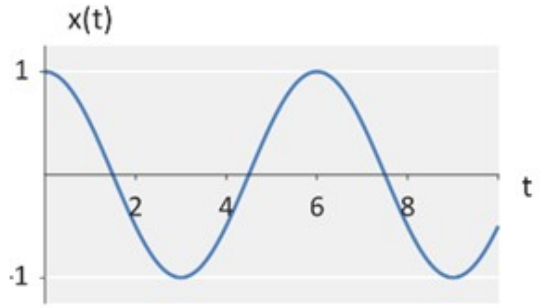
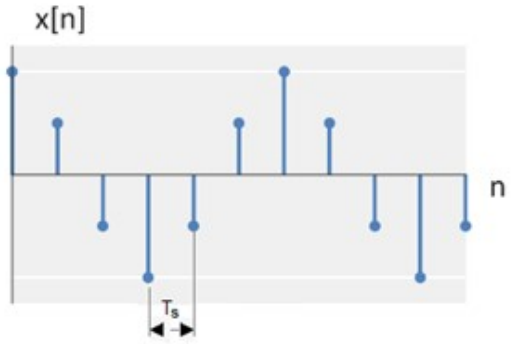
SISTEMI – SIMULAZIONI

Prof. Fischetti P,



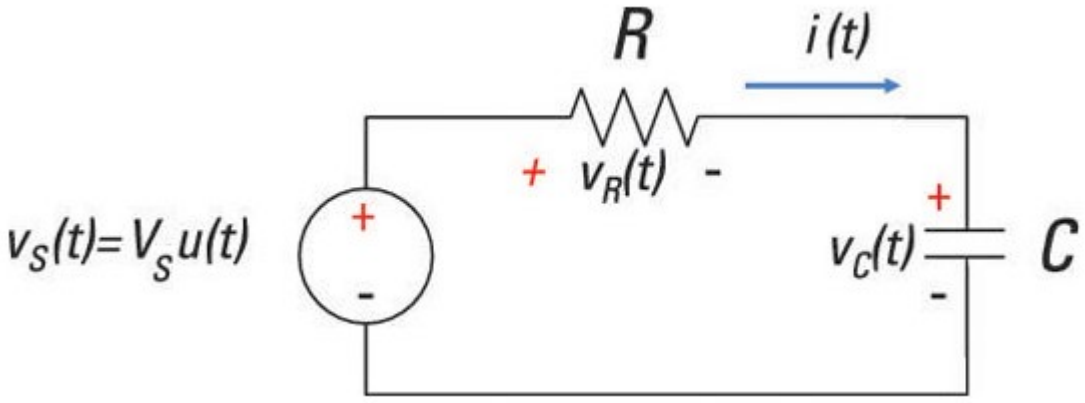
Continuous-time signal
or Analogic Signal



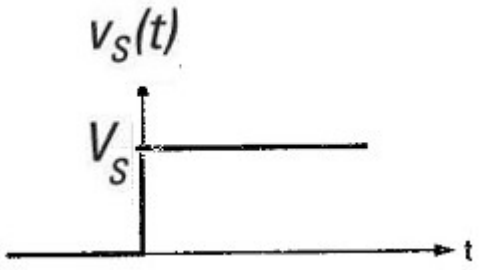
Discrete-time signal
or Digital Signal

Ts: Sampling time (=1/fs Sampling frequency)

Esempio circuito RC Serie con ingresso step



Dove $v_s(t)$ sta per source voltage o tensione di ingresso.



Equazione Differenziale (ODE);

