

|  |  |
| --- | --- |
|  | Lab 2: Sampling techniques. |
|  |  |
|  | Akshar Panchani ID- 202101522  CT303 Digital Communication  10/8/23 |

**Lab 2: Sampling techniques and matlab code**

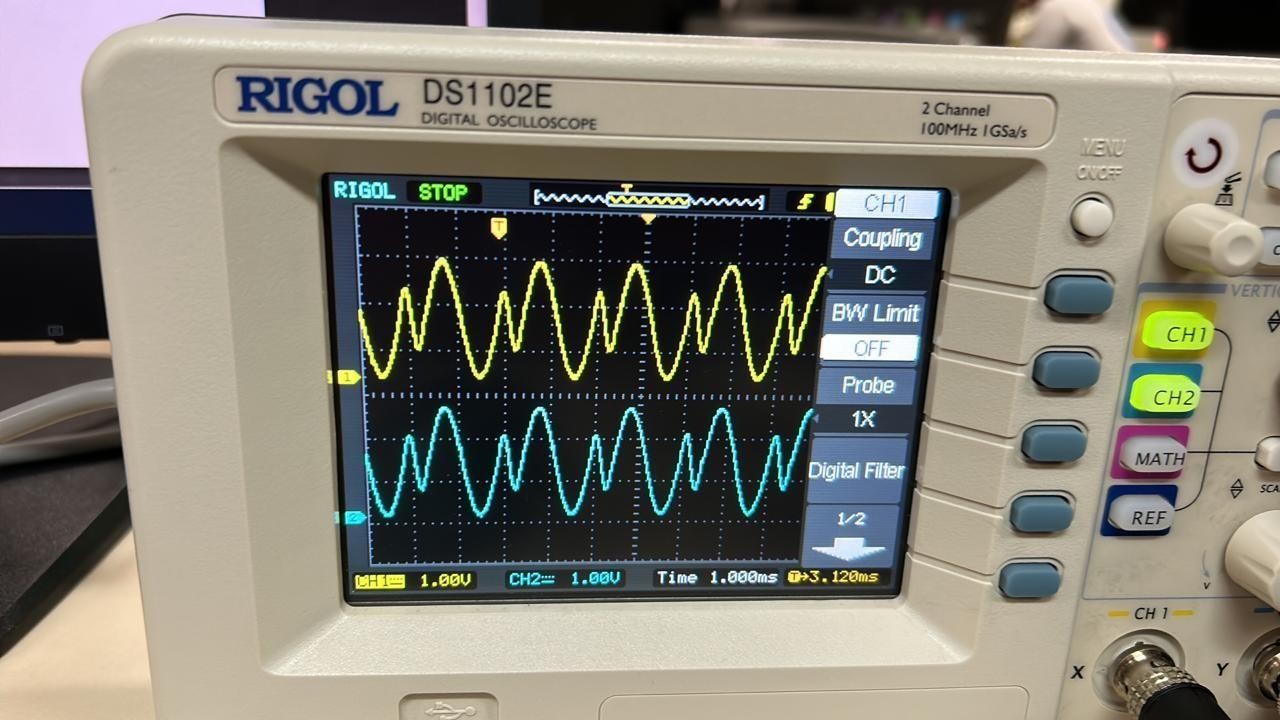
1. Determine the Nyquist sampling rate and the Nyquist sampling interval for the following signals.
   * x(t) = sinc(100πt) + 3 sinc2 (60πt)
   * x(t) = 1 + cos(2000πt) + sin(4000πt).
   * x(t) = sinc(100πt) + 3 sinc^2 (60πt)

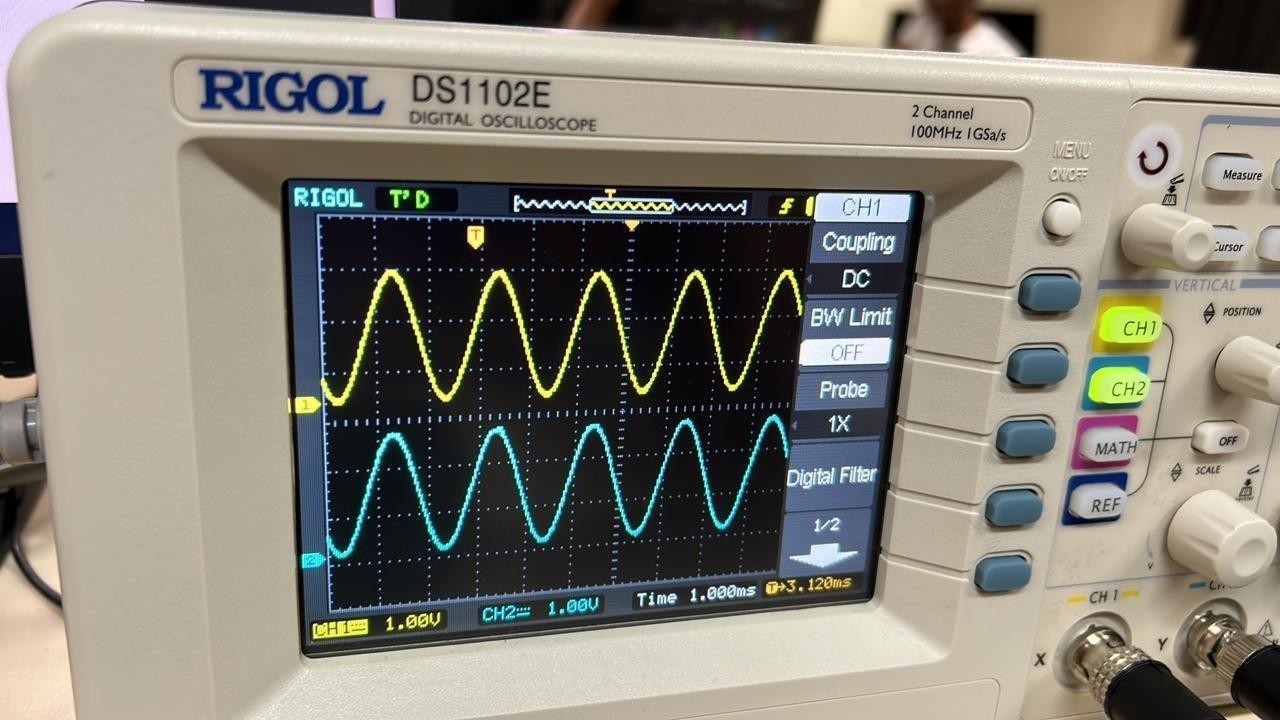
Here, the expression can be expanded as x(t) = sin(100πt)/100πt + 3\*(1-cos(120πt))/2/(60πt)^2, so, the maximum frequency component here is 120π Hz. So, f = 120π/2π = 60Hz which implies Nyquist rate = 2\*60Hz = 120Hz and Nyquist interval = 1/120s = 0.008333333333s.

