Name – Aarsh Bhavsar

Student ID – 202101474 Lab - 2

1. (Tutorial Exercise) 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)

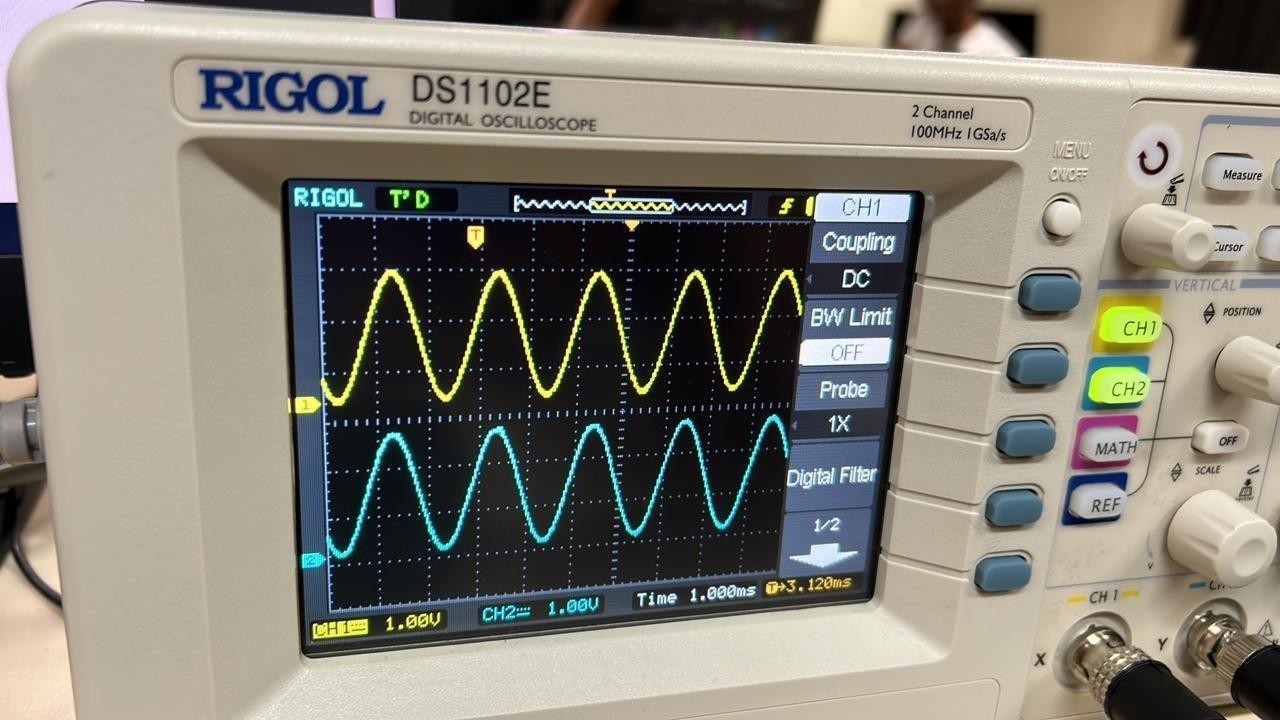
Here, the expression can be expanded as x(t) = sin(100πt)/100πt + 3\*(1-cos(120πt))/2/(60πt)^2, so, the maximum frequency component here is 120π Hz. So, f = 120π/2π = 60Hz which implies Nyquist rate = 2\*60Hz = 120Hz and Nyquist interval = 1/120s = 0.008333333333s.

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

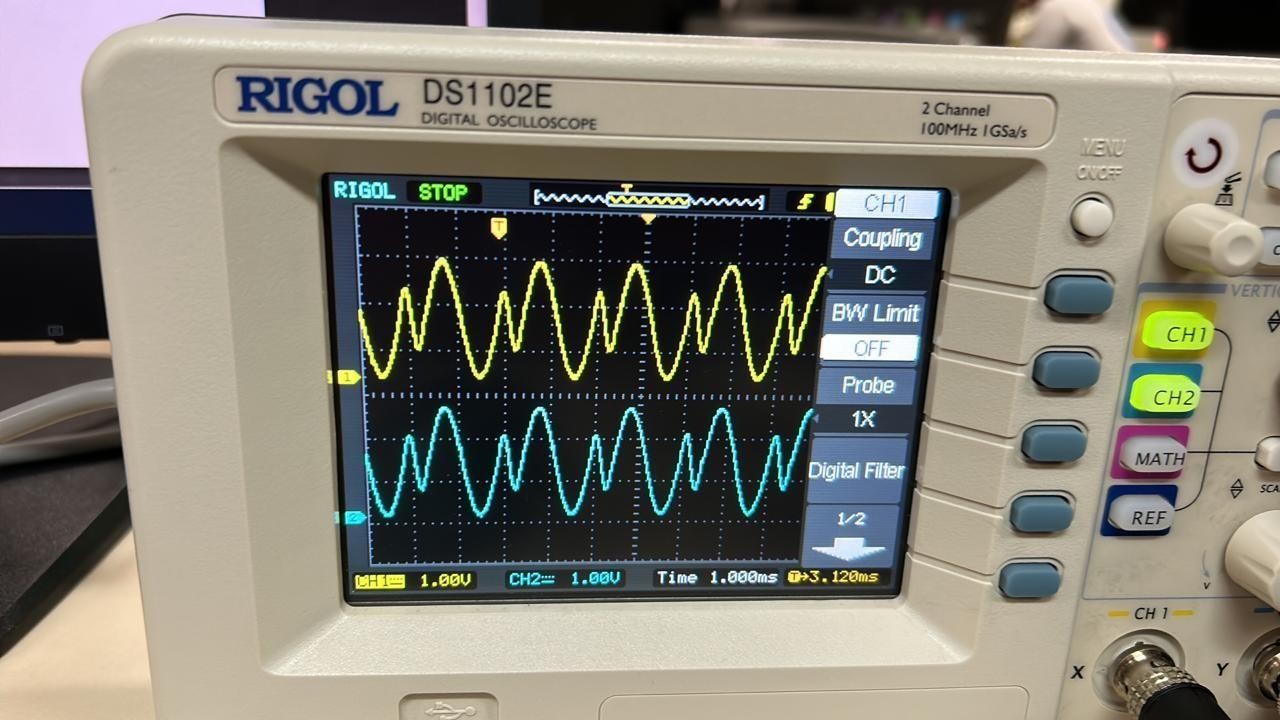
Here, the expression has maximum frequency component as 4000π Hz. So, f = 4000π/2π Hz = 2000Hz and Nyquist rate = 2\*2000Hz = 4000Hz. The Nyquist interval = 1/4000s = 0.00025s.

1. MATLAB Exercise

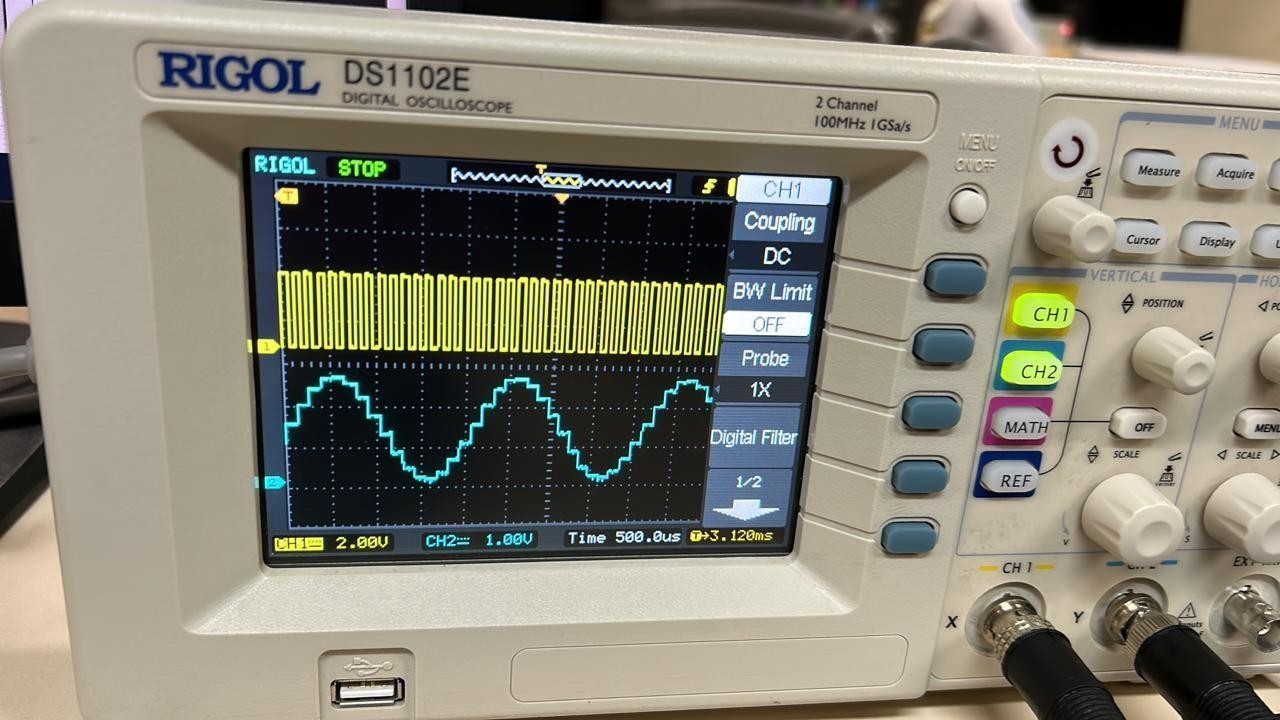
# Experiment 1

3.