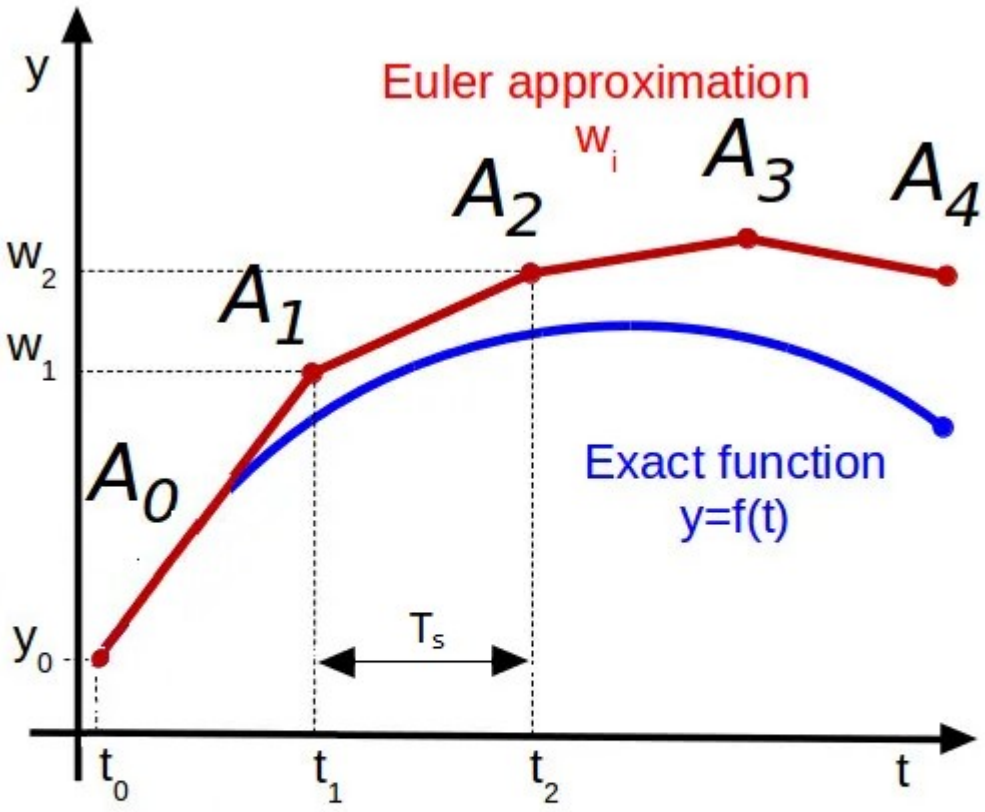
$$v_s(t) = R \cdot C \cdot \frac{dv_c(t)}{dt} + v_c(t)$$

Soluzione dell'equazione differenziale all'ingresso a gradino:

$$v_c(t) = V_s \cdot \left(1 - e^{\frac{-t}{RC}}\right) + V_0 \cdot e^{\frac{-t}{RC}}$$

Dove V_s e' il valore massimo del gradino in ingresso e V_0 e' la tensione iniziale sul condensatore.

Simulazione dell'equazione differenziale. Esistono vari metodi utilizziamo il piu' semplice:



$$t_1 = t_0 + T_s$$

$$t_2 = t_1 + T_s = t_0 + 2 \cdot T_s$$

$$t_3 = t_2 + T_s = t_0 + 2 \cdot T_s + T_s = t_0 + 3 \cdot T_s$$

.....

Backward Euler Method:

$$\frac{dy(t)}{dt} \cong \frac{y(nT_s) - y\big((n-1)T_s\big)}{T_s}$$

Che applicata al circuito RC serie porta alla equazione alle differenze:

$$v_s(nT_s) = R \cdot C \cdot \frac{v_c(nT_s) - v_c\big((n-1)T_s\big)}{T_s} + v_c(nT_s)$$

Da cui si ottiene:

$$v_c(nT_s) = \frac{v_s(nT_s) \cdot T_s + RC \cdot v_c\big((n-1)T_s\big)}{T_s + RC}$$