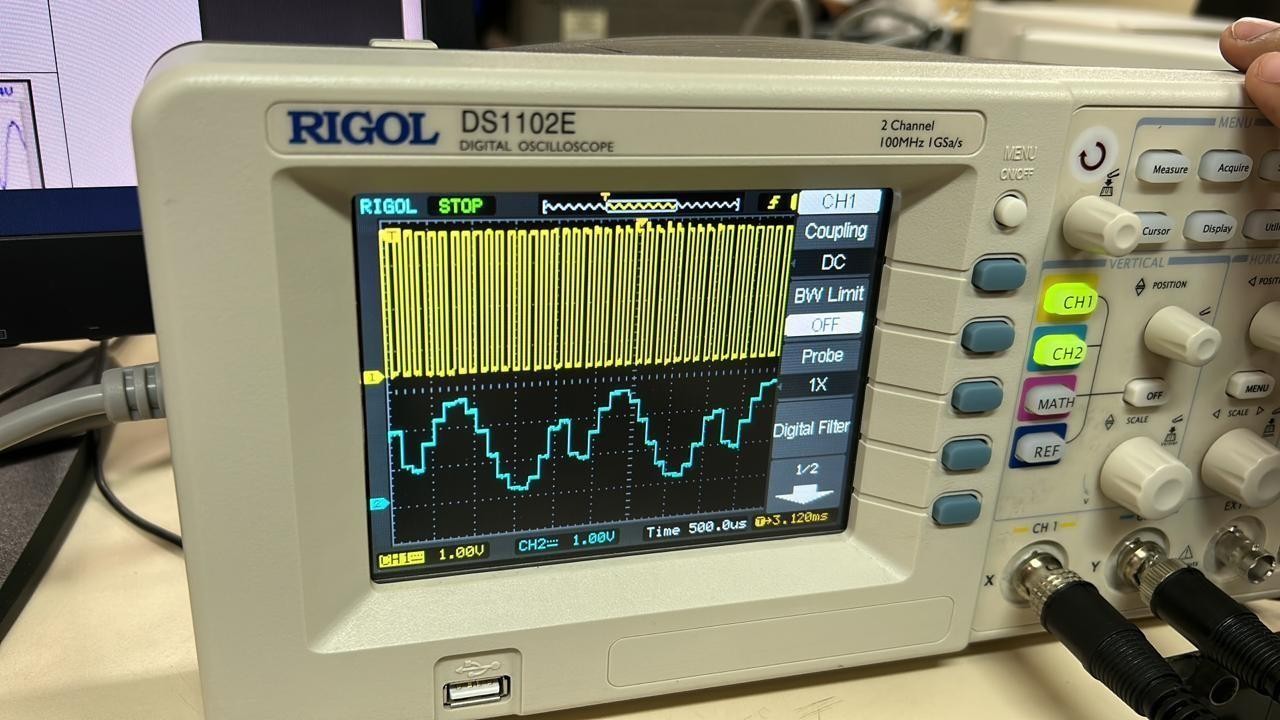
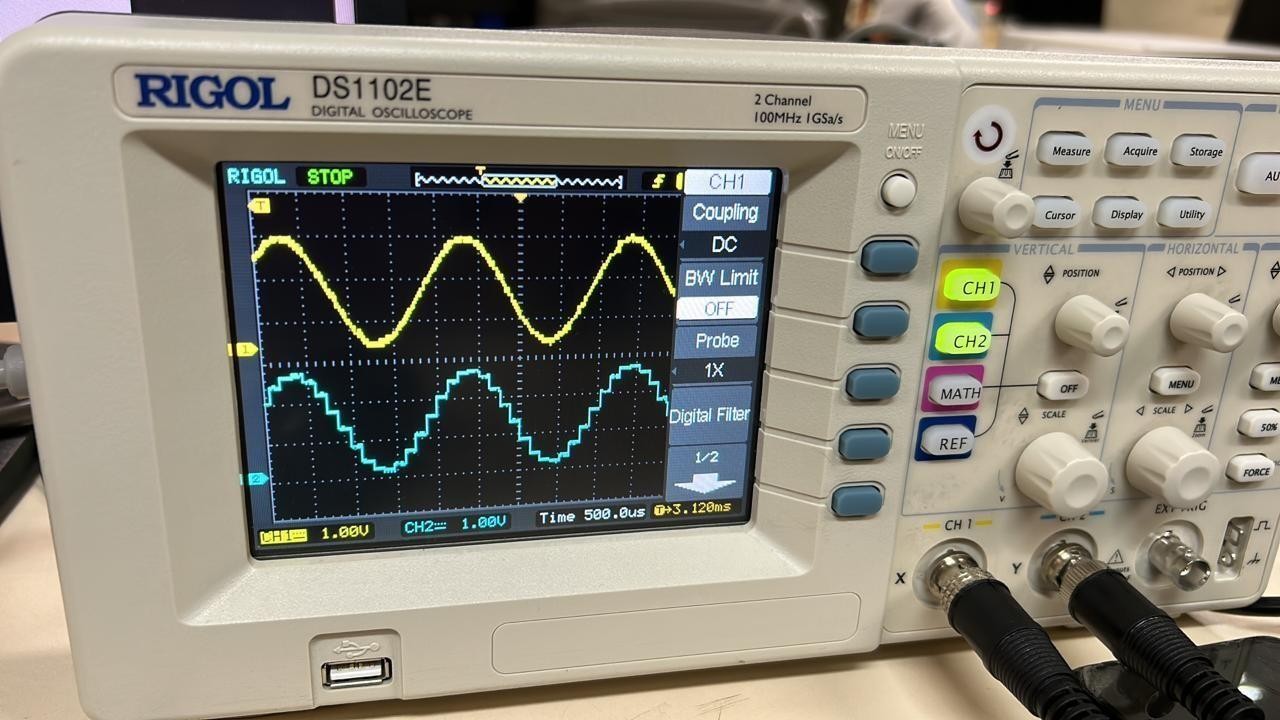
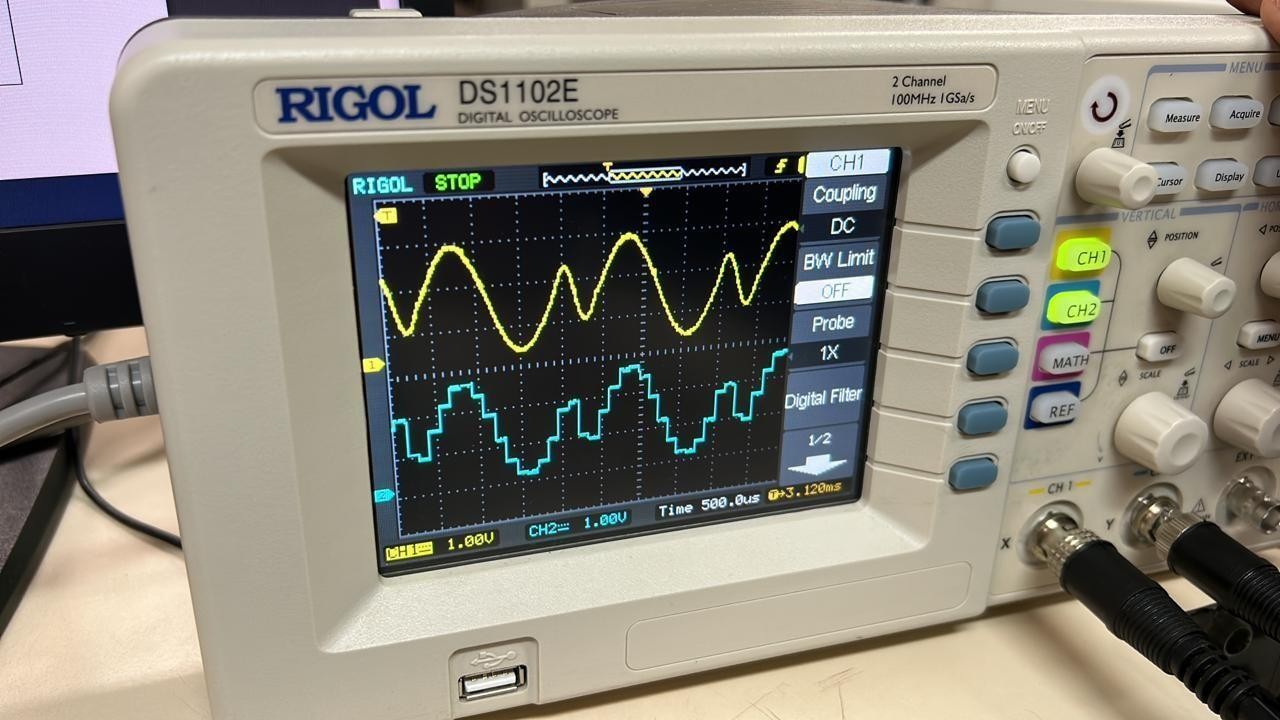
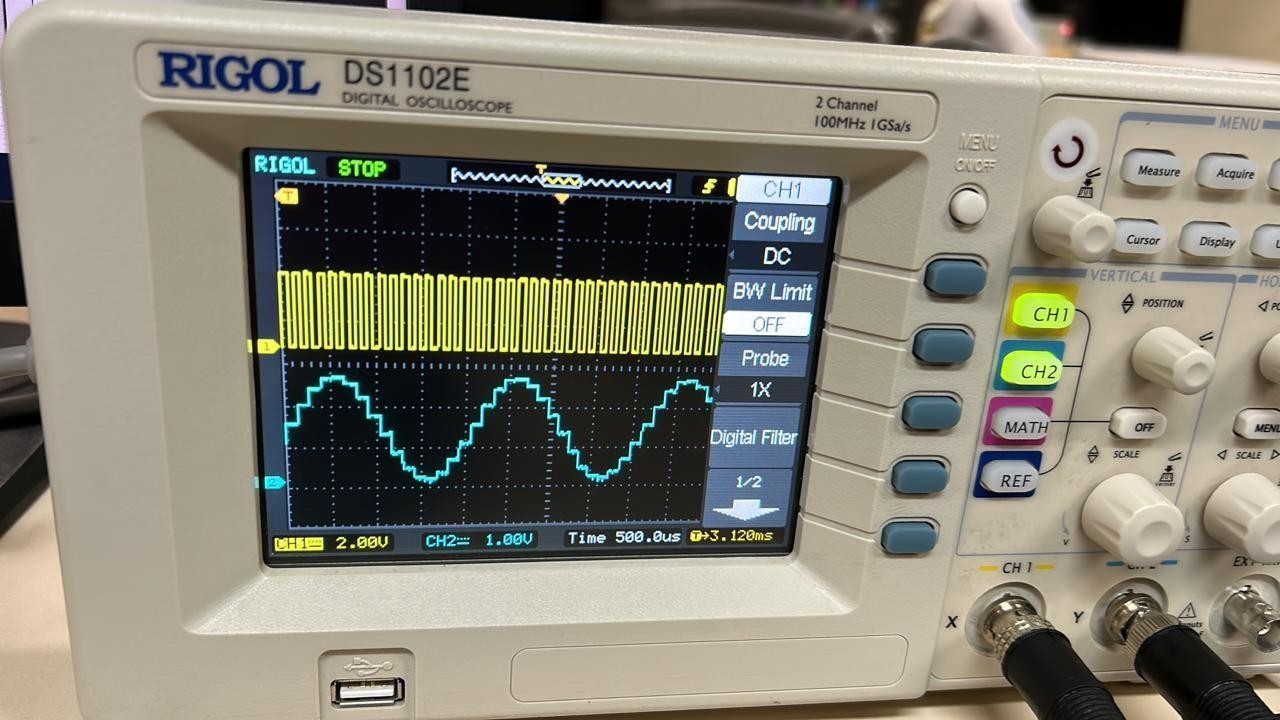
* + x(t) = 1 + cos(2000πt) + sin(4000πt). x(t) = 1 + cos(2000πt) + sin(4000πt).

Here, the expression has maximum frequency component as 4000π Hz. So, f = 4000π/2π Hz = 2000Hz and Nyquist rate = 2\*2000Hz = 4000Hz. The Nyquist interval = 1/4000s = 0.00025s.

# Experiment 1

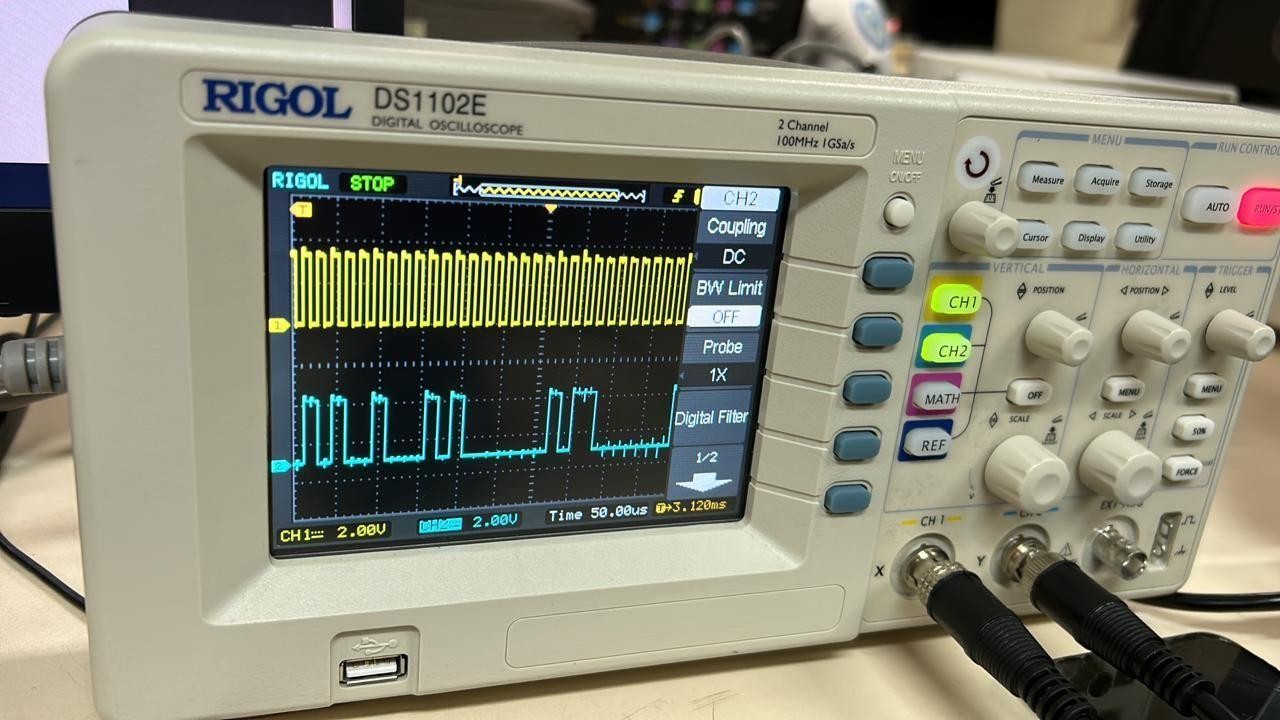
Experiment 1.1 , 1.2 , 1.3 , 1.4 , 1.5 , 1.6 Respectively

