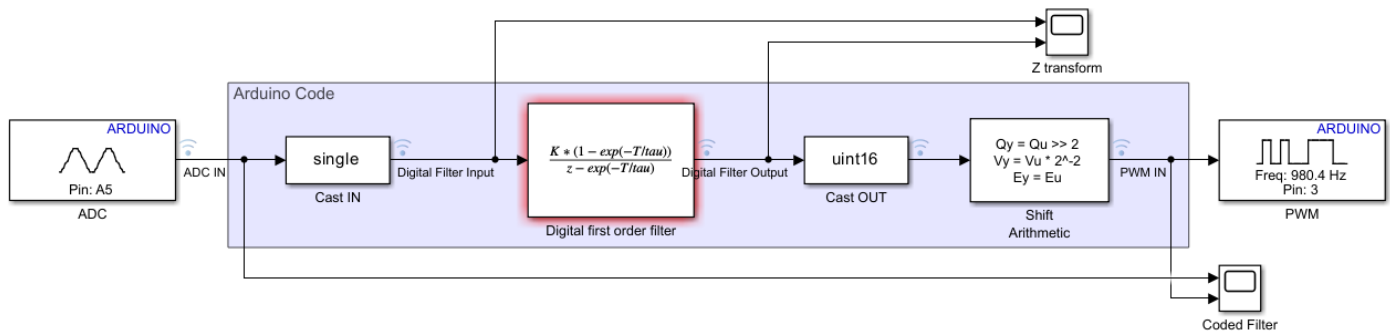
Superare questo limite potrebbe portare a risultati errati o a un funzionamento non ottimale del sistema.

Bisogna bilanciare la velocità di campionamento desiderata con le capacità computazionali di Arduino per garantire un funzionamento affidabile del sistema.

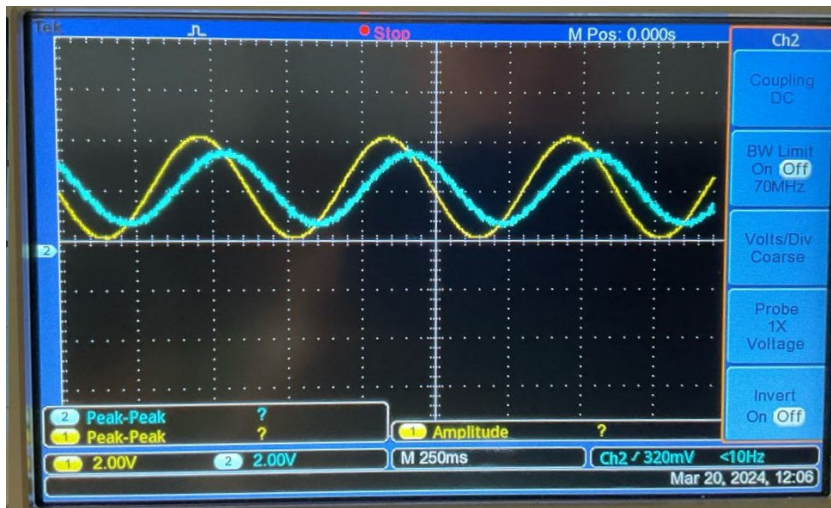


## ADC con Filtro Digitale

Successivamente si inserisce di nuovo il filtro digitale.

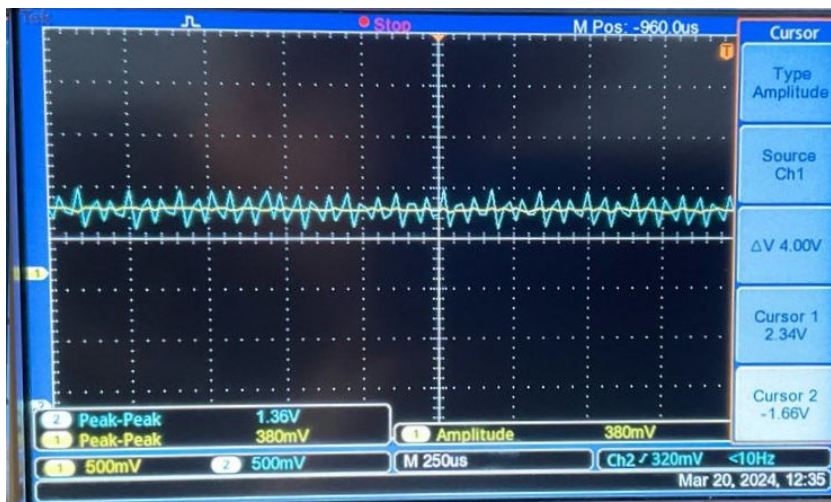


Si nota come la frequenza di taglio ora corrisponde alla  $f_c$  del filtro digitale, questo perchè tra i due filtri ha la prevalenza il filtro digitale, avendo una banda passante più piccola ( $f_c = 1,6 \text{ Hz}$ )



L'immagine successiva ritrae l'effetto ondulatorio del convertitore digitale-analogico realizzato tramite PWM seguita da un filtro passa basso (LPF).

Regolando il parametro Tau del filtro, si può diminuire l'effetto ondulatorio a discapito di un ritardo del segnale.