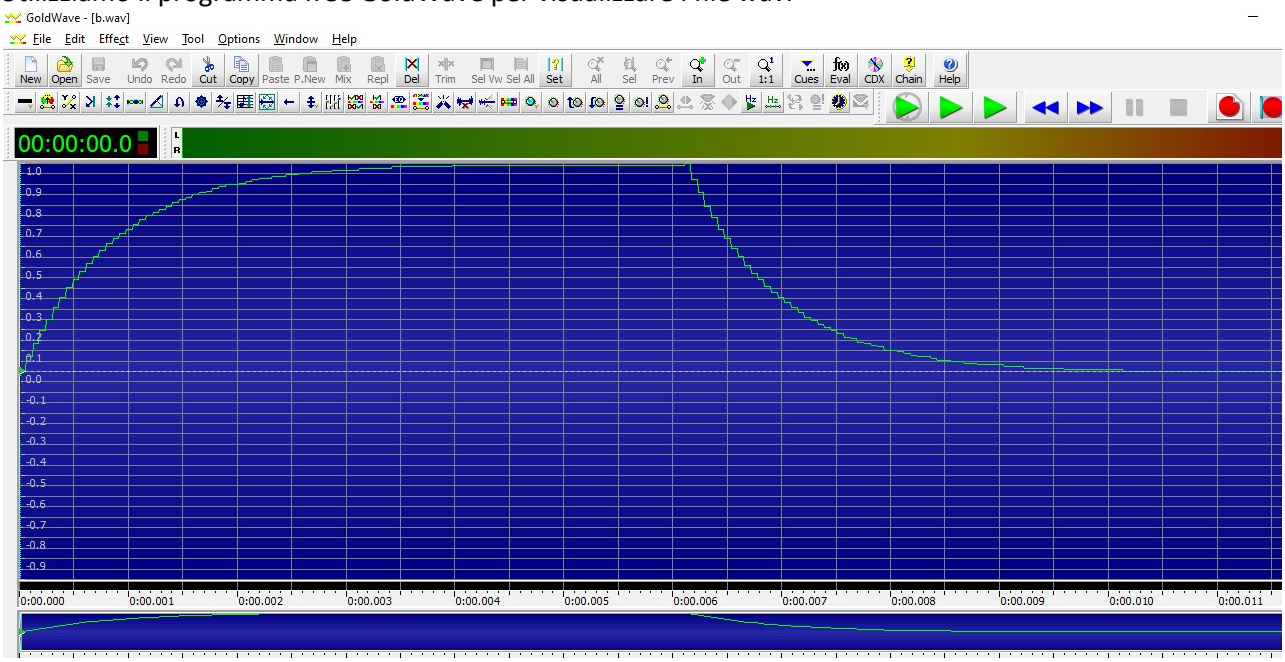
step.c	RC.c

#include <stdio.h>
 	
int main(){
int i;
for(i=0; i < 100; i++)
printf("4.0\n");
for(i=0; i < 100; i++)
printf(".0\n");
return 0;
}
	#include <stdio.h>
	
	int main(){
	float Vin=0, Vout=0, R=8000, C=100E-9,fs=16384.,Ts=1./fs;
	fprintf(stderr,"R=%.0f, C=%G, tau=%f, 1/tau=%f, fs=%f, Ts=%E\n",R,C,R*C,1./(R*C),fs,Ts);
	while(scanf("%f",&Vin)>0)
	{
	Vout=((Ts*Vin)+(R*C*Vout))/(Ts+R*C);
	printf("%f\n",Vout);
	}
	return 0;
	}

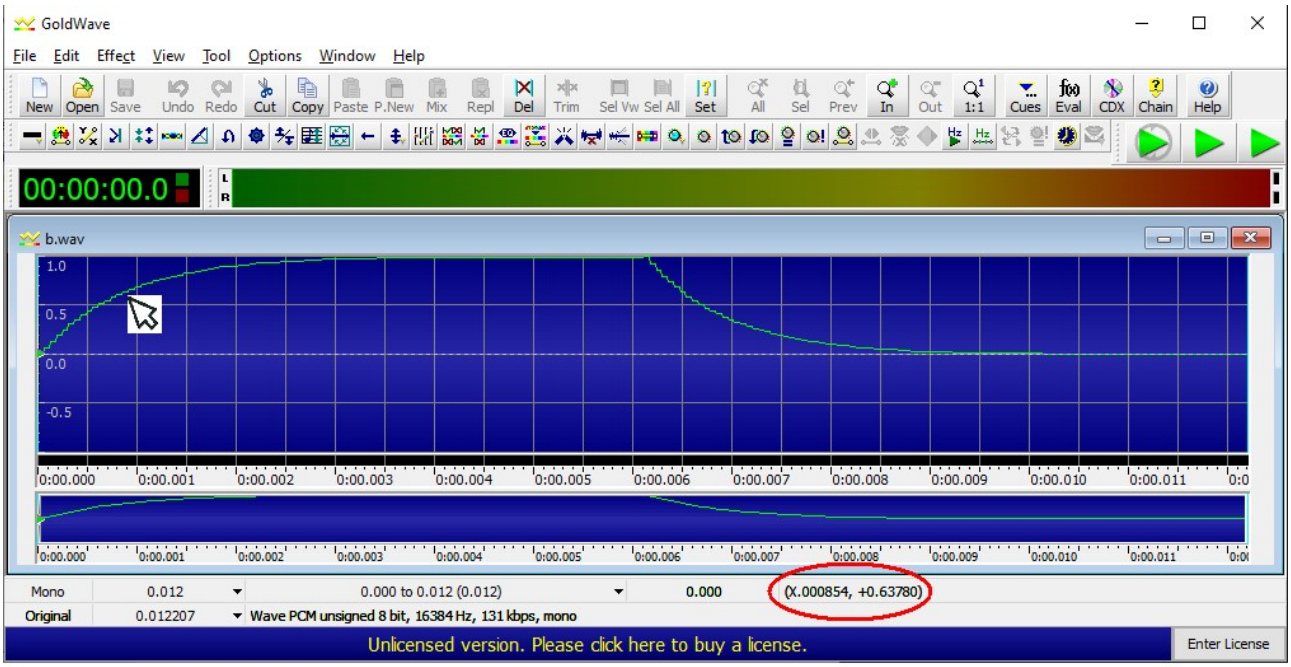
C:\ > step| rc > a.txt & normalize < a.txt | atowav> b.wav & goldwave b.wav
R=8000, C=1E-07, tau=0.000800, 1/tau=1249.999941, fs=16384.000000, Ts=6.103516E-05