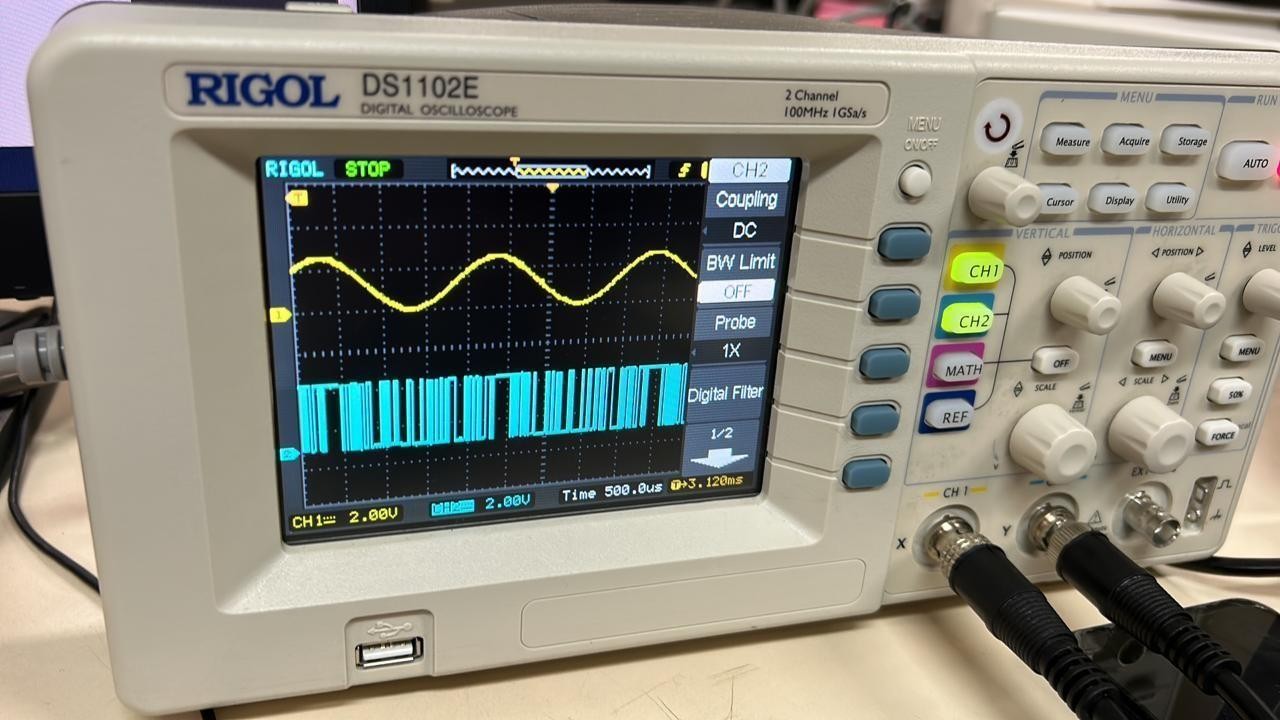


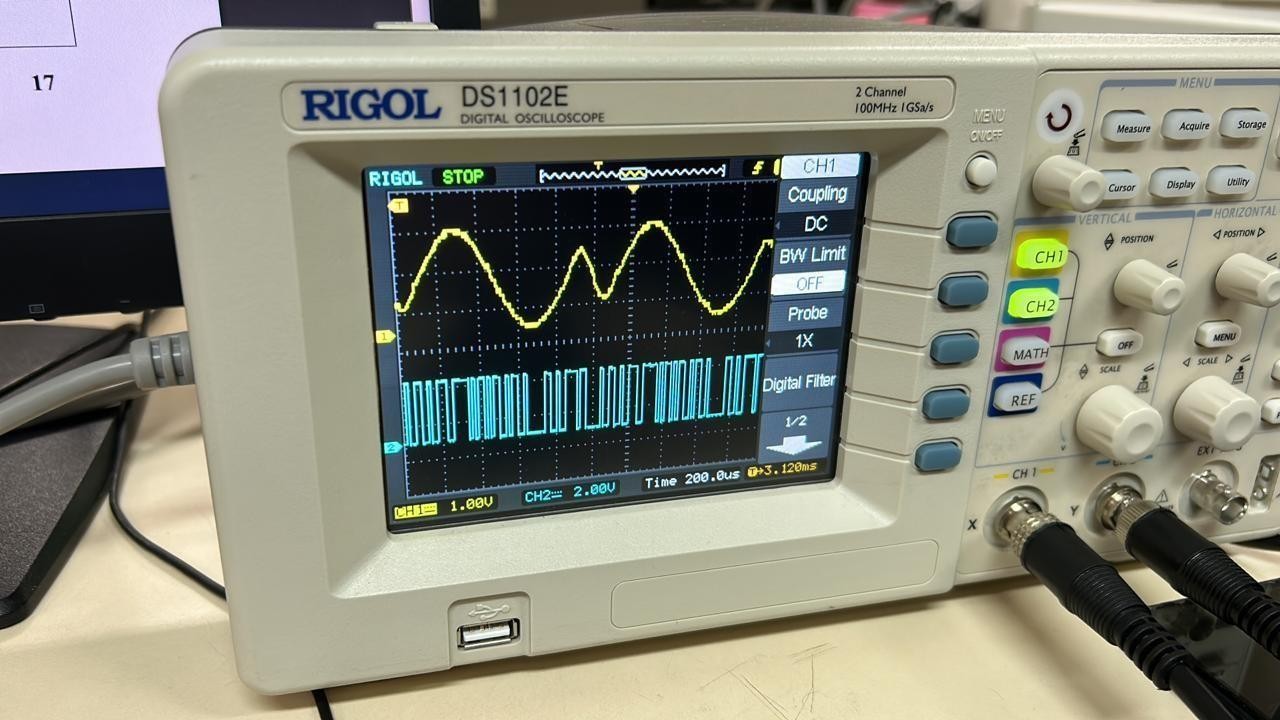
 

# Experiment 2

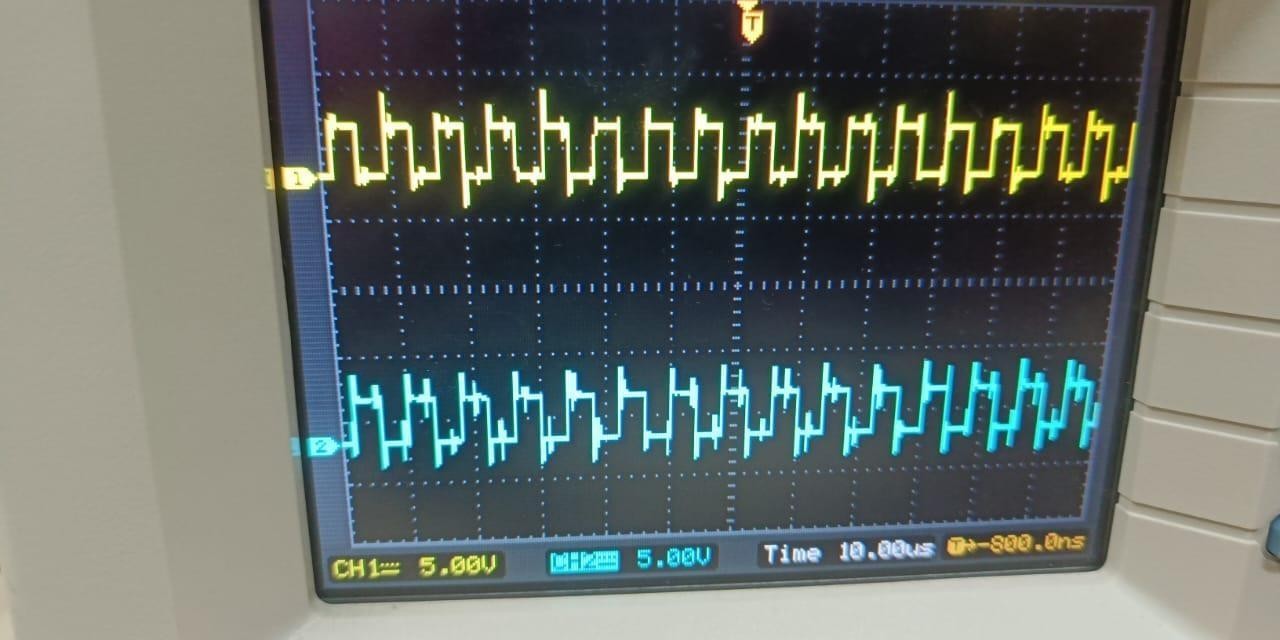
# Experiment 2.1, 2.2, 2.3







# Experiment 3

experiment 3.1, 3.2, 3.3 ,3.4 respectively

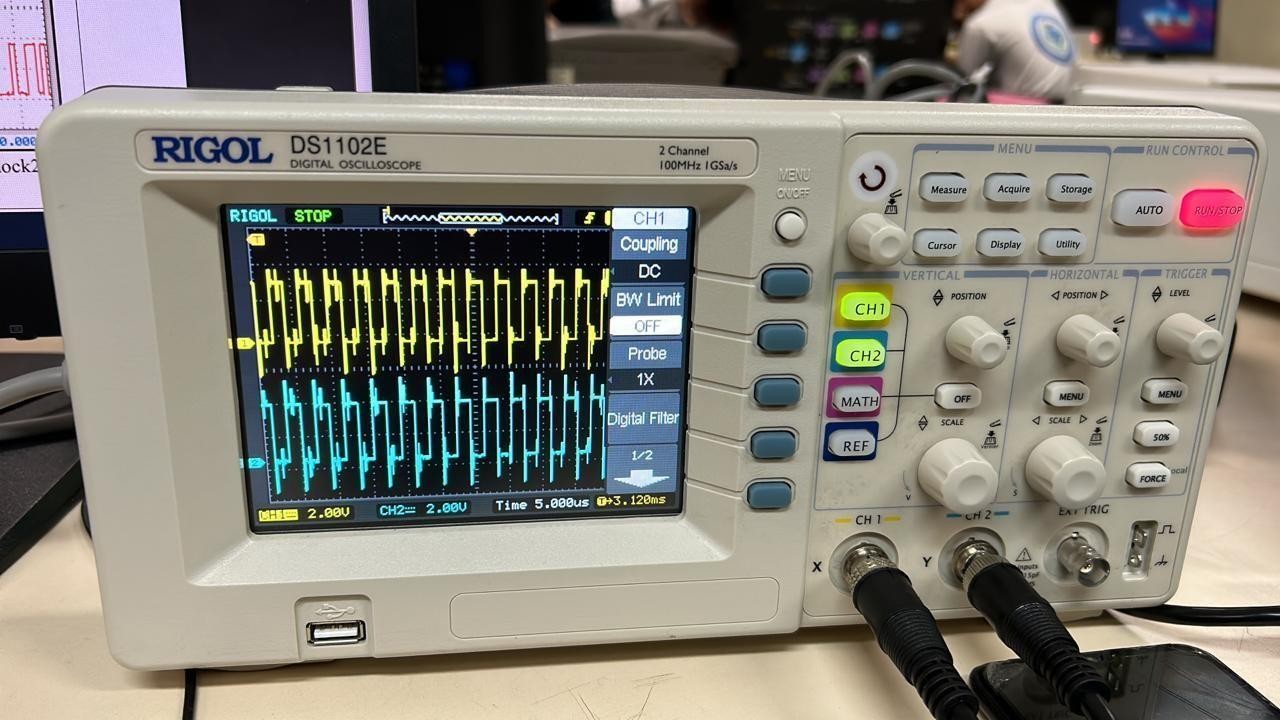