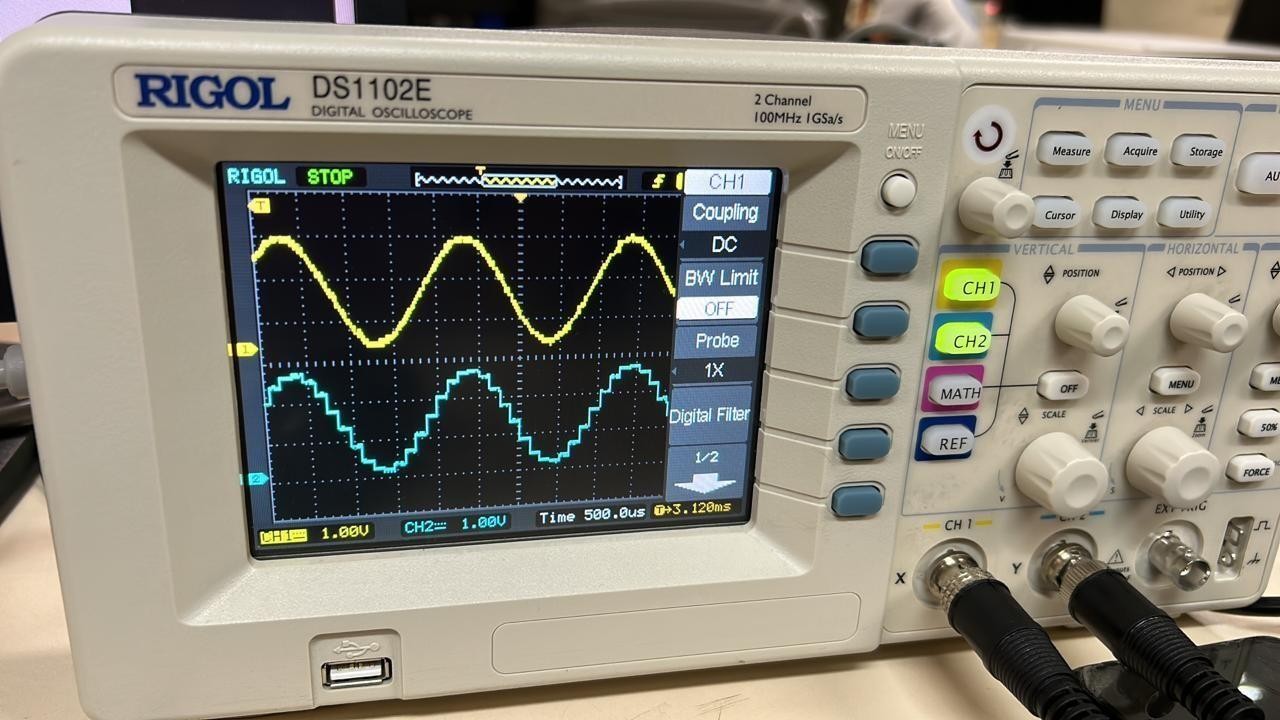
1.1



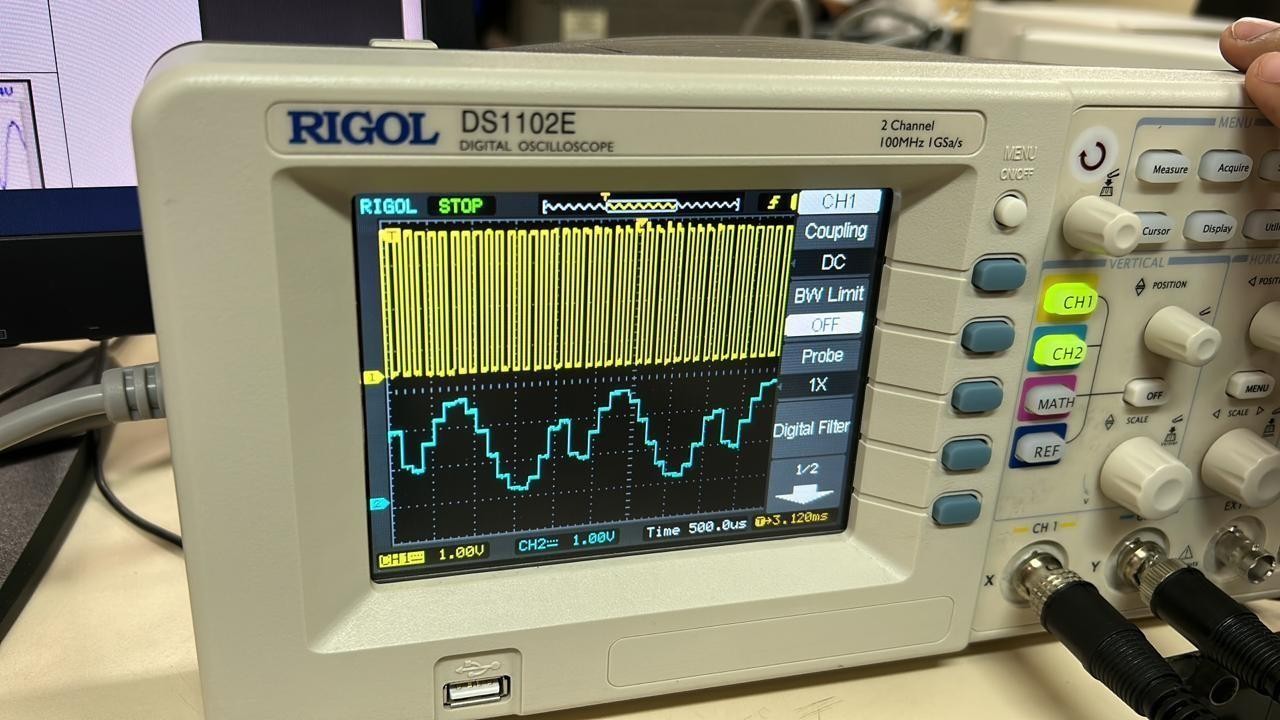
1.2



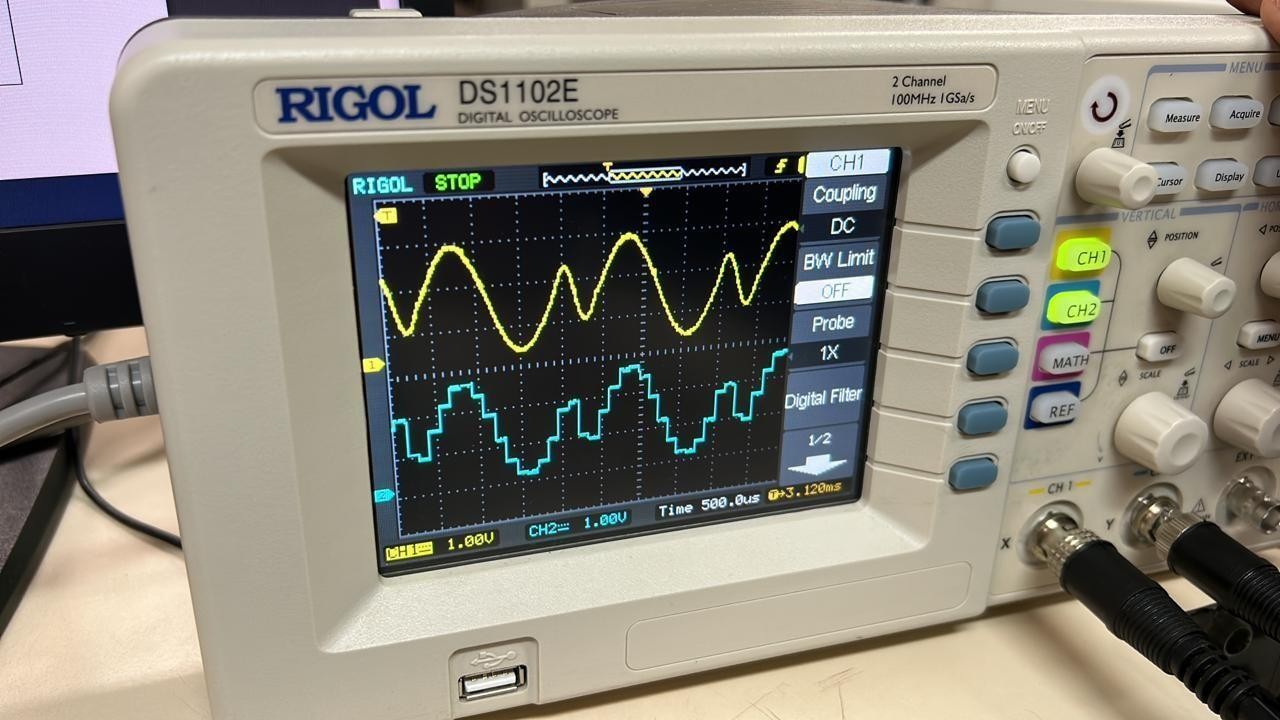
1.3



1.4

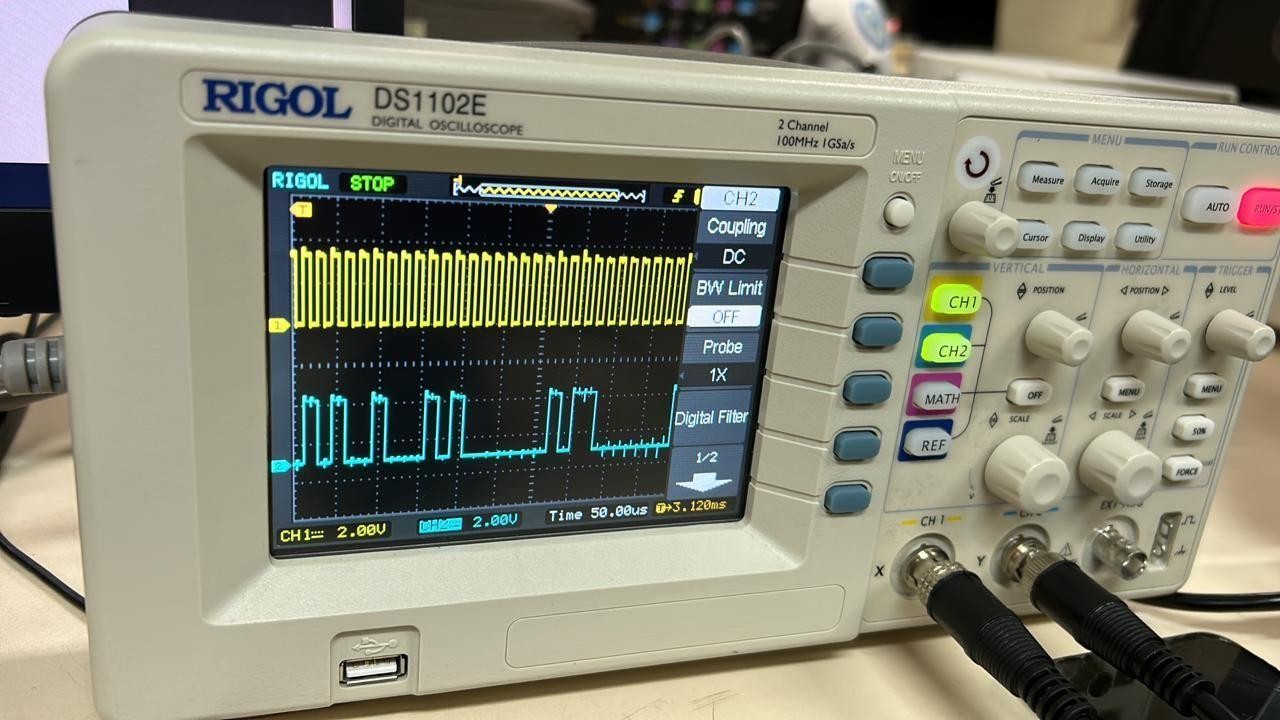


1.5