# Confronto Filtro Analogico vs Digitale

## FILTRO ANALOGICO

Si crea analogicamente un filtro passa-basso RC la cui funzione di trasferimento coincide con quella data.

```
s=tf('s');  
tau = 0.1; %costante di tempo del sistema  
G = 1/(1+s*tau)
```

G =

$$\frac{1}{0.1 s + 1}$$

Continuous-time transfer function.  
Model Properties

La frequenza di taglio è :

```
fc=1/(2*pi*tau) %[Hz]
```

fc = 1.5915

Si nota che è la stessa tau del filtro digitale implementato precedentemente, così permettendo un diretto confronto tra i due.

Analizzando il comportamento del filtro a frequenze diverse attorno a quella di taglio si ottiene:

```
freq = [0.160, 1.60, 16.0, 160]*2*pi; %Convertita in w [Rad/s]  
M = [0.947, 0.684, 0.1, 0.0138];  
phi = [-5.94, -43.776, -82.64, -89.7] * pi / 180; %[Rad]  
resp = M .* (cos(phi) + 1j * sin(phi));  
sperimental_H_A = frd(resp, freq);
```

## FILTRO DIGITALE:

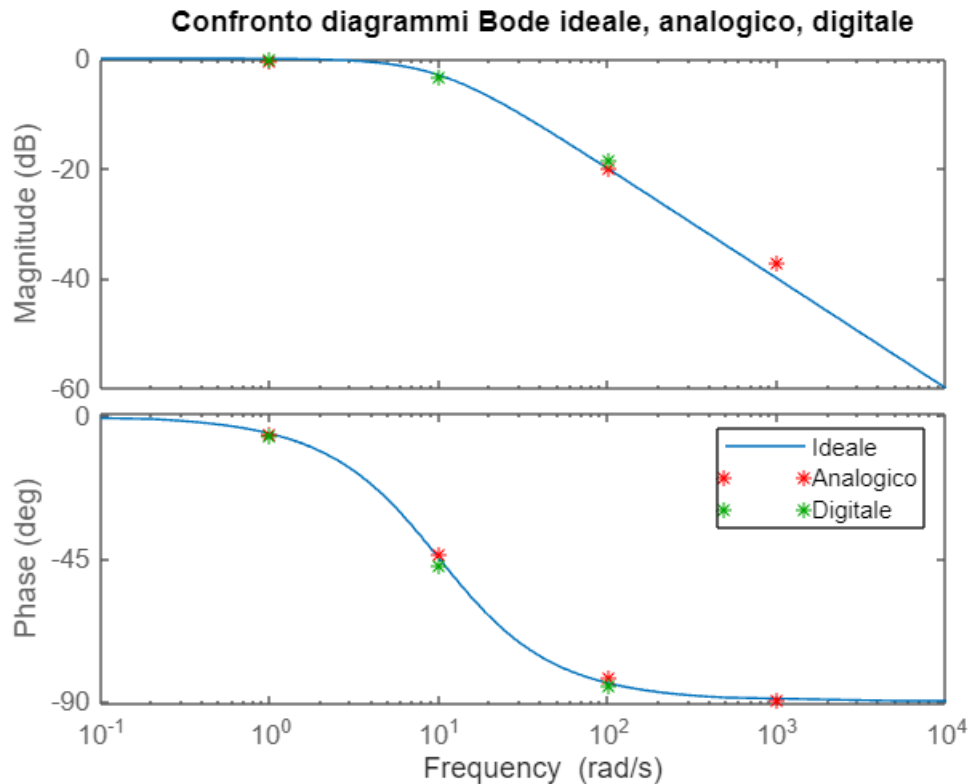
Per realizzare e studiare il comportamento del filtro digitale saranno necessari ADC (Convertitori Analogici e Digitali, già implementati nella struttura di Arduino) e i DAC (Convertitori Digitali-Analogici). La scheda Arduino non è dotata di questi convertitori, ma è fornito solamente da alcune uscite PWM. Per realizzare il DAC, si va a filtrare il segnale in uscita dal PWM attraverso un filtro analogico passa-basso.

```
freq_DIG=[0.16 1.6 16]*2*pi;  
M_DIG = [0.98 0.68 0.12];  
phi_DIG=[-6.512 -47.354 -85.2]*pi/180;  
resp_DIG=M_DIG.*(cos(phi_DIG)+j*sin(phi_DIG));  
sperimental_H_D=frd(resp_DIG,freq_DIG);  
bode(G);
```

```
hold;
```

Current plot held

```
bode(sperimental_H_A,'r*')  
bode(sperimental_H_D,'g*')  
legend('Ideale','Analogico','Digitale');  
title('Confronto diagrammi Bode ideale, analogico, digitale');
```



## CONCLUSIONE

In conclusione, l'analisi comparativa tra un filtro digitale e un filtro analogico alla stessa frequenza di taglio (1.6 Hz) ha rivelato che, nonostante le similitudini nelle loro capacità di filtraggio, le differenze risiedono principalmente nella loro implementazione e nelle caratteristiche di progettazione.

Mentre il filtro digitale offre una maggiore flessibilità e programmabilità grazie alla sua natura basata su software, richiede una maggiore attenzione ai dettagli tecnici come la frequenza di campionamento e la modulazione a larghezza di impulso (PWM).

D'altro canto, il filtro analogico presenta vantaggi intrinseci legati alla sua semplicità e alla sua gestione più diretta del segnale, senza la necessità di conversioni digitali o parametri aggiuntivi.

Pertanto, la scelta tra un filtro digitale e uno analogico dipenderà dalle esigenze specifiche dell'applicazione, valutando attentamente i trade-off tra complessità, precisione e facilità d'uso.