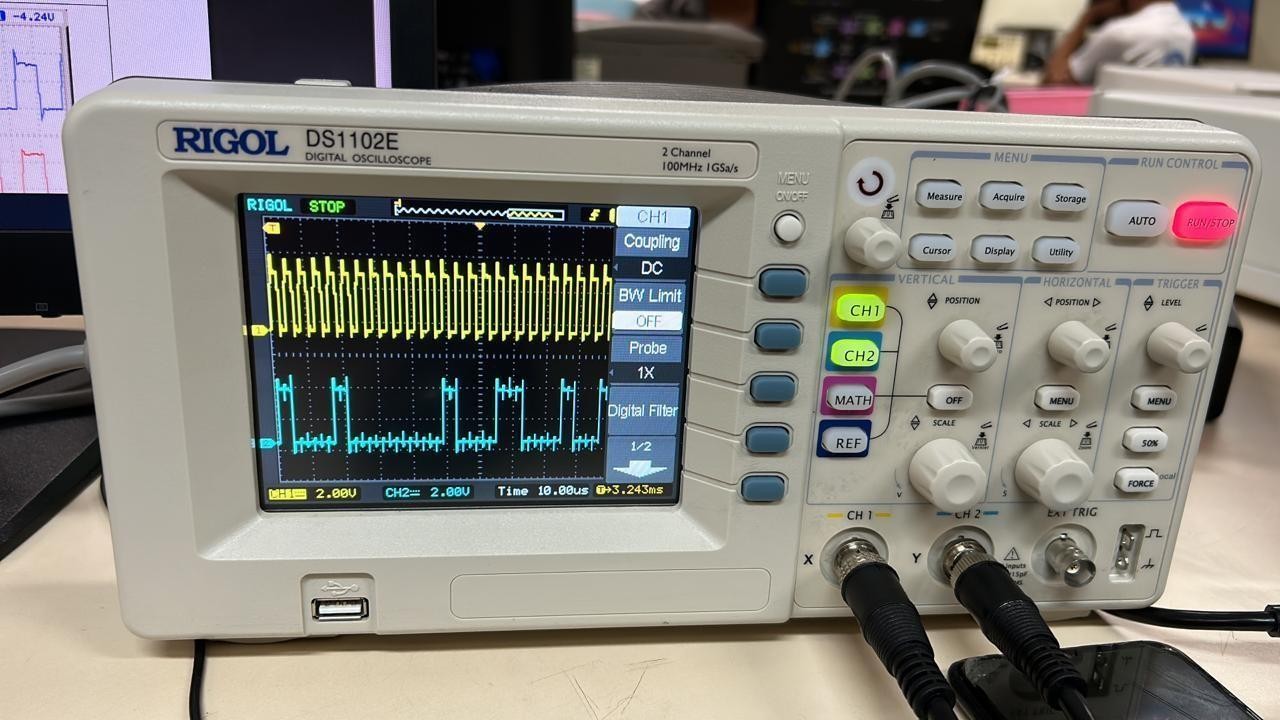


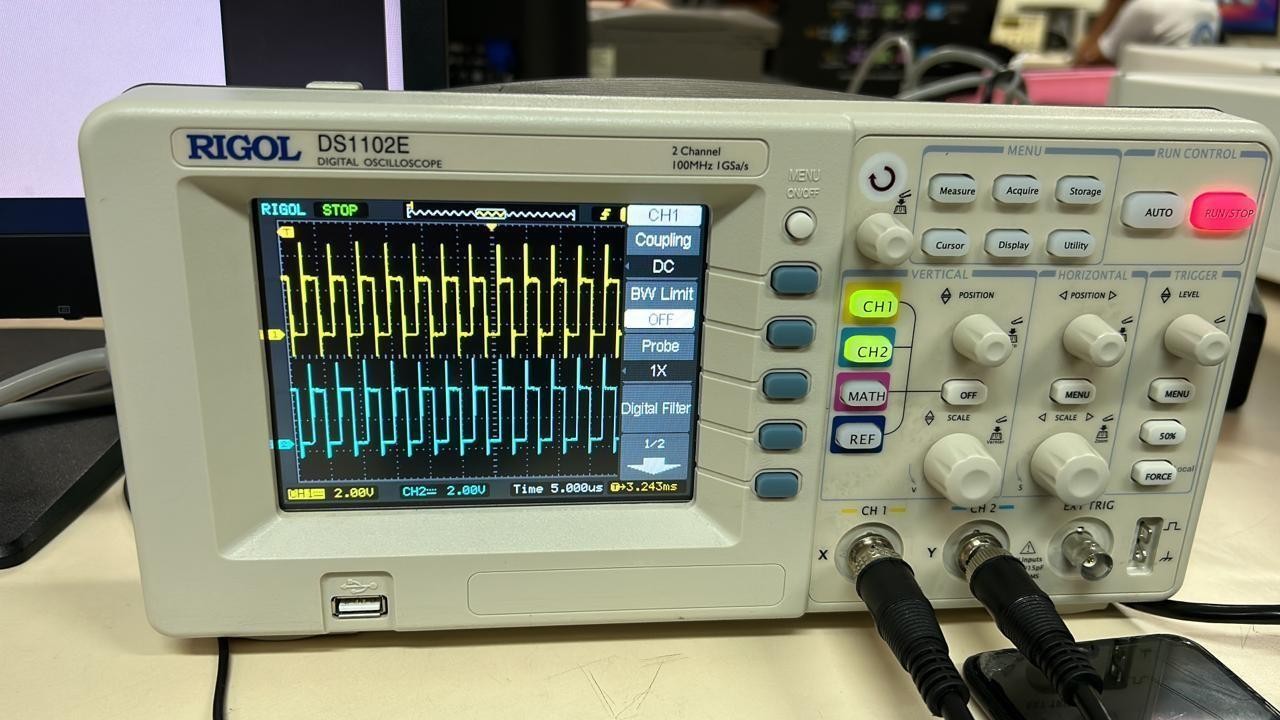


# Experiment 4

experiment 4.1, 4.2, 4.3 respectively

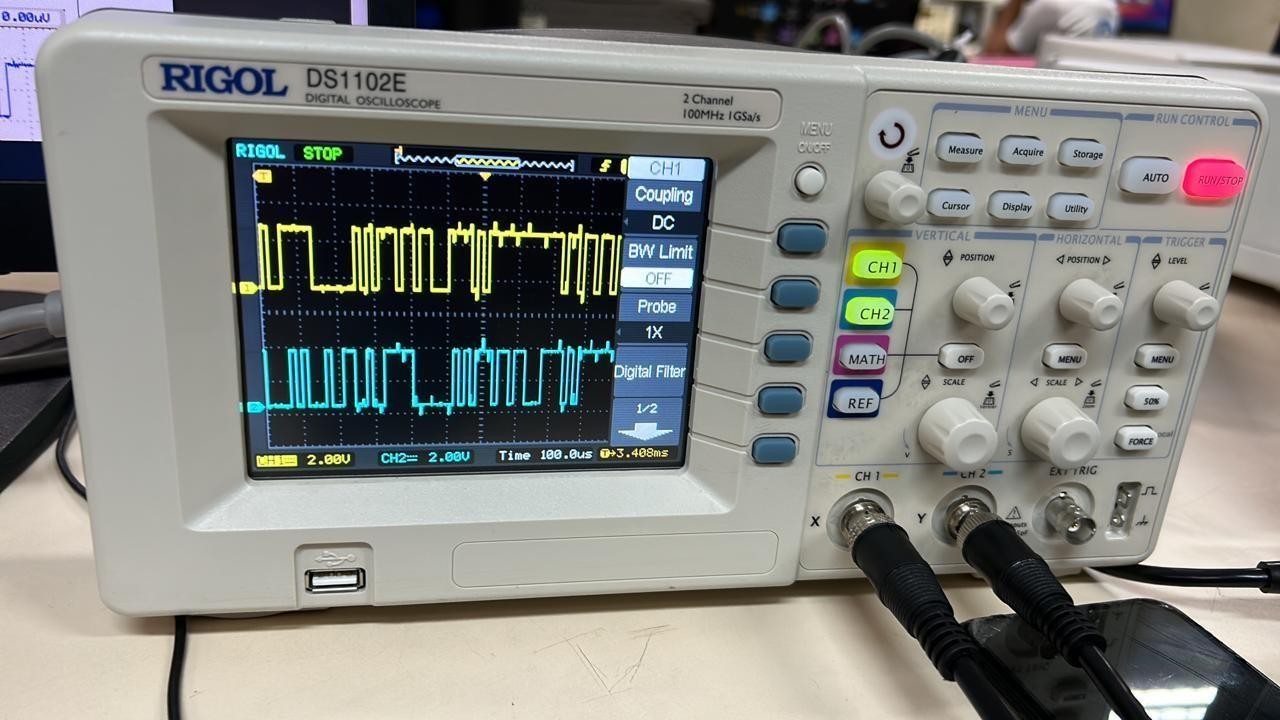
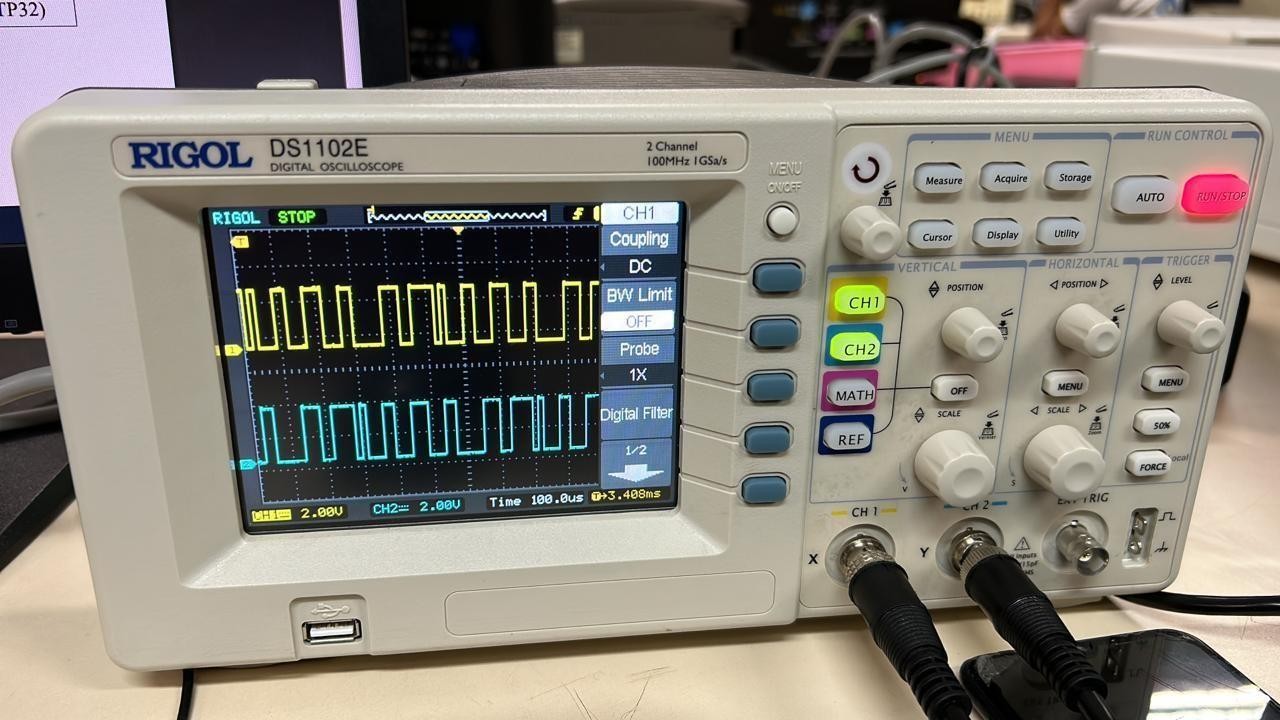




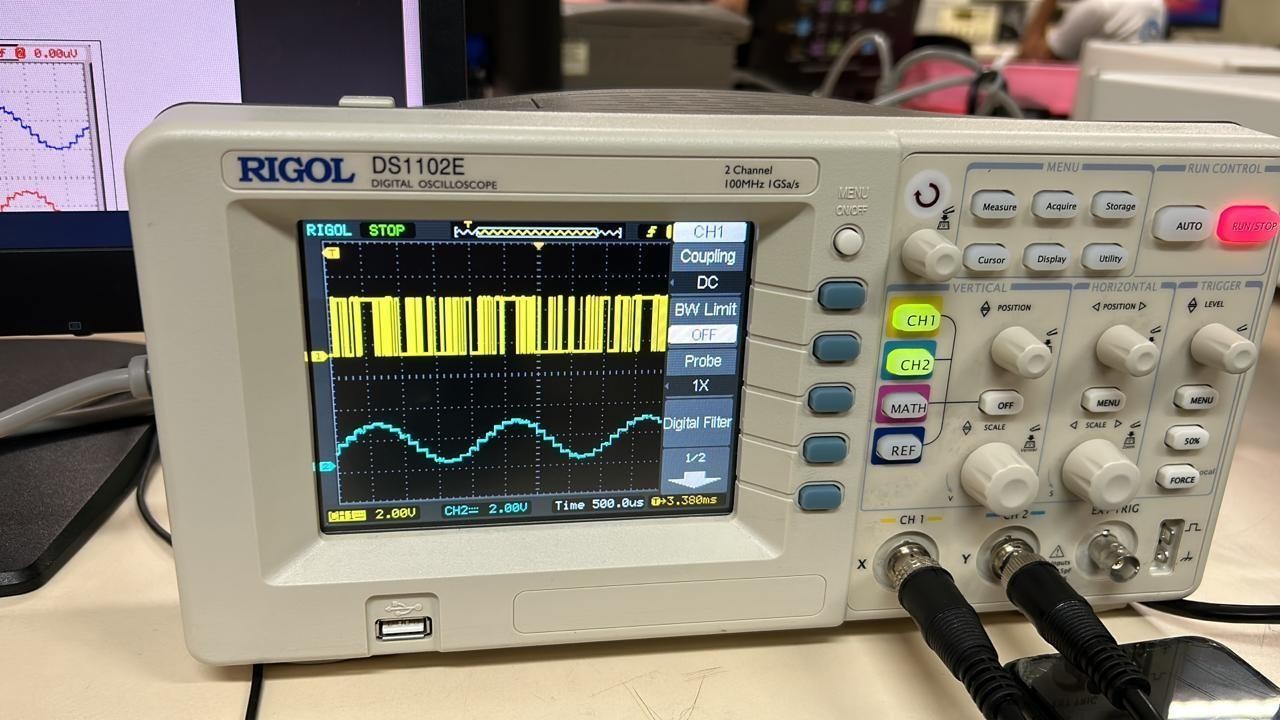
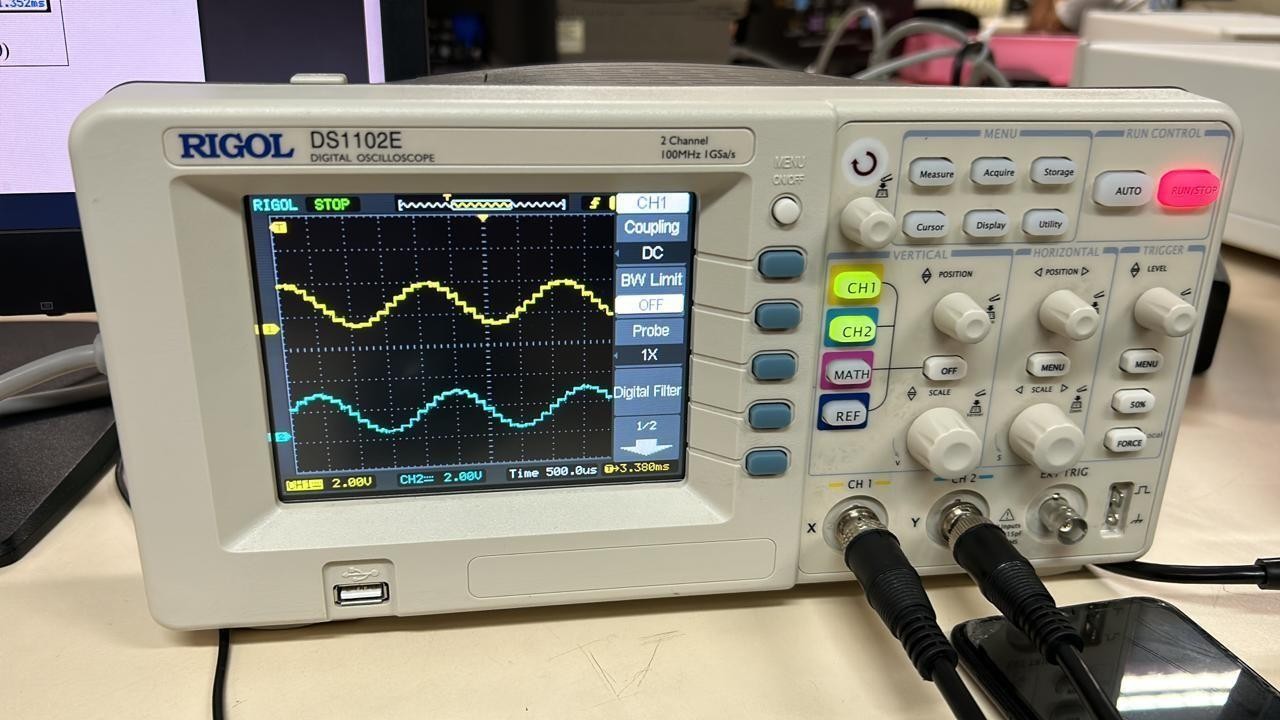