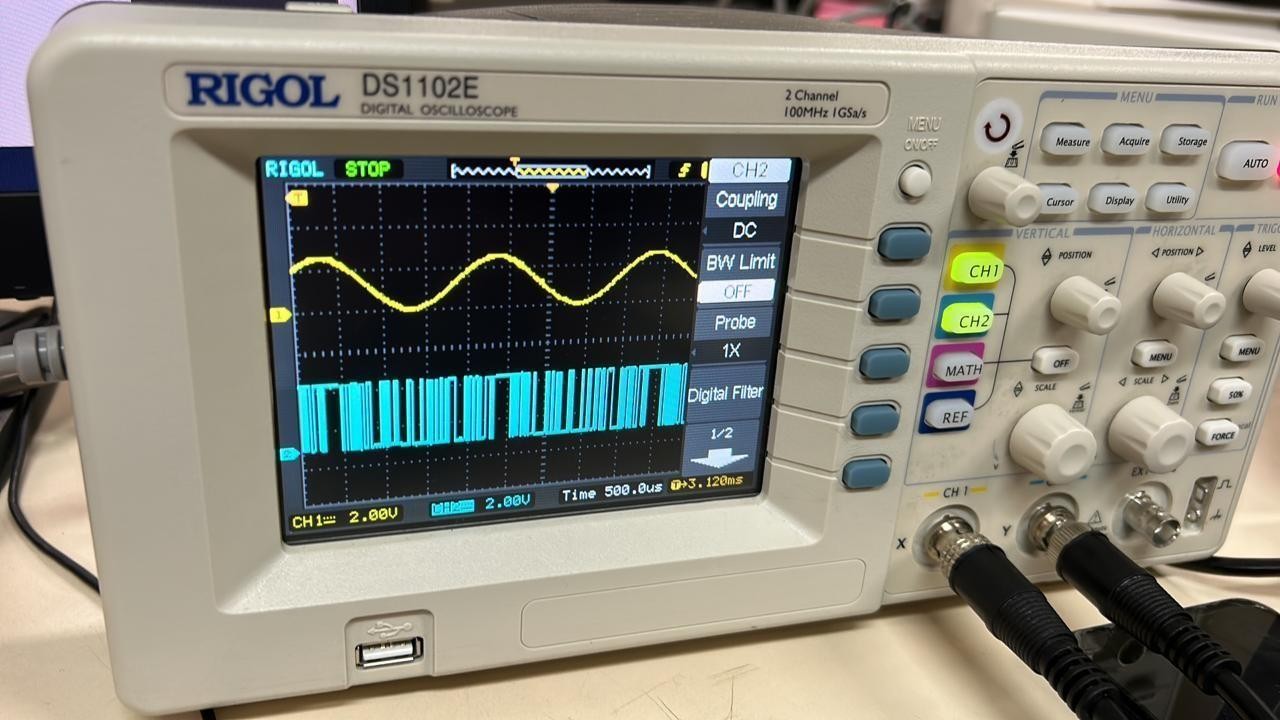


1.6

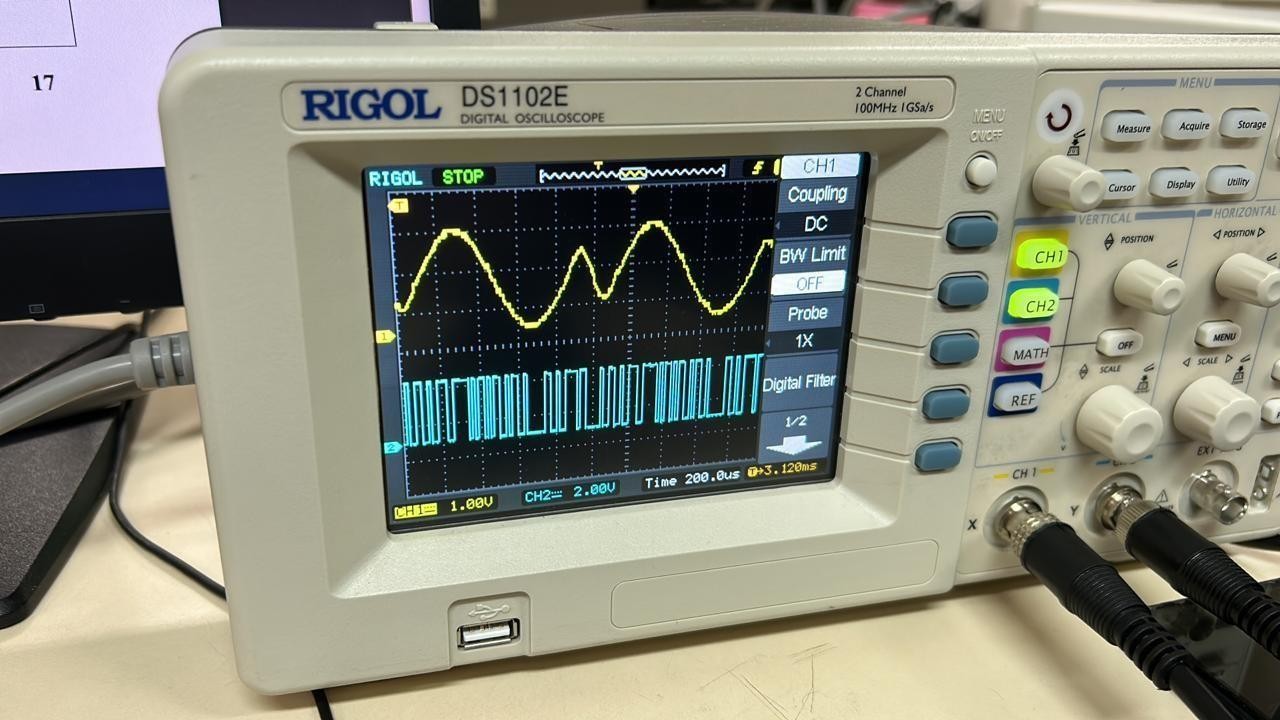
# Experiment 2



2.1

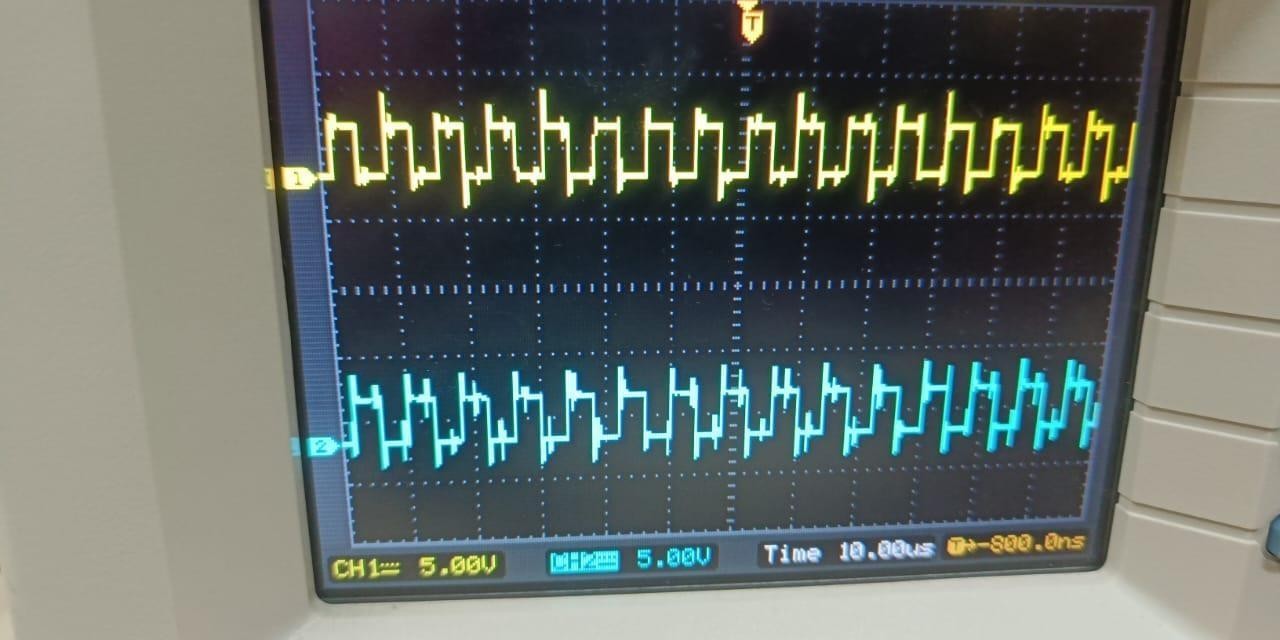


2.2



2.3

# Experiment 3



3.1