Quindi la costante di tempo τ vale:0.008s

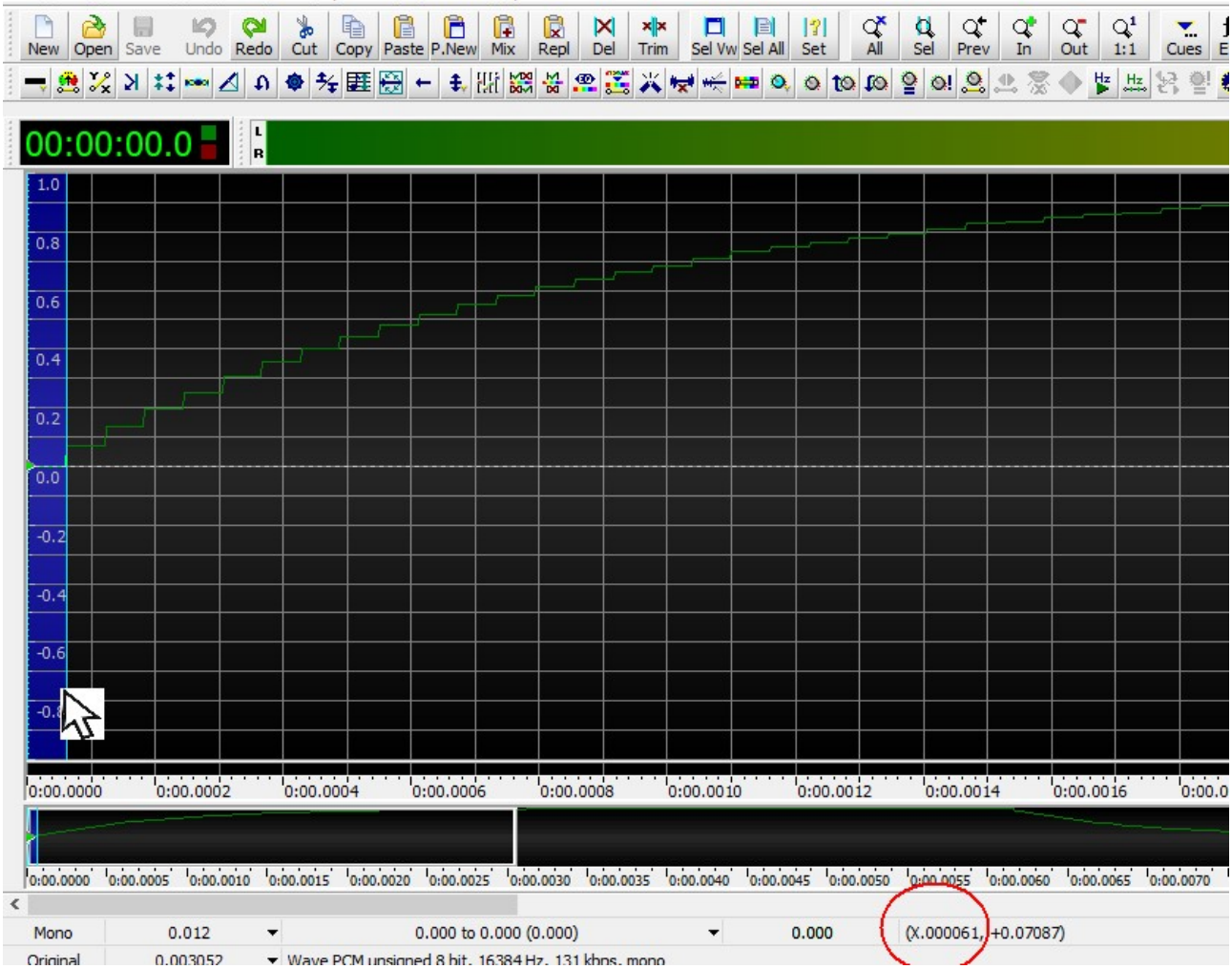
Utilizziamo il programma free GoldWave per visualizzare i file wav.