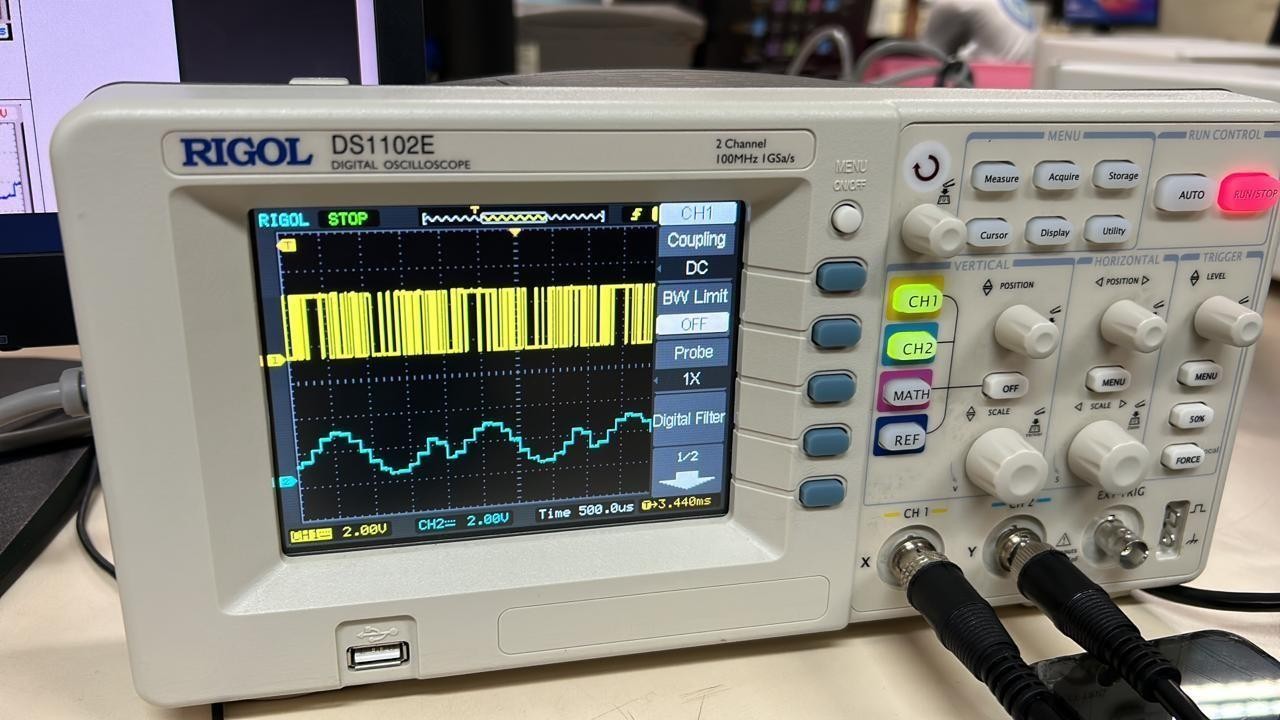
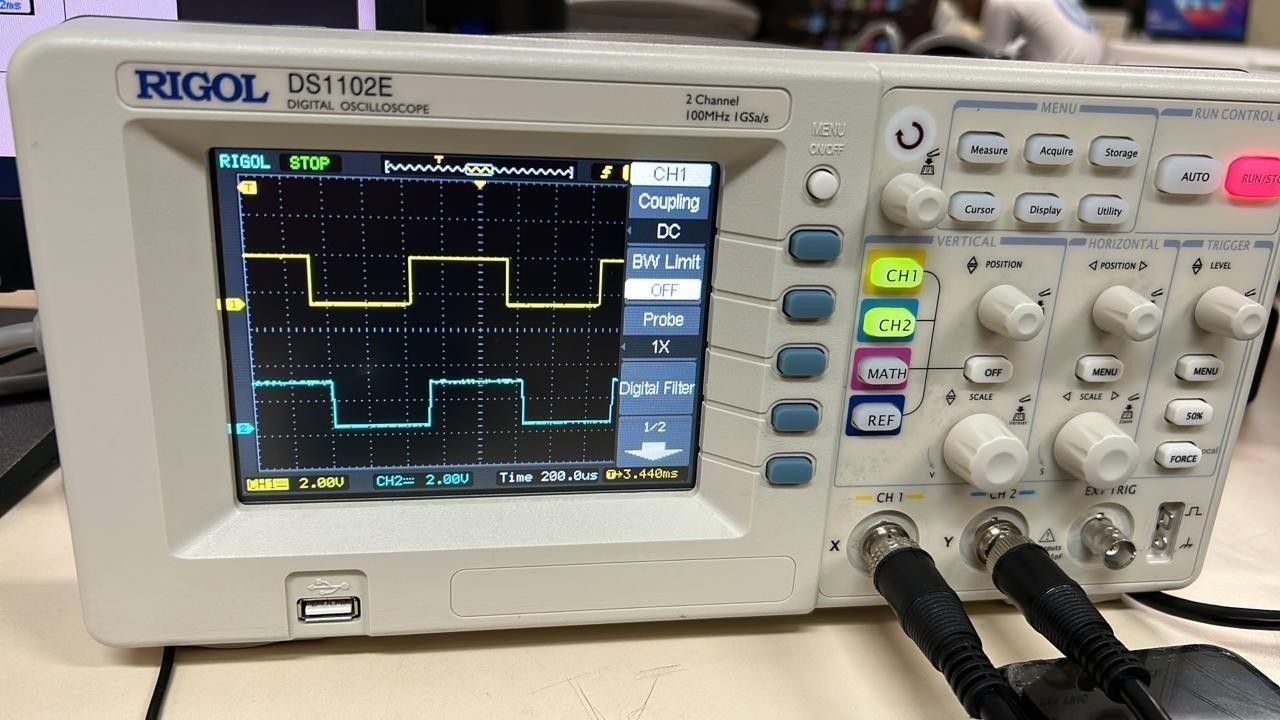
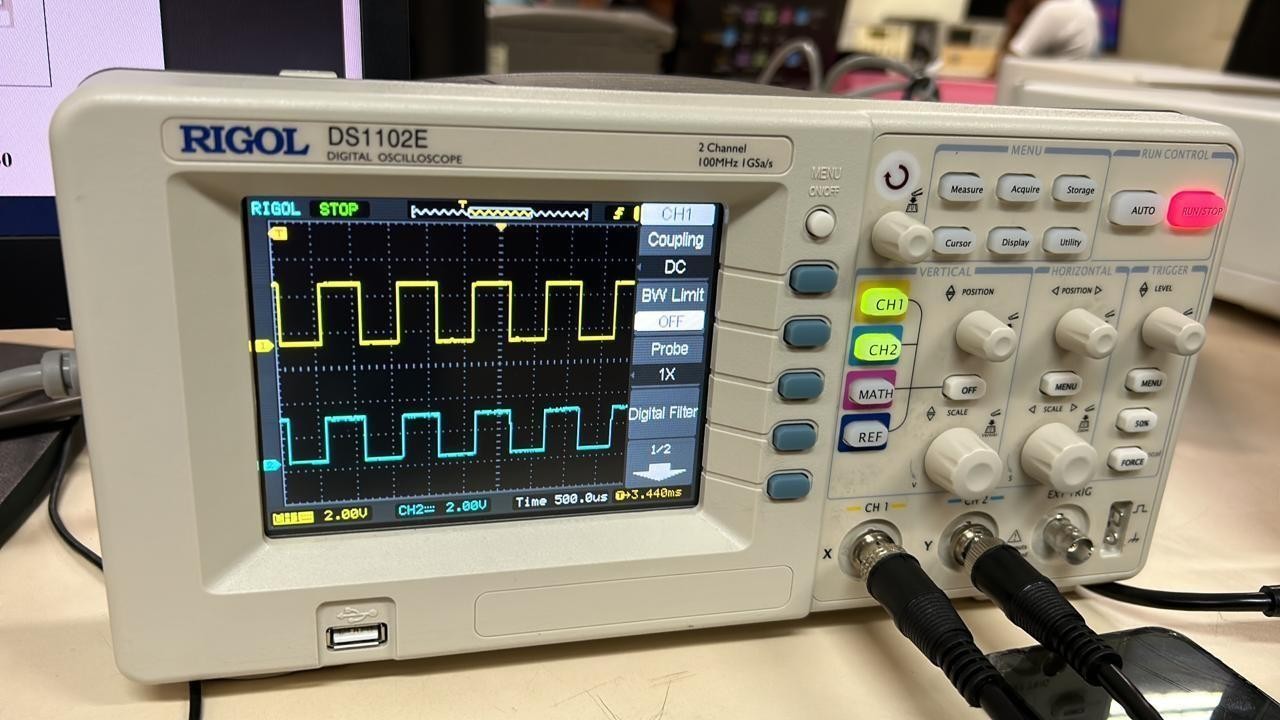
# Experiment 5