3.2

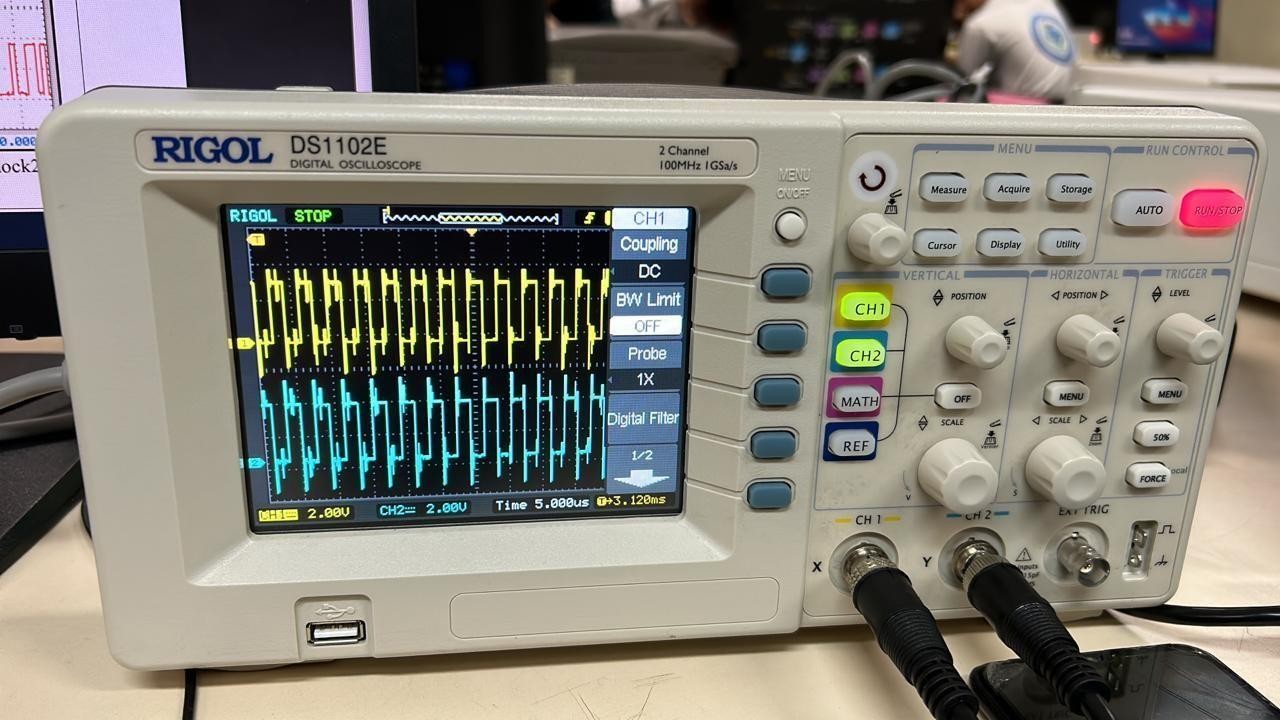


3.3

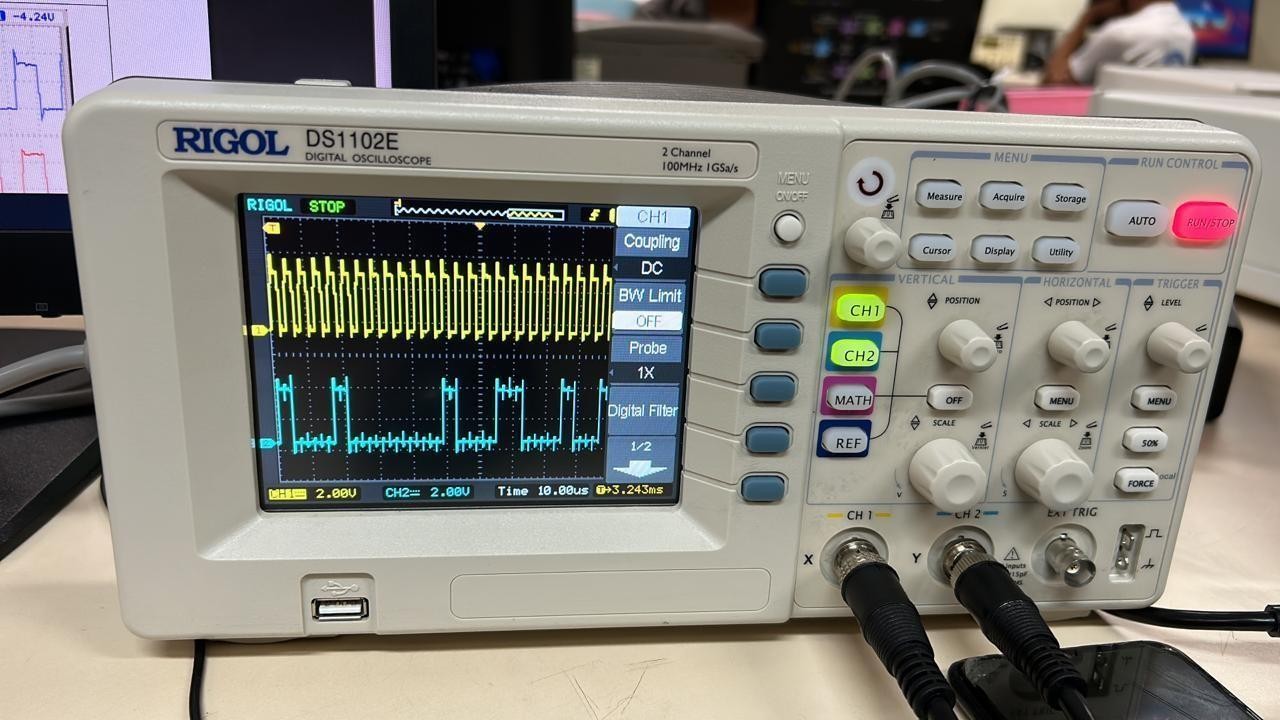


3.4

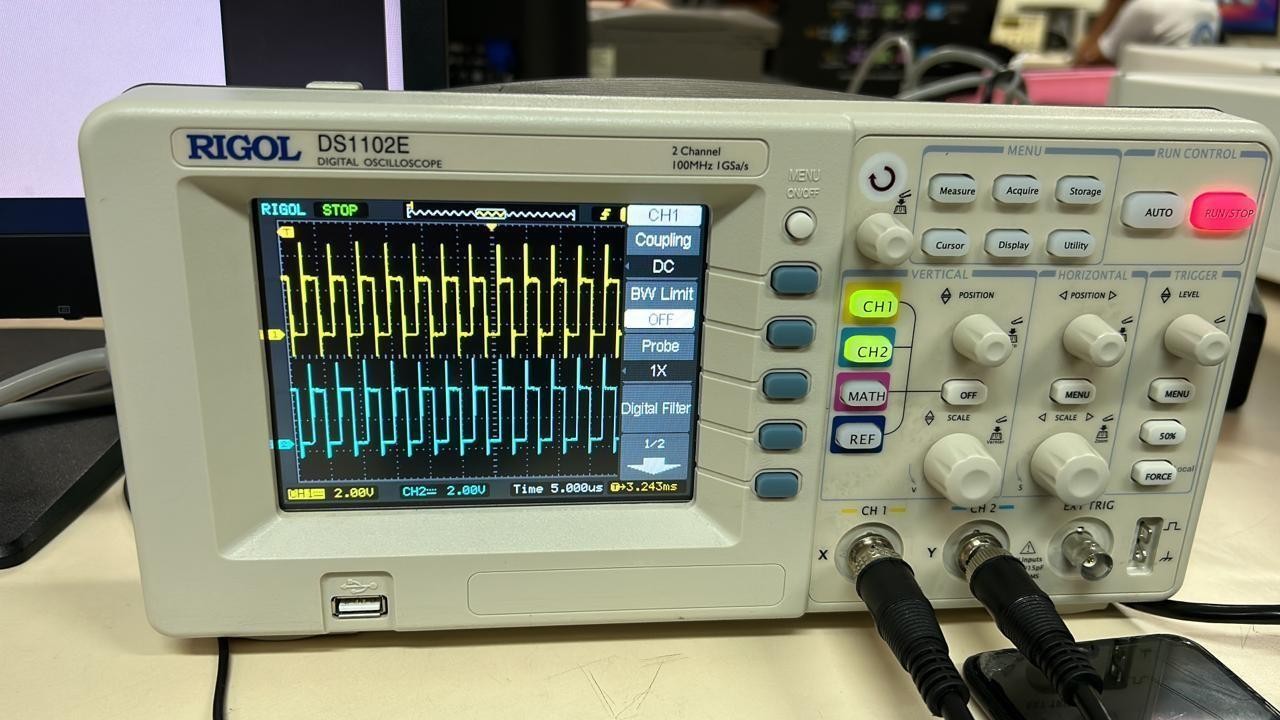
# Experiment 4



4.1

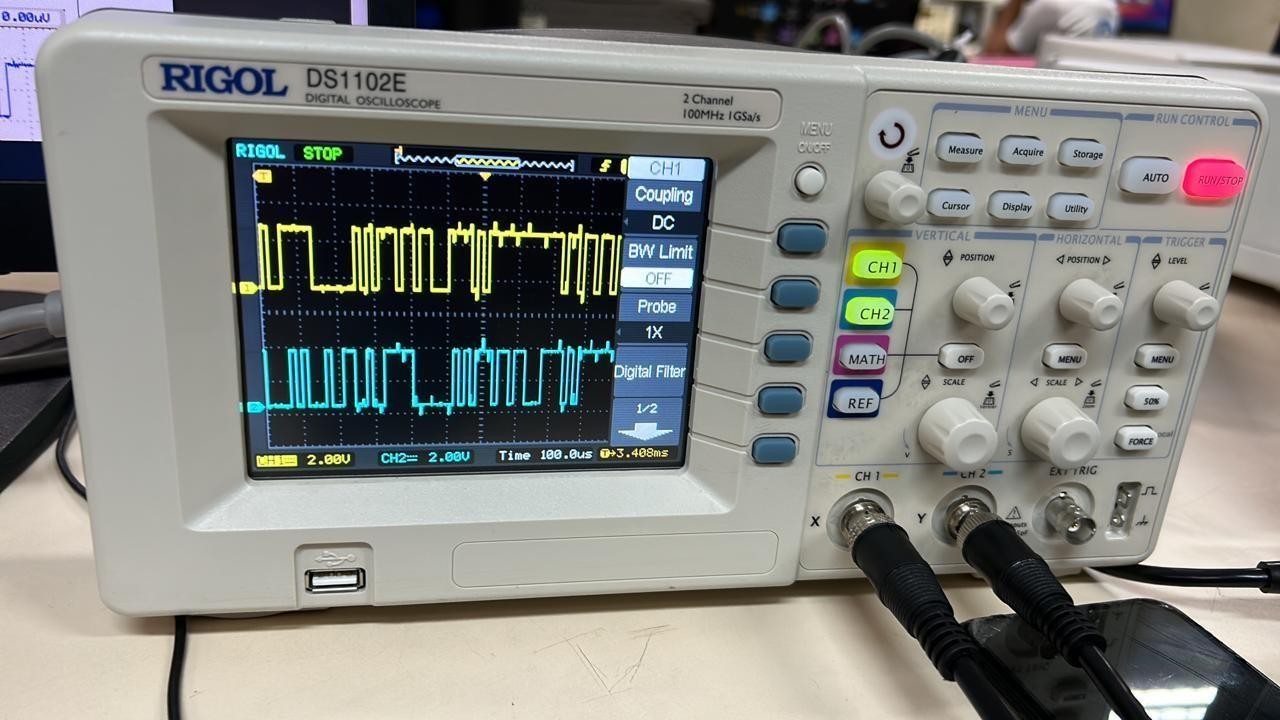