Verifichiamo che a τ s abbiamo il 63% del valore max. :



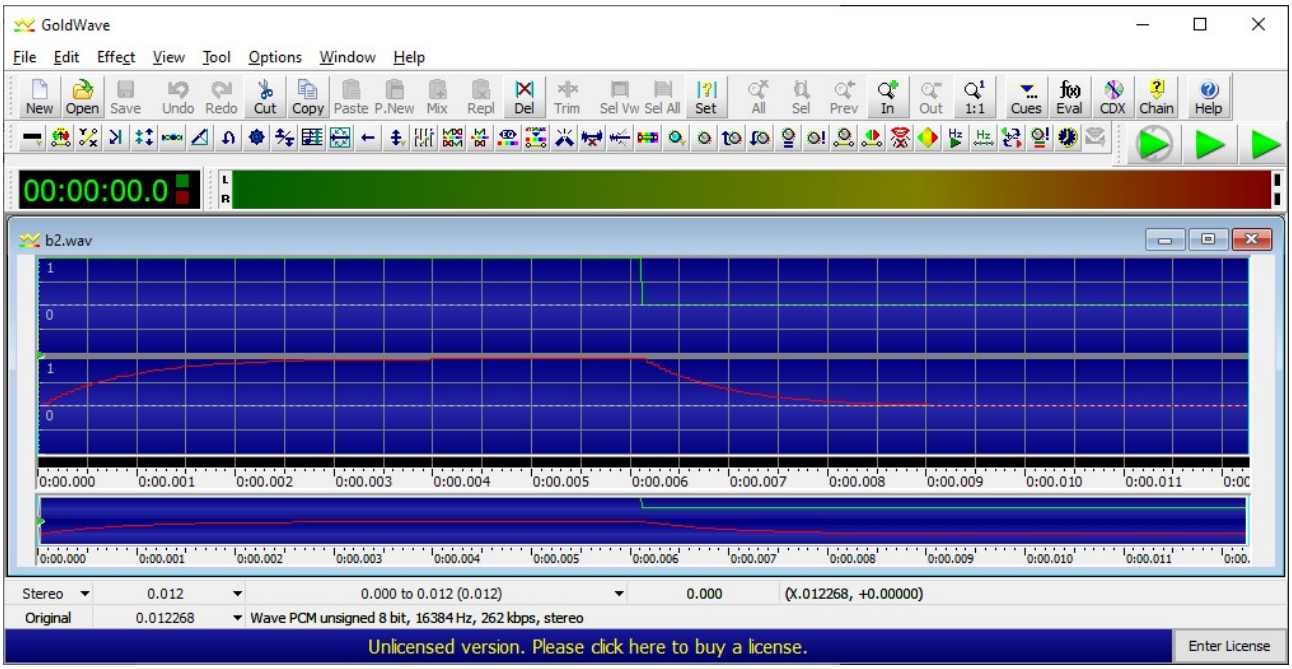
Zoomando si puo' verificare il valore dell'intervallo di campionamento:



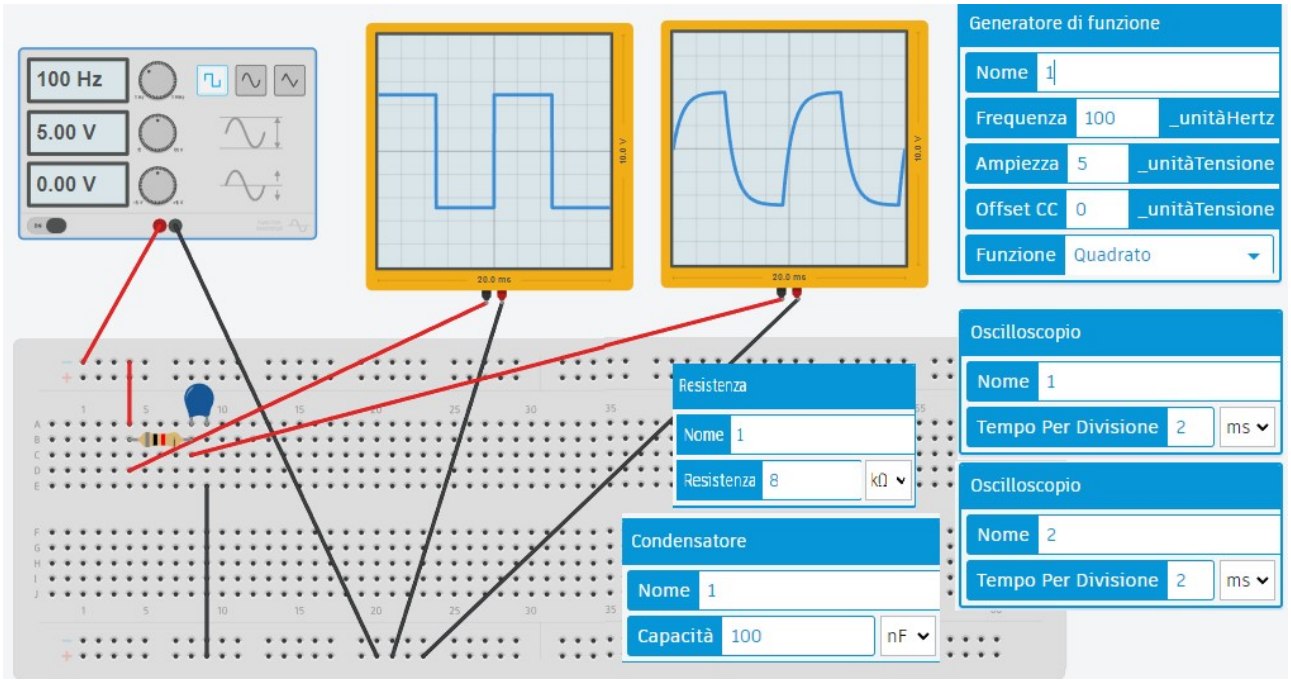
Possiamo sfruttare la capacita' dei file wav sterereo per visualizzare il segnale di ingresso e i lsegnale di uscita (i comandi seguenti possono essere messi in un file .bat per comodita'):

C:\>step > a.txt & rc < a.txt > b.txt & normalize < b.txt > bn.txt

C:\>normalize < a.txt | atowav -c2 -2bn.txt > b2.wav
C:\> goldwave b2.wav



Simulazione con Tonkerkit:



sin.c – Generatore di segnale sinusoidale digitale

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
 	
int main(int argc, char *argv[]) {
float f = 5000; //frequenza della sinusoide da generare
float T=1./f; //Periodo della sinusoide da generare
int S=16384; //Frequenza di campionamento

```
        float Ts = 1. / S;  //Tempo di campionamento
        float t = 0; //la variabile tempo
        float y=0;  // variabile di uscita
        float YM = 1; //Val. Max Segnale di uscita

        int i;
        int NT=3;//Num Periodi da generare
        /*  T/Ts e' il numero di campioni in un periodo */

        for (i=0; i < (T/Ts)*NT; i++){
                        y = YM * sin(2 * 3.14 * f * t);
                        printf ("%f\n",y);
                        t += Ts;
        }
}
```