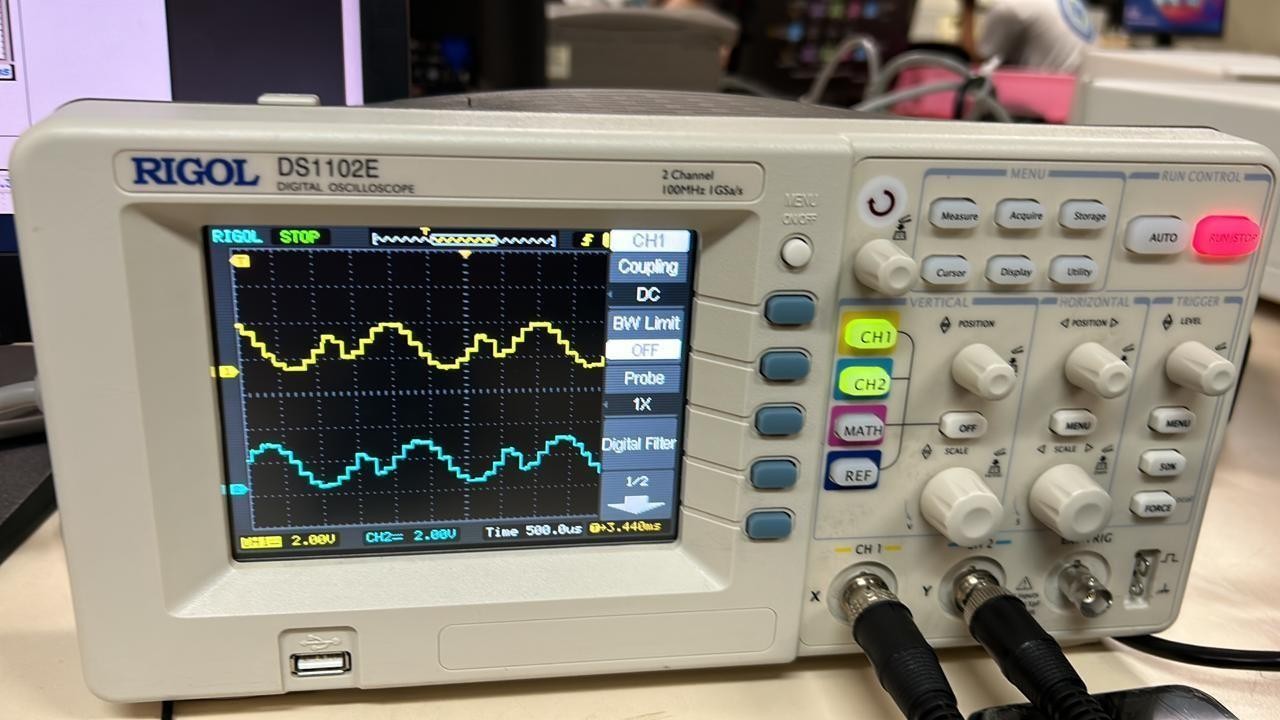
# Experiment 5.1, 5.2



# Experiment 6

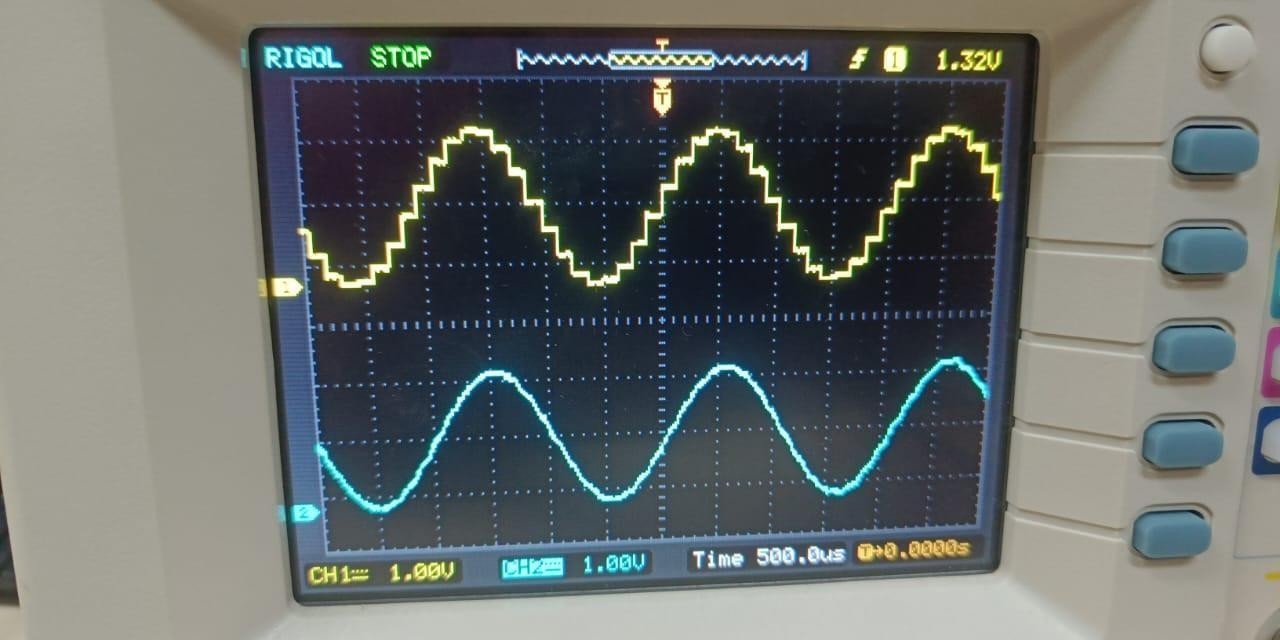
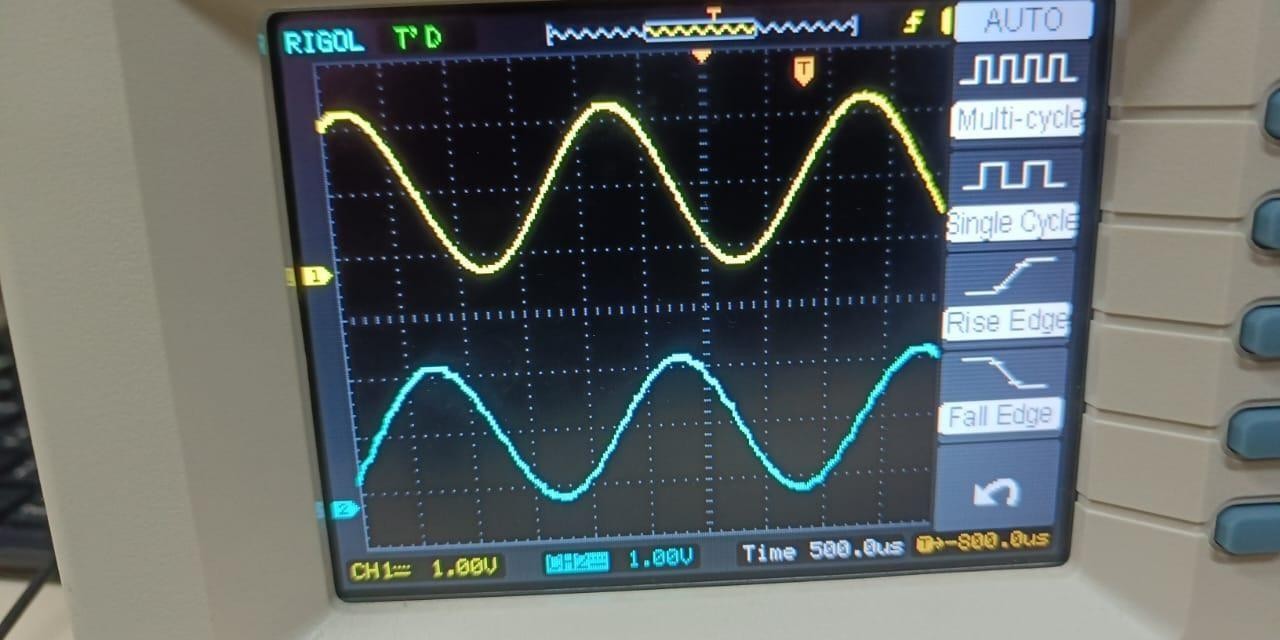
Experiment 6.1, 6.2, 6.3, 6.4 , 6.5, 6.6 respectively

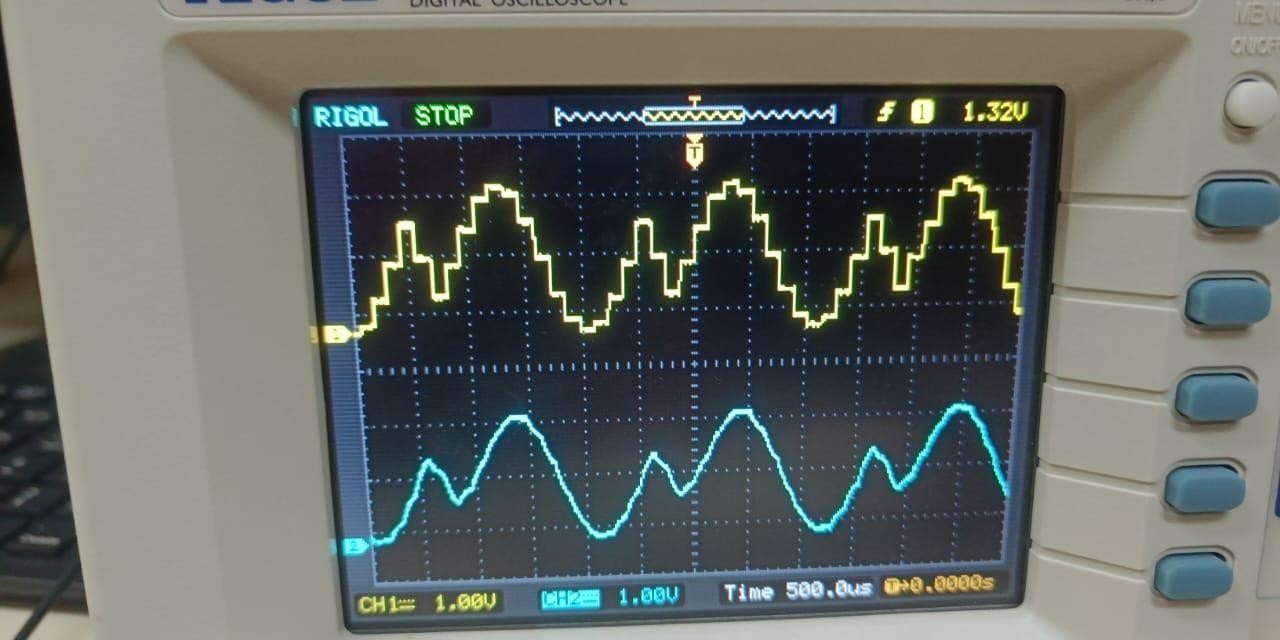
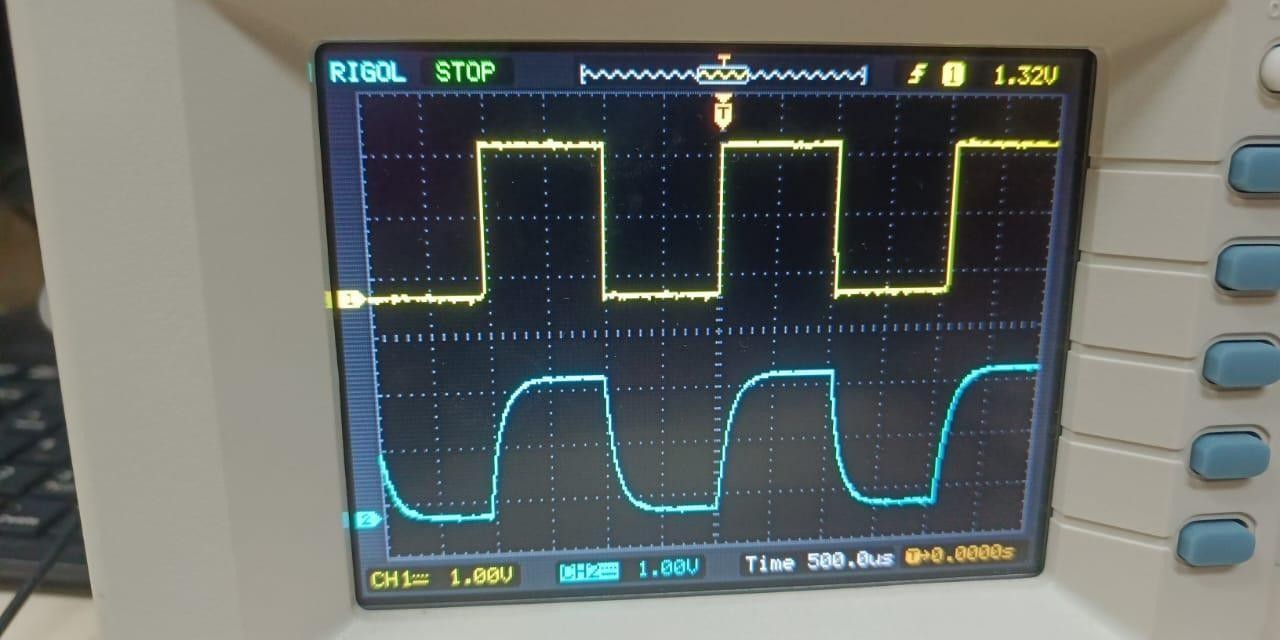
 