4.2

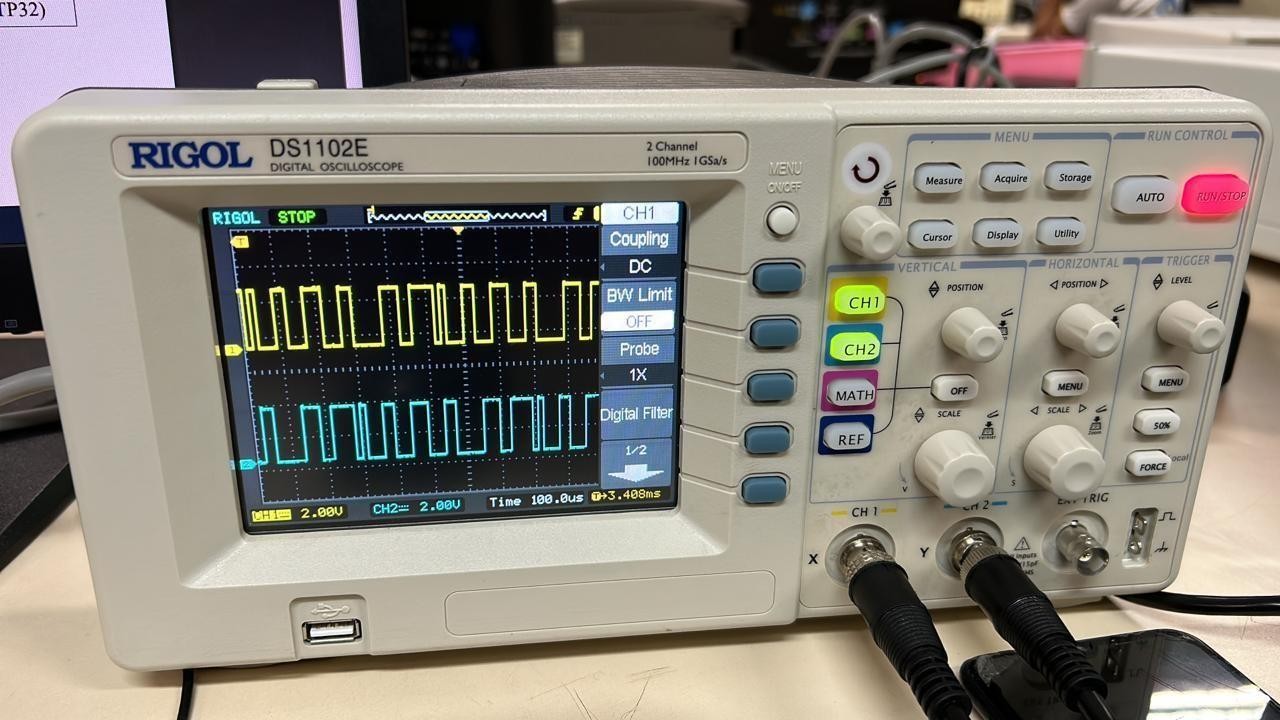


4.3

# Experiment 5

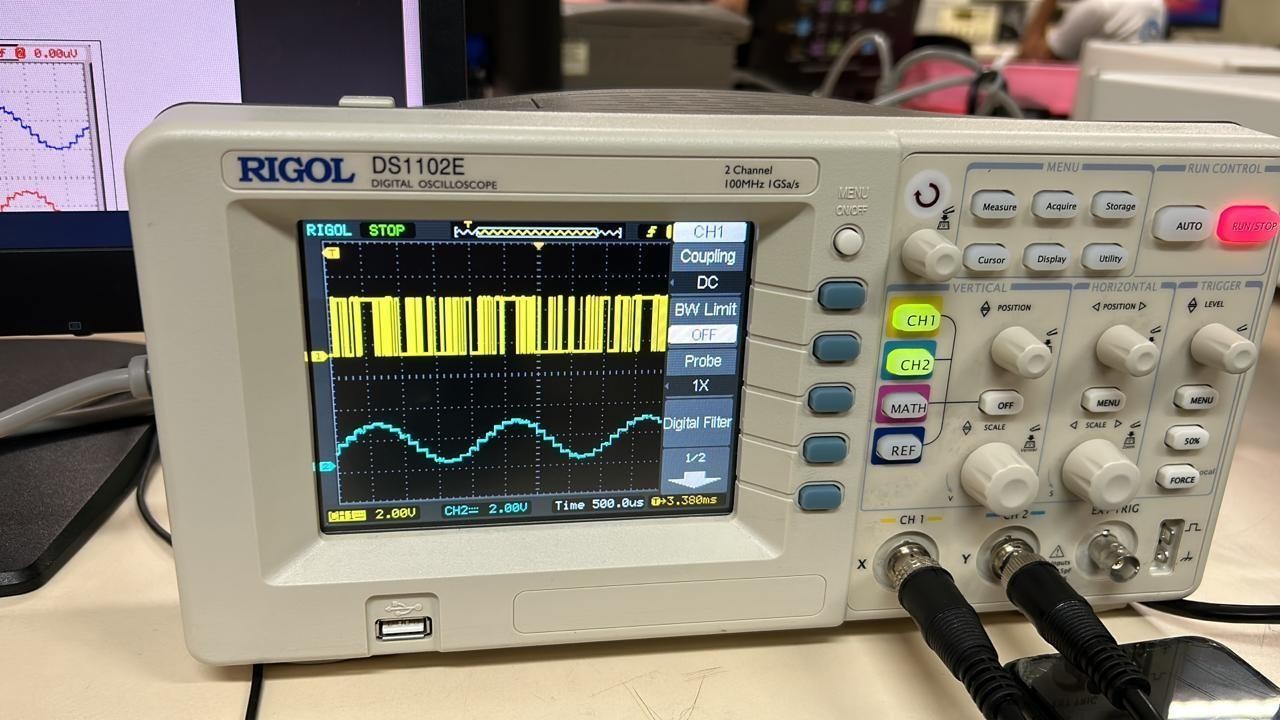


5.1

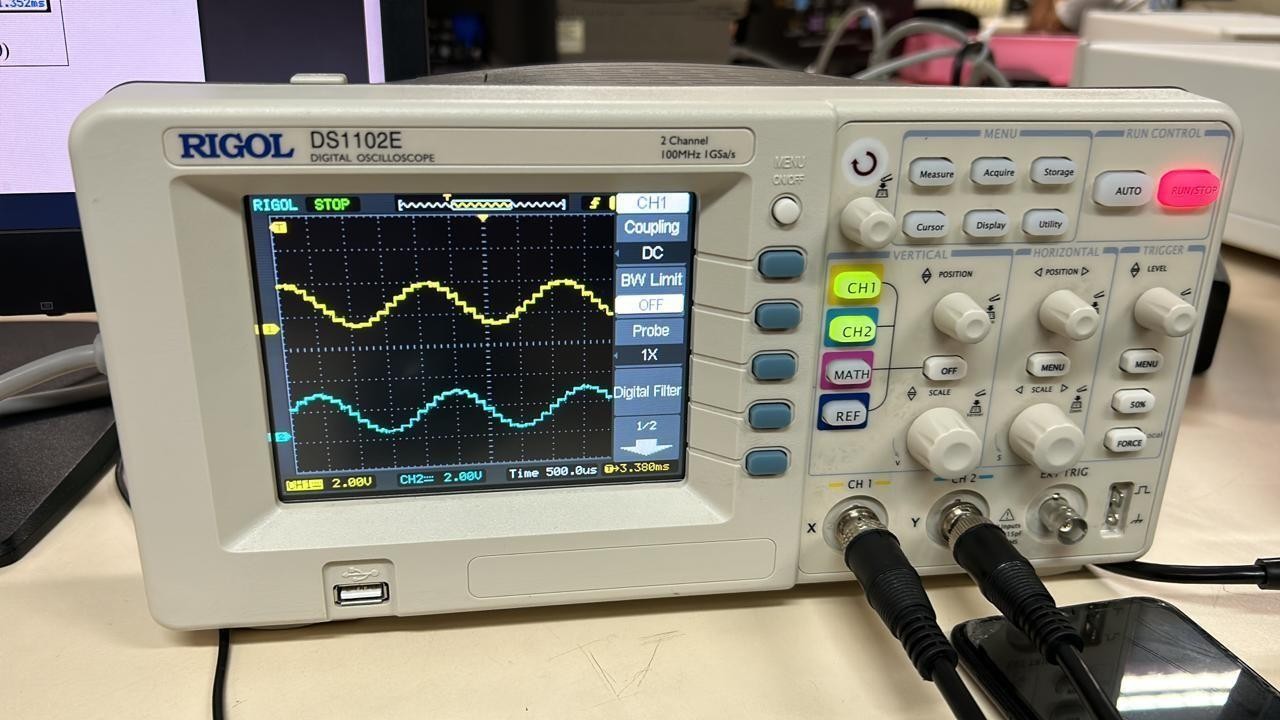


5.2

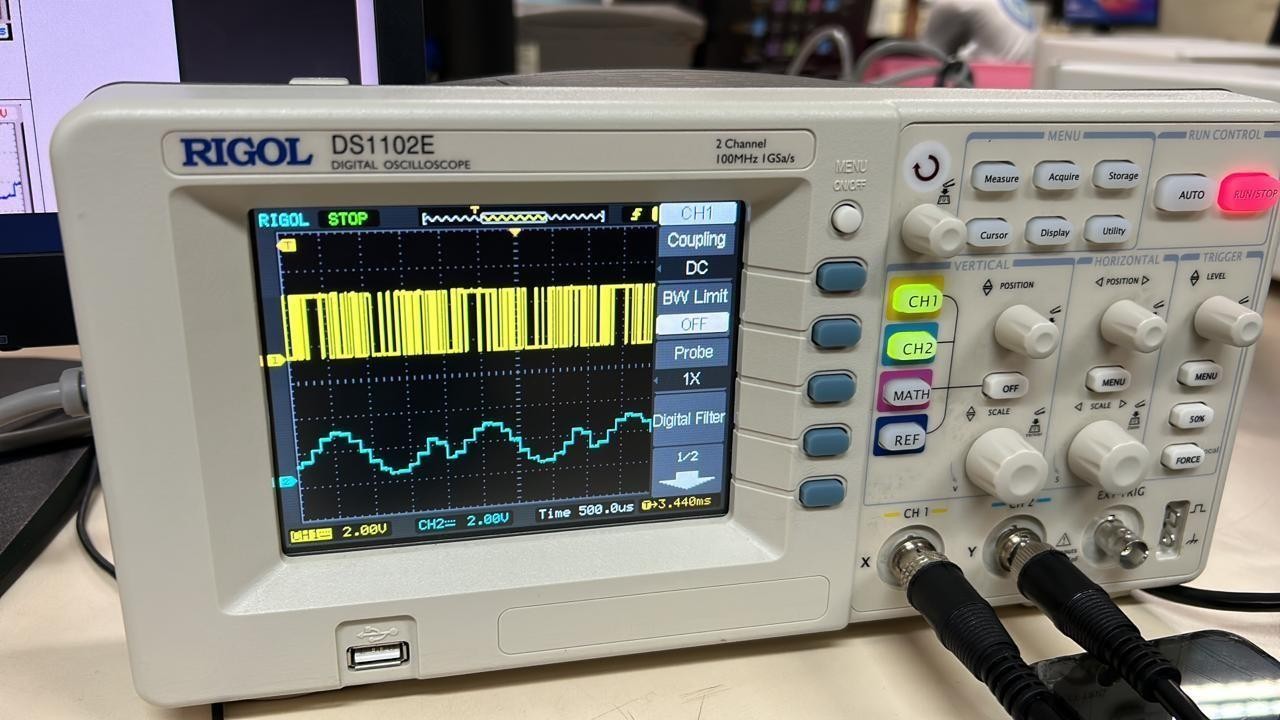
# Experiment 6



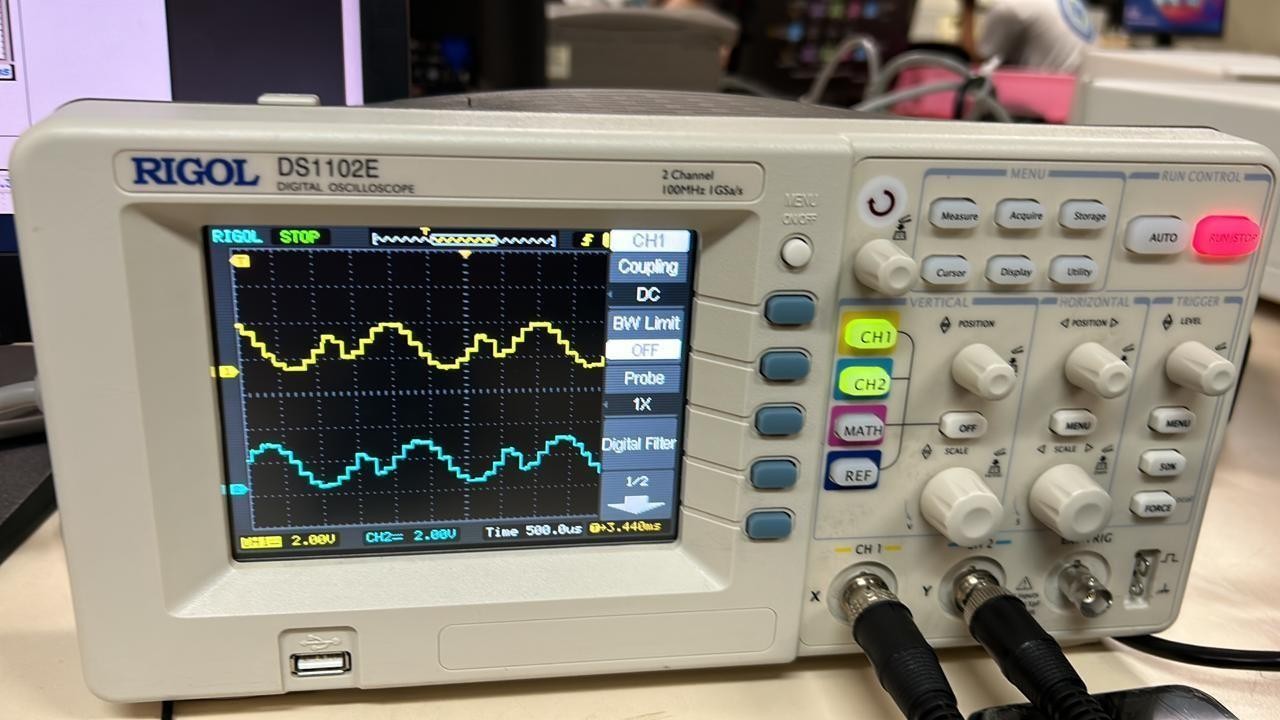
6.1



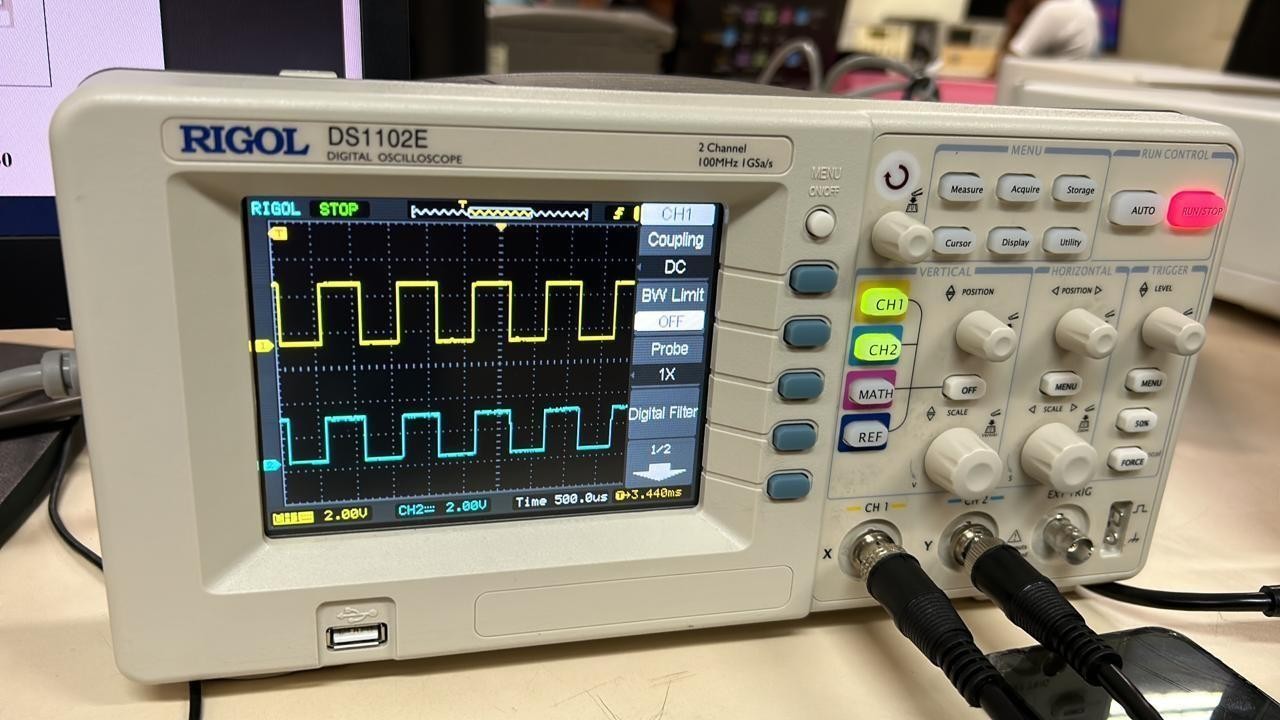
6.2



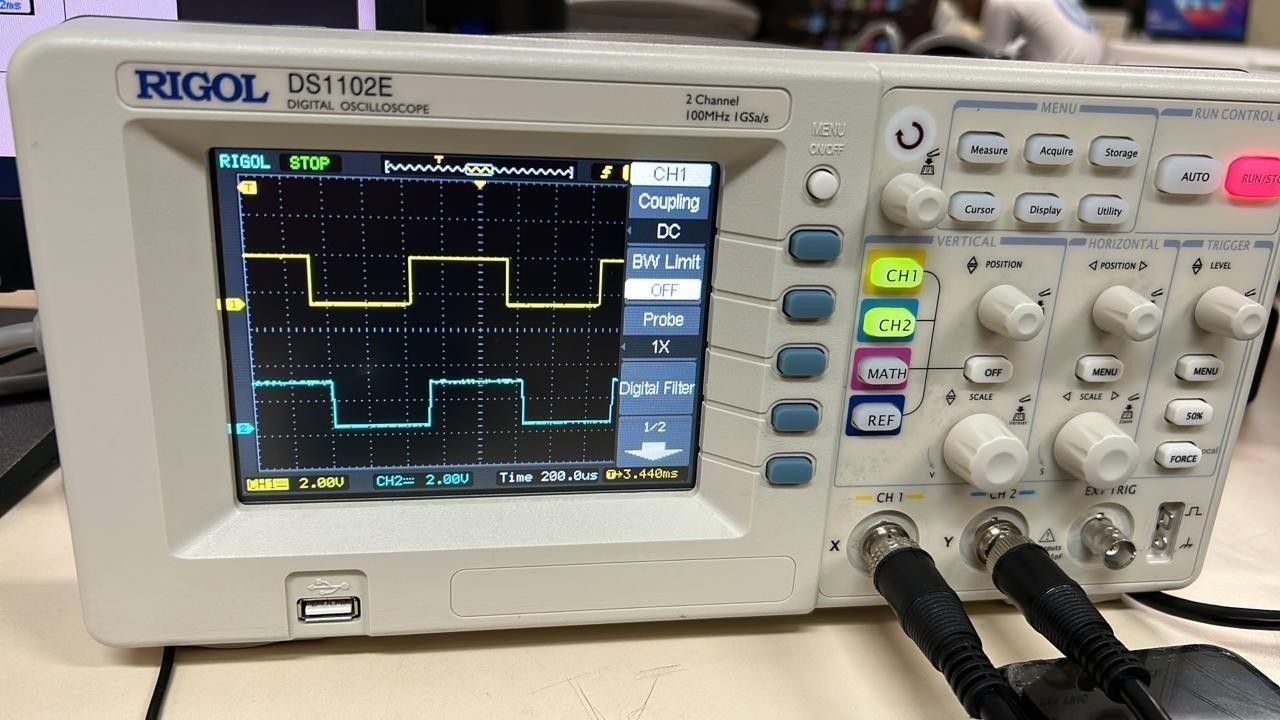
6.3



6.4