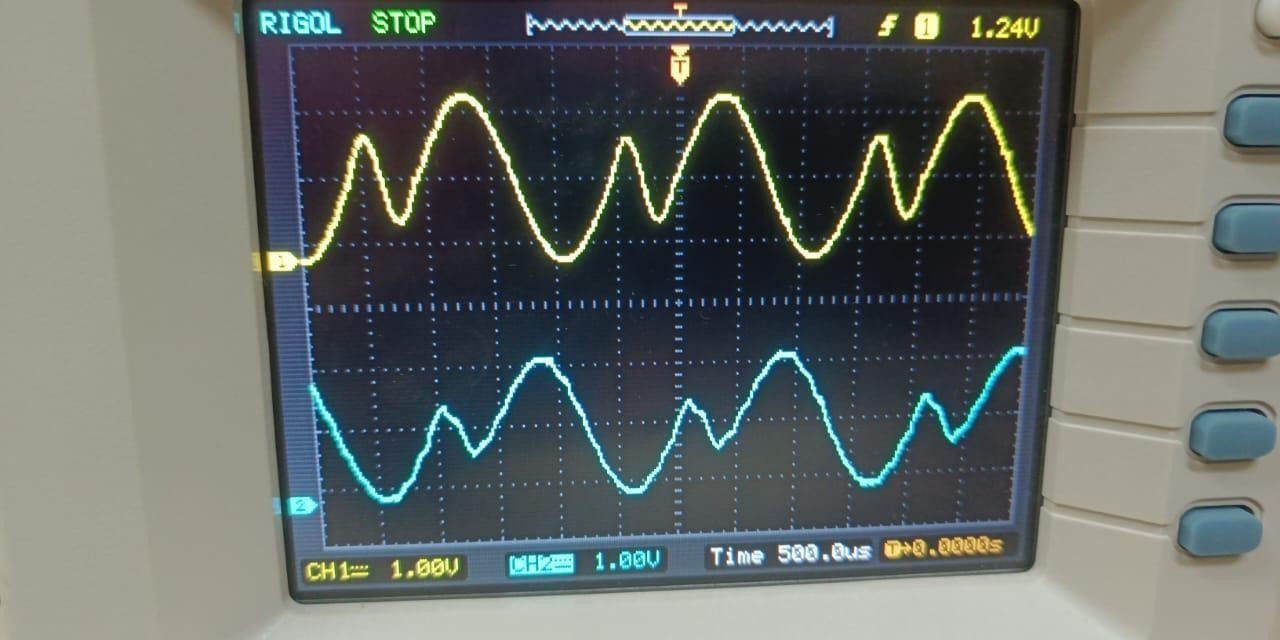
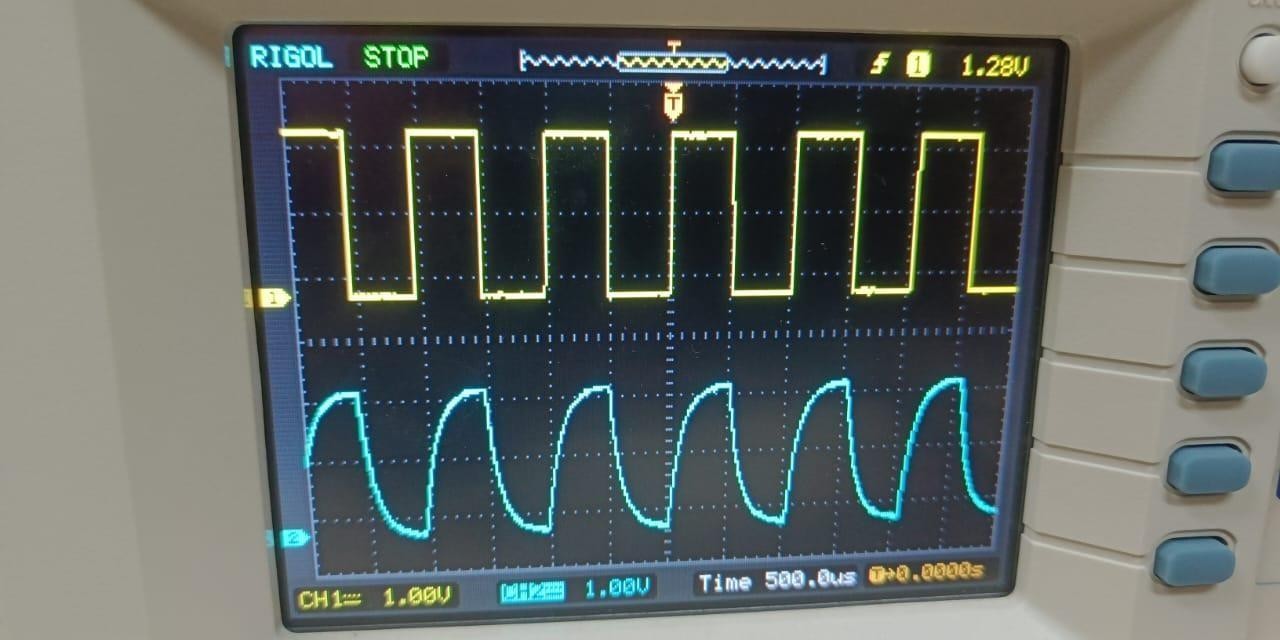


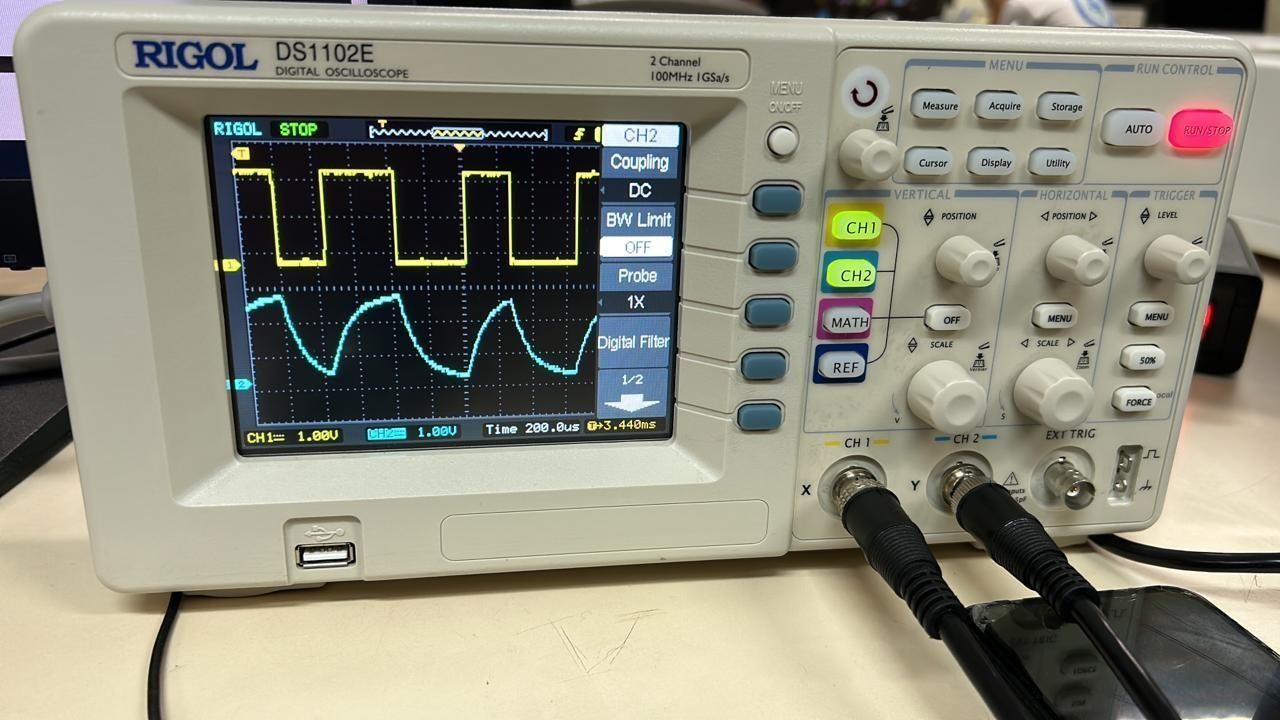
# Experiment 7

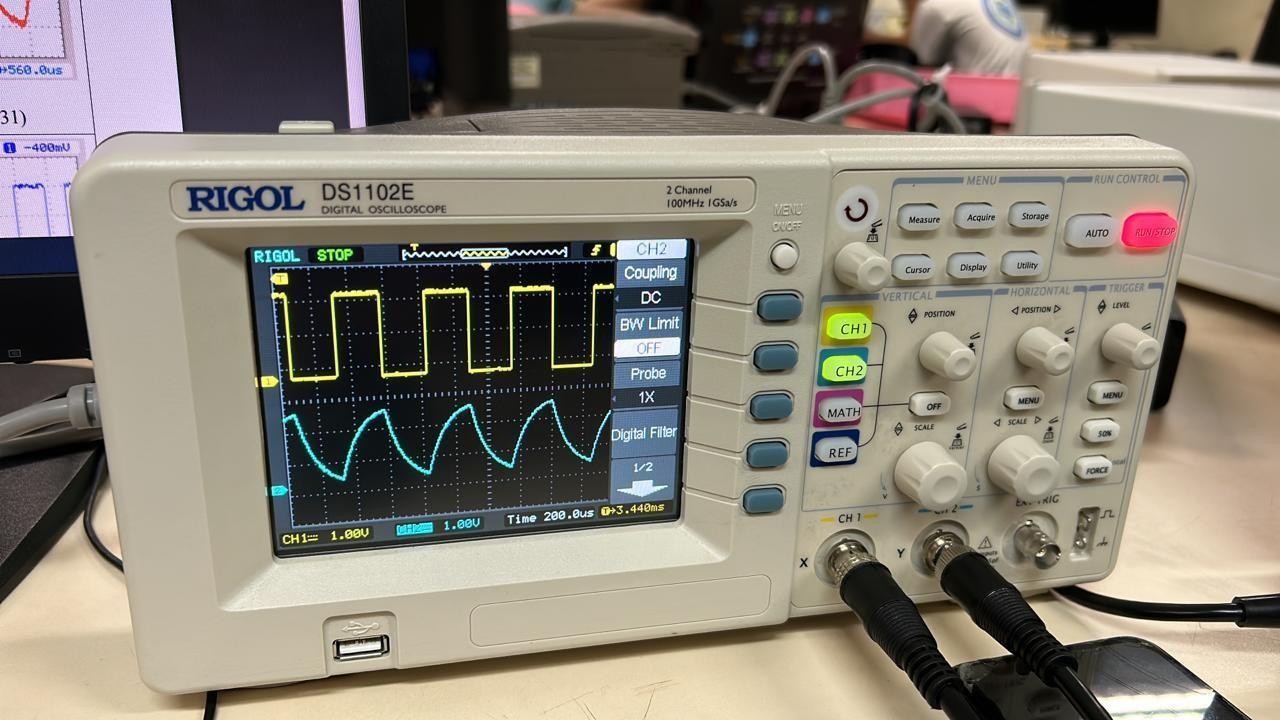
# Experiment 7.1, 7.2, 7.3, 7.4 , 7.5, 7.6,7.7, 7.8, 7.9 respectively

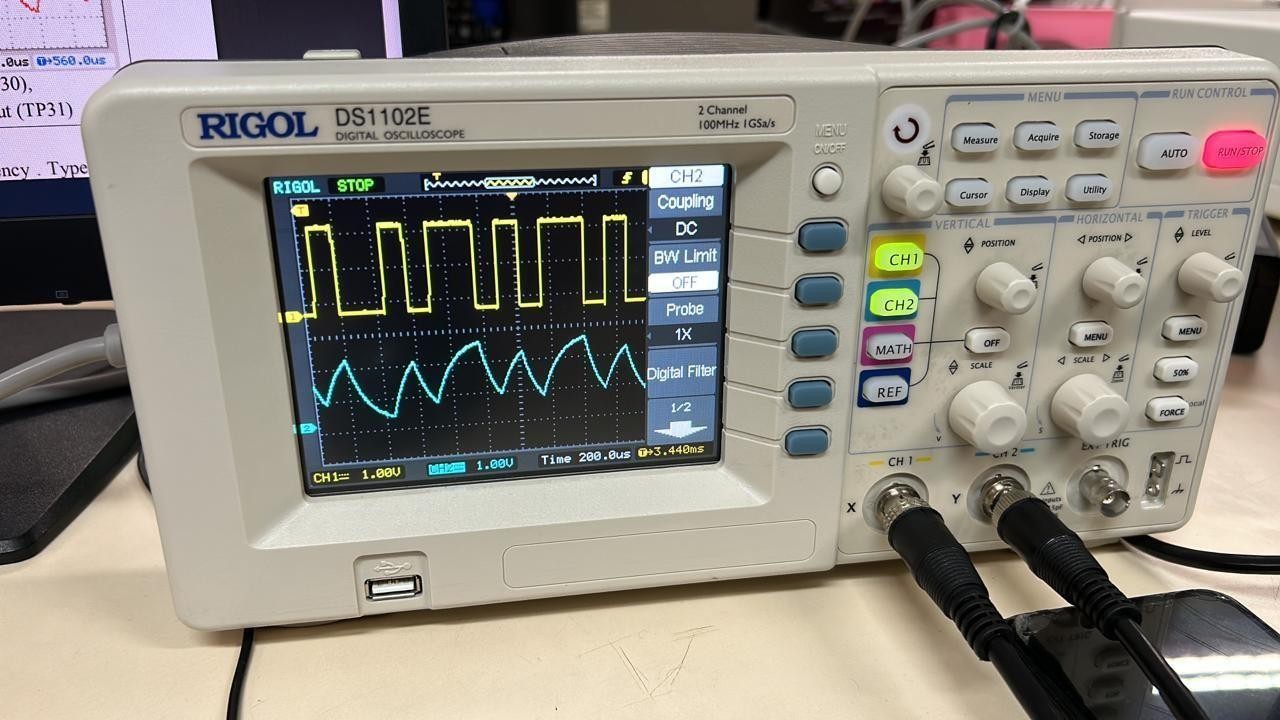










clear;clf; td= 0.002 ; t = [0:td:2];

xsig=cos (10\*pi\*t) +cos(16\*pi\*t); Lsig=length(xsig);

ts=0.02;