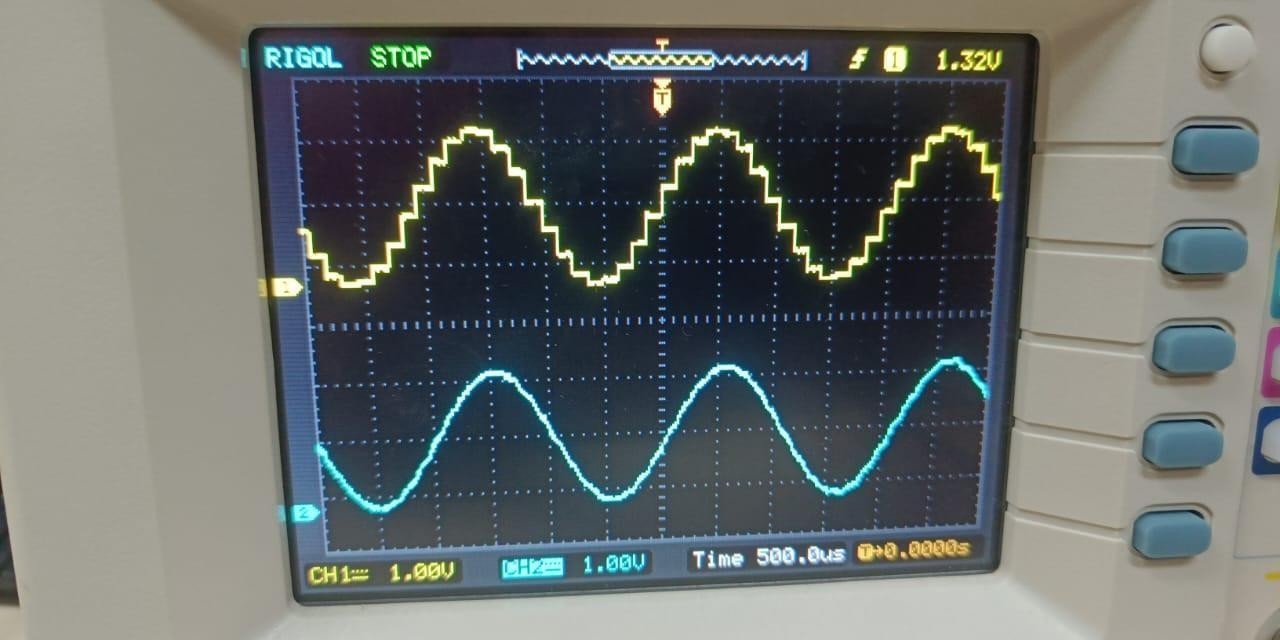


6.5

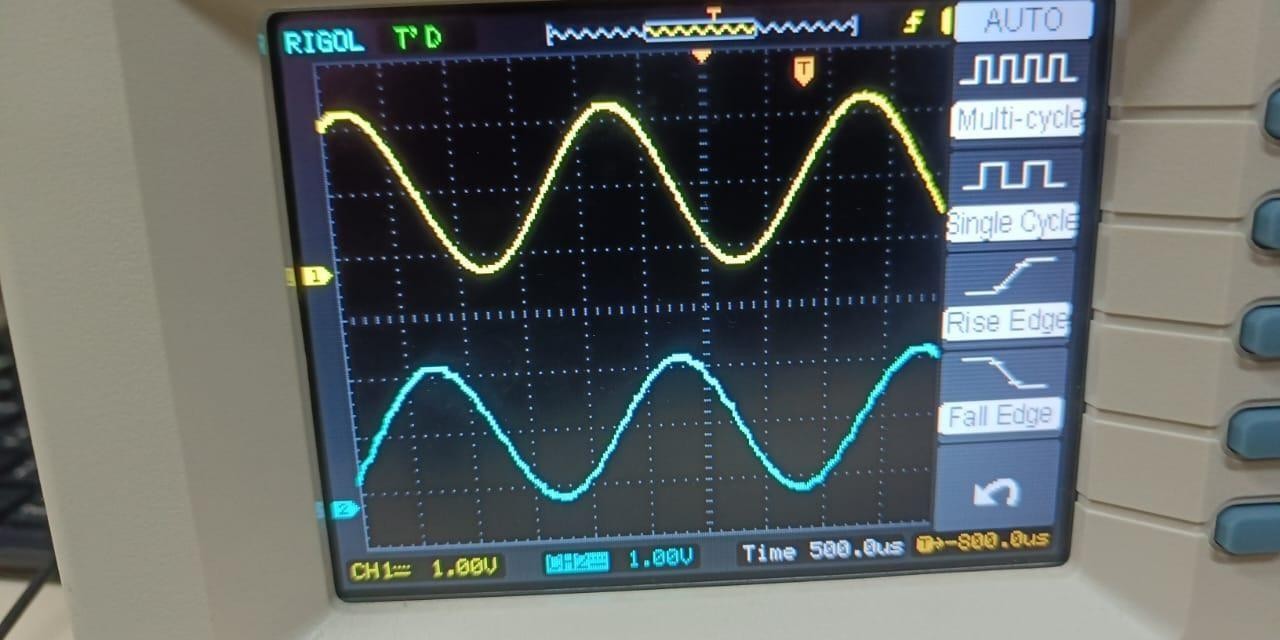


6.6

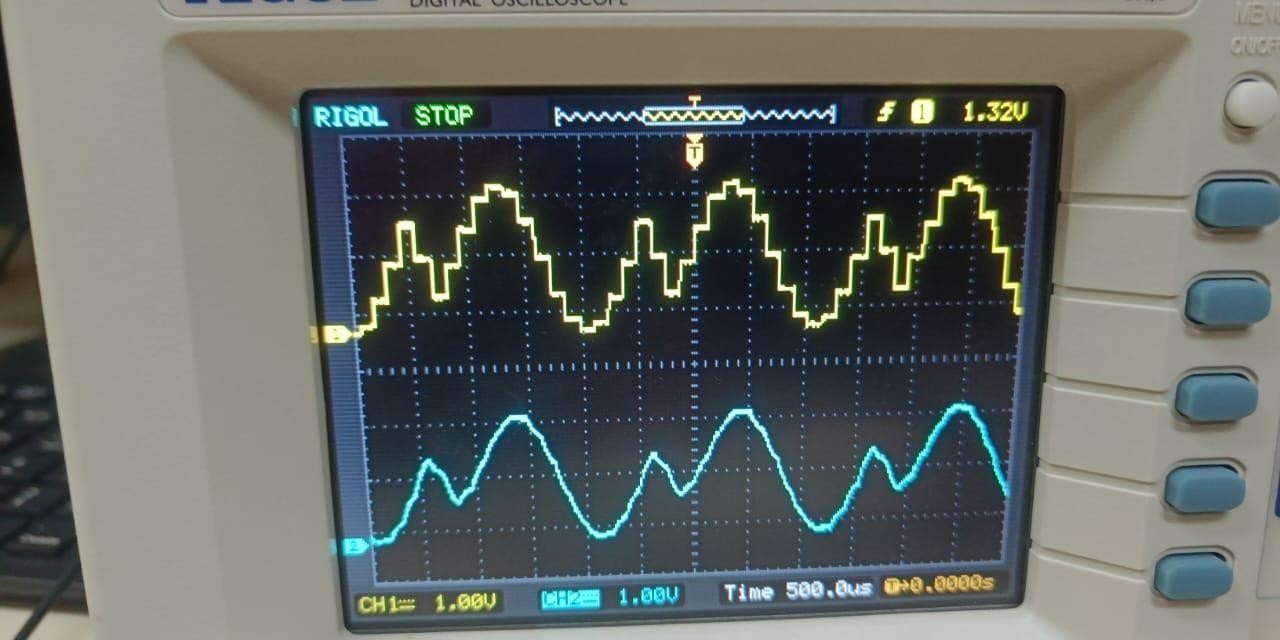
# Experiment 7



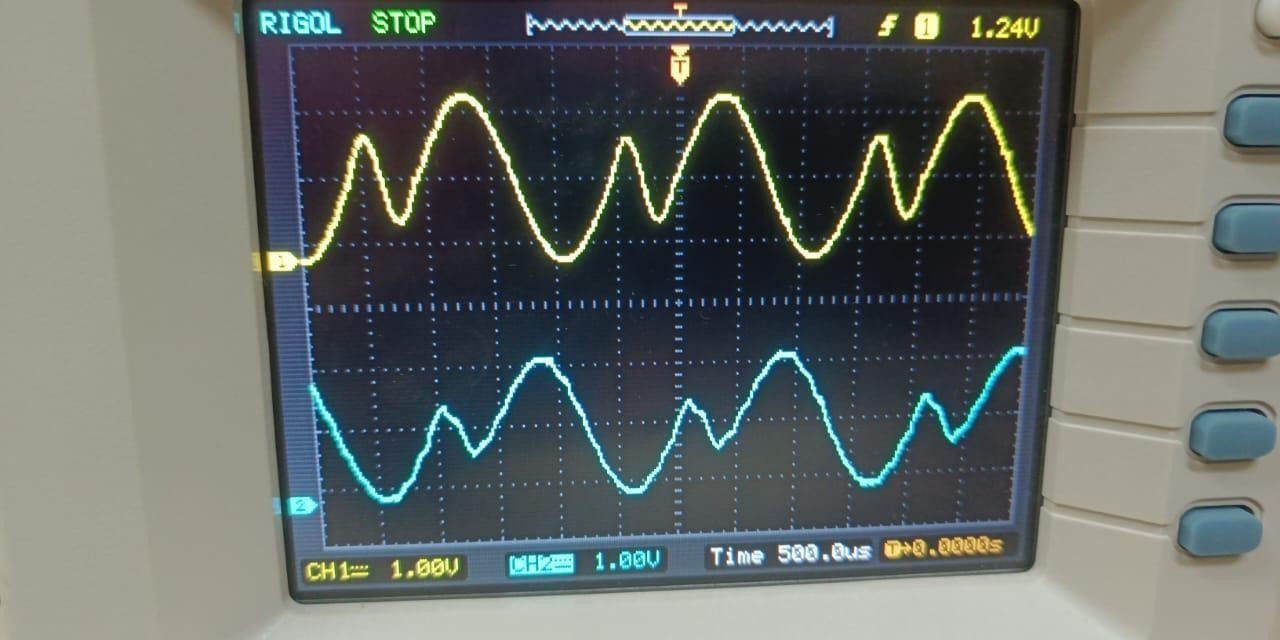
7.1