Nfactor=ts / td;

[ s\_out , sq\_out , sqh\_out , Delta , SQNR ] = sampandquant (xsig,16,td,ts) ;

Lfft= 2.^ceil(log2(Lsig)+1) ; Fmax= 1/ (2\*td ) ;

Faxis=linspace(-Fmax ,Fmax , Lfft ) ; Xsig=fftshift ( fft ( xsig , Lfft ) ); S\_out=fftshift ( fft ( s\_out ,Lfft ) );

figure (1) ;

subplot (311) ; sfig1a=plot(t,xsig , ' k ');

hold on ; sfig1b=plot(t,s\_out(1:Lsig) ,' b ' ) ; hold off ;

set(sfig1b , 'LineWidth' , 2); xlabel( ' time ( sec ) ');

title( ' Signal {\it g}({ \it t } ) and its uniform samples ' ) ; subplot(312); sfig1c=plot(Faxis , abs(Xsig));

xlabel( ' frequency ( Hz ) '); axis([ - 300 300 0 600])

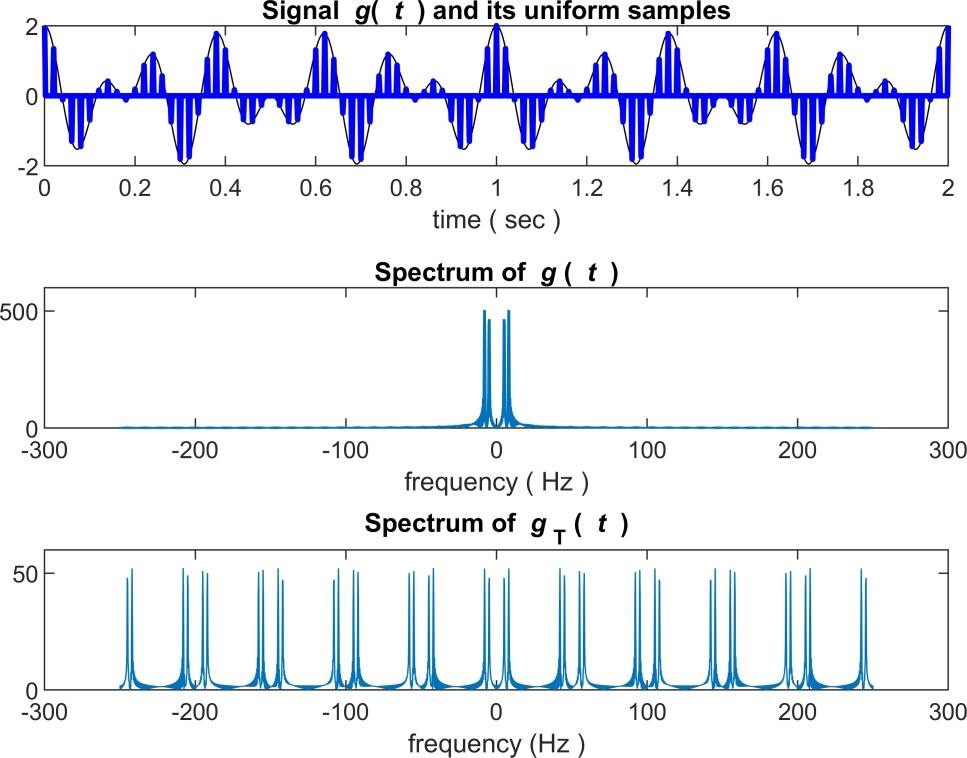
set(sfig1c , 'LineWidth' , 1 ) ; title ( ' Spectrum of {\it g} ( {\it t } ) '); subplot(313) ; sfig1d=plot(Faxis , abs( S\_out ) );

xlabel ( ' frequency (Hz ) ');

axis ( [ - 300 300 0 600 /Nfactor ] )

set(sfig1c , 'LineWidth' , 1 ); title ( ' Spectrum of {\it g }\_T ( {\it t } ) ');

1



% calculate the recons tructed signal from ideal sampl ing and

BW=100 ;

H\_lpf= zeros(1, Lfft); H\_lpf(Lfft/2-BW : Lfft/2+BW-1) = 1 ; S\_recv=Nfactor\* S\_out .\* H\_lpf ;

s\_recv=real(ifft(fftshift(S\_recv))) ;

s\_recv= s\_recv(1:Lsig) ;

figure(2)

subplot (211) ; sfig2a=plot(Faxis , abs(S\_recv) ); xlabel ( ' frequency ( Hz ) ');

axis ( [ -300 300 0 600] );

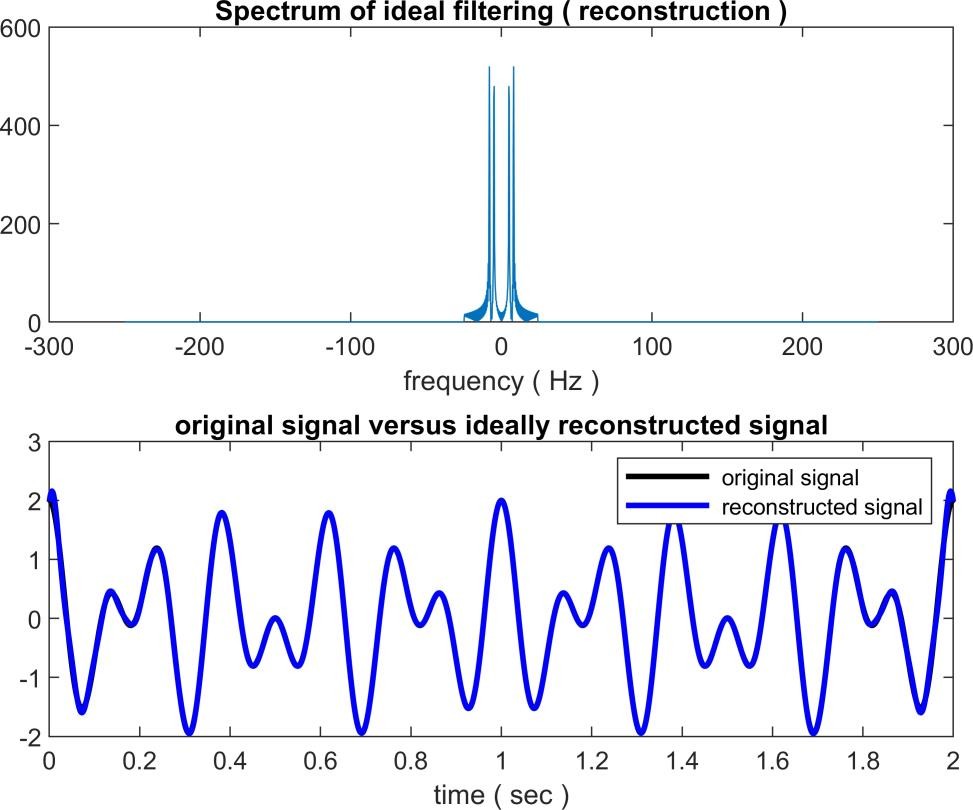
title ( ' Spectrum of ideal filtering ( reconstruction ) ');

subplot(212 ) ; sfig2b=plot(t,xsig , ' k- . ' , t , s\_recv(1 : Lsig) , ' b '); legend( ' original signal ', ' reconstructed signal ');

xlabel( ' time ( sec ) ');

title( ' original signal versus ideally reconstructed signal '); set( sfig2b, 'LineWidth' ,2) ;

2



% non-ideal reconstruction ZOH=ones(1, Nfactor ) ;

s\_ni =kron( downsample( s\_out , Nfactor ),ZOH) ; S\_ni = fftshift( fft( s\_ni , Lfft ) );

S\_recv2 =S\_ni .\*H\_lpf; % ideal filtering

s\_recv2 =real(ifft( fftshift( S\_recv2 ))); % reconstructed f-domain s\_recv2 =s\_recv2 (1:Lsig) ; % reconstructed t-domain

figure (3)

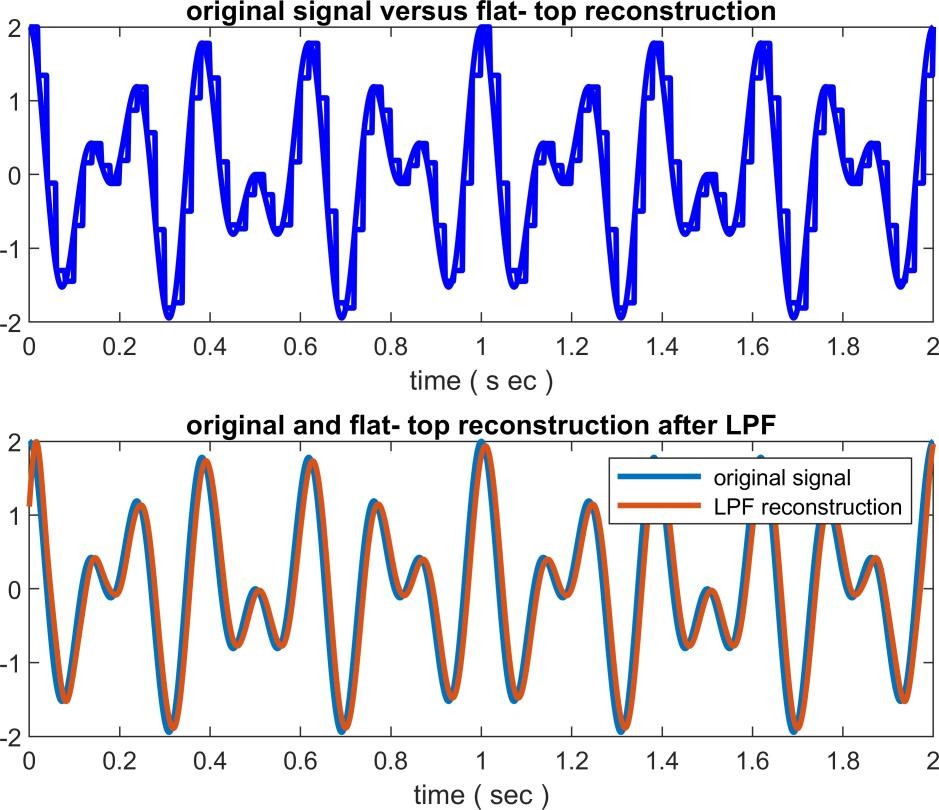
subplot(211); sfig3a=plot(t,xsig , 'b' , t , s\_ni (1:Lsig) , ' b '); xlabel( ' time ( s ec ) ');

title ( ' original signal versus flat- top reconstruction '); subplot(212); sfig3b=plot(t,xsig , t , s\_recv2 (1:Lsig) ) ; legend( ' original signal ', ' LPF reconstruction ');

xlabel ( ' time ( sec ) ');

set( sfig3a, 'LineWidth' ,2) ; set(sfig3b , 'LineWidth' ,2) ; title ( ' original and flat- top reconstruction after LPF ' );

3



function [ q\_out ,Delta , SQNR]=uniquan( sig\_in , L )

sig\_pmax=max( sig\_in ) ;

sig\_nmax=min( sig\_in ) ;

Delta= ( sig\_pmax- sig\_nmax ) /L;

q\_level=sig\_nmax+Delta/2 : Delta : sig\_pmax-Delta / 2 ; L\_sig=length( sig\_in ) ;

sigp=(sig\_in- sig\_nmax ) / Delta+1/2 ;

qindex=round( sigp ) ;

qindex=min ( qindex , L ) ;

q\_out =q\_level(qindex) ;

SQNR=20.\*log10(norm(sig\_in)/norm( sig\_in-q\_out ));

end

function [ s\_out , sq\_out , sqh\_out , Delta , SQNR] = sampandquant ( sig\_in , L , td , ts)

if ( rem(ts/td , 1 ) ==0 )

nfac=round( ts/td ) ; p\_zoh=ones(1,nfac ) ; s\_out=downsample( sig\_in , nfac ) ;

[ sq\_out , Delta , SQNR] =uniquan( s\_out , L ) ; s\_out=upsample( s\_out,nfac ) ; sqh\_out=kron(sq\_out ,p\_zoh ) ;

sq\_out=upsample( sq\_out , nfac ); else

warning( 'Error ! ts / td is not an integer ! ');

s\_out= [] ; sq\_out= [] ; sqh\_out= [] ; Delta= [] ; SQNR= [] ;

end

4

else

warning( 'Error ! ts / td is not an integer ! ');

s\_out= [] ; sq\_out= [] ; sqh\_out= [] ; Delta= [] ; SQNR= [] ; end

end