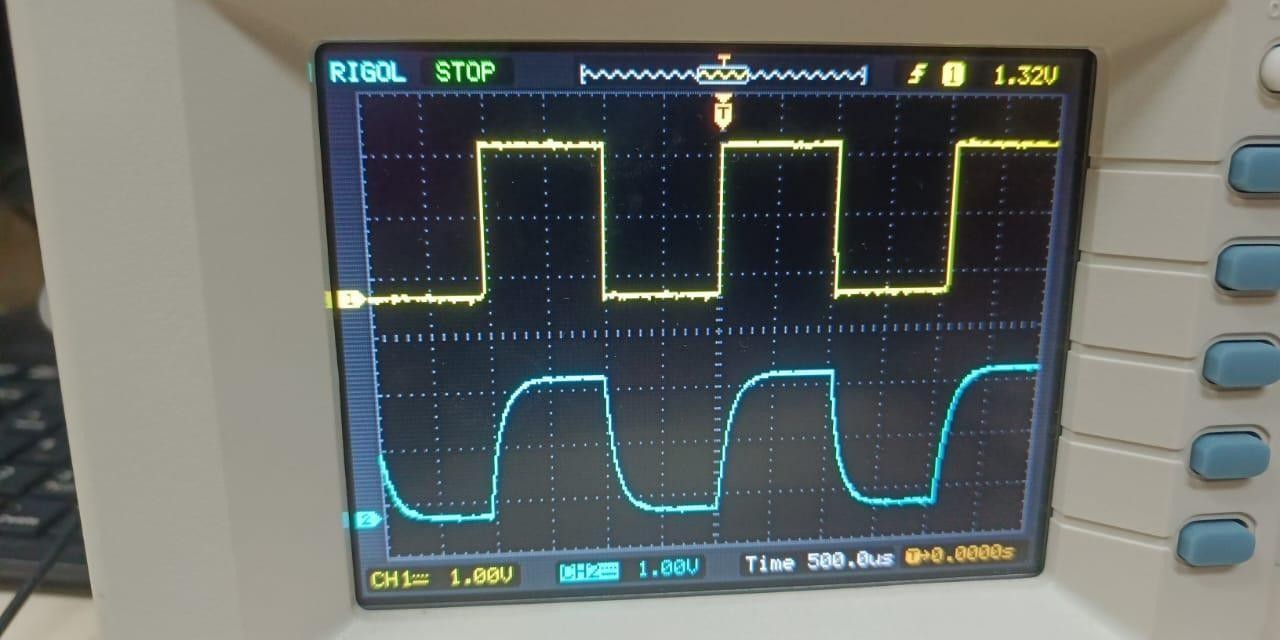
7.2



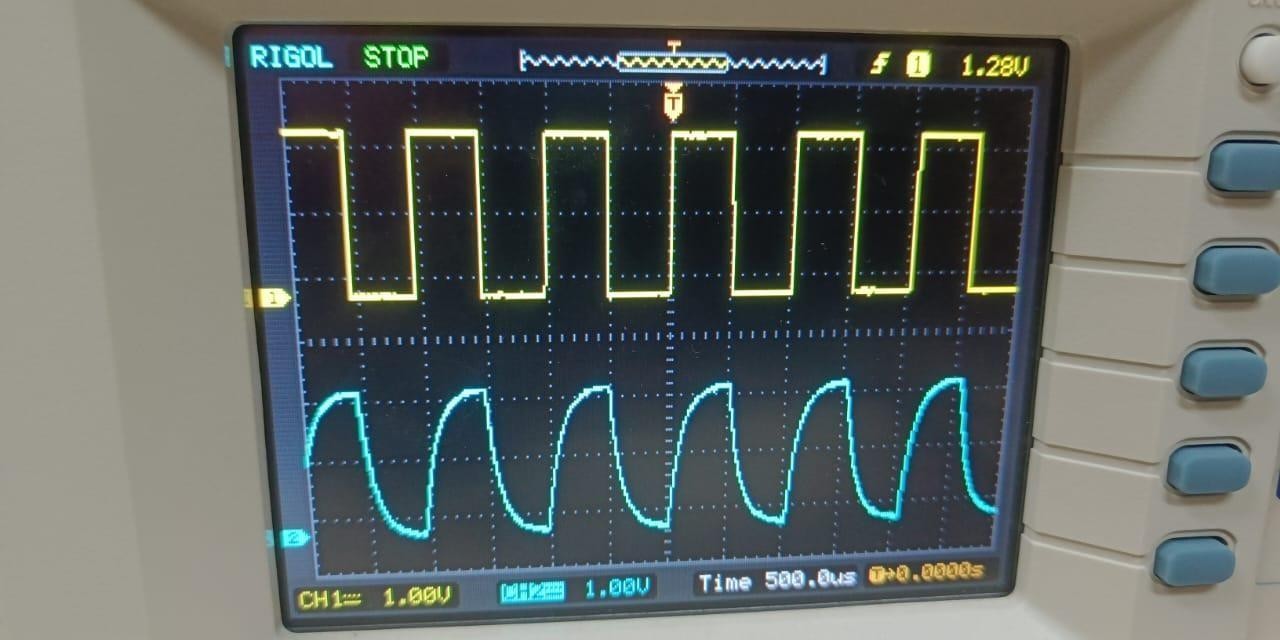
7.3



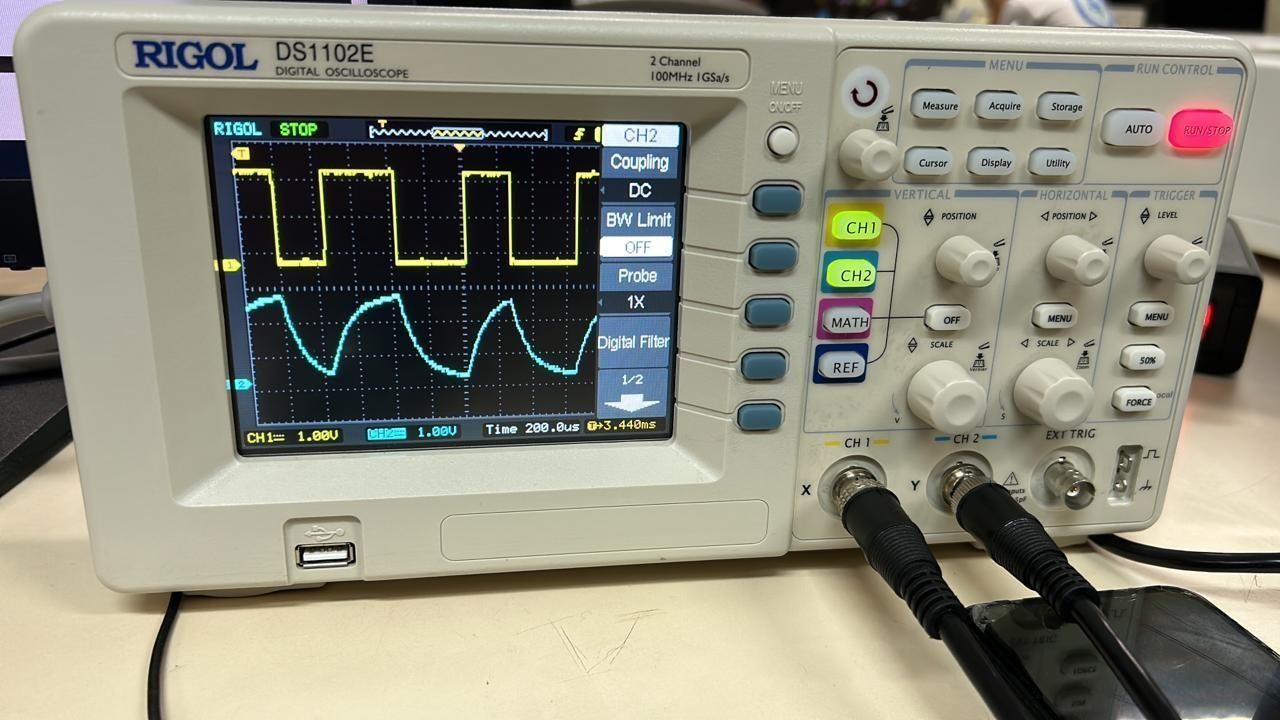
7.4



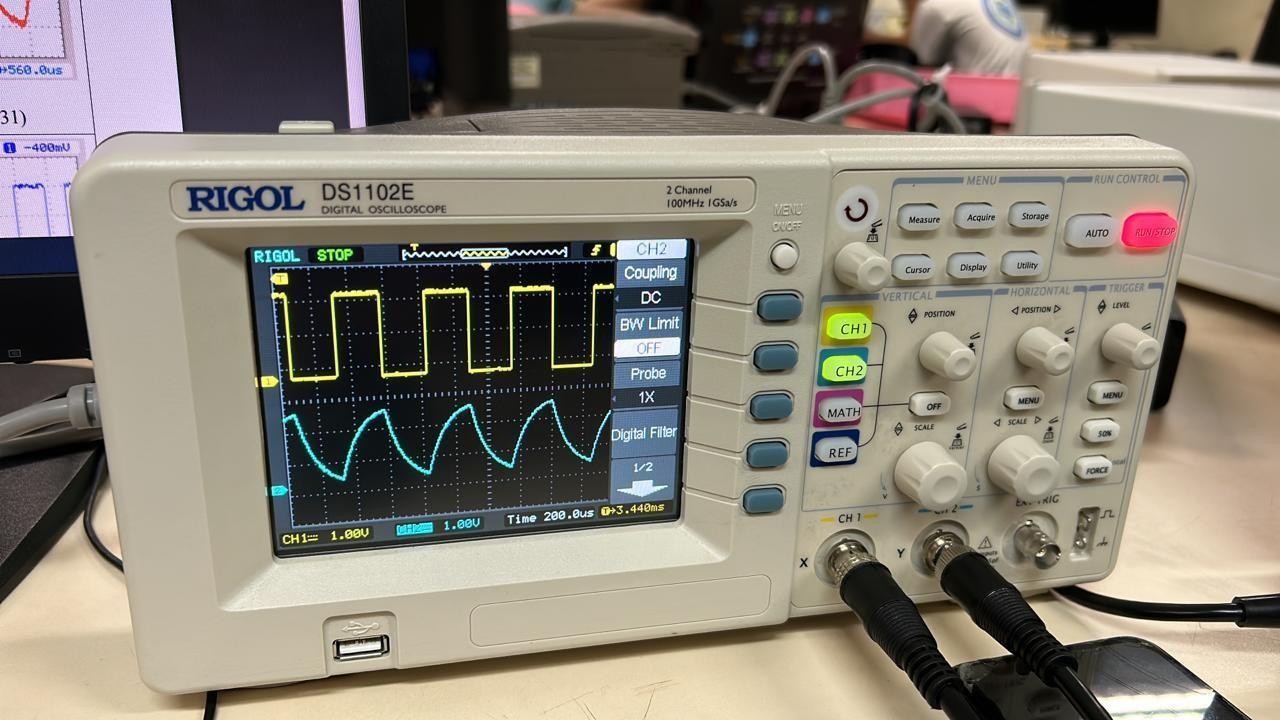
7.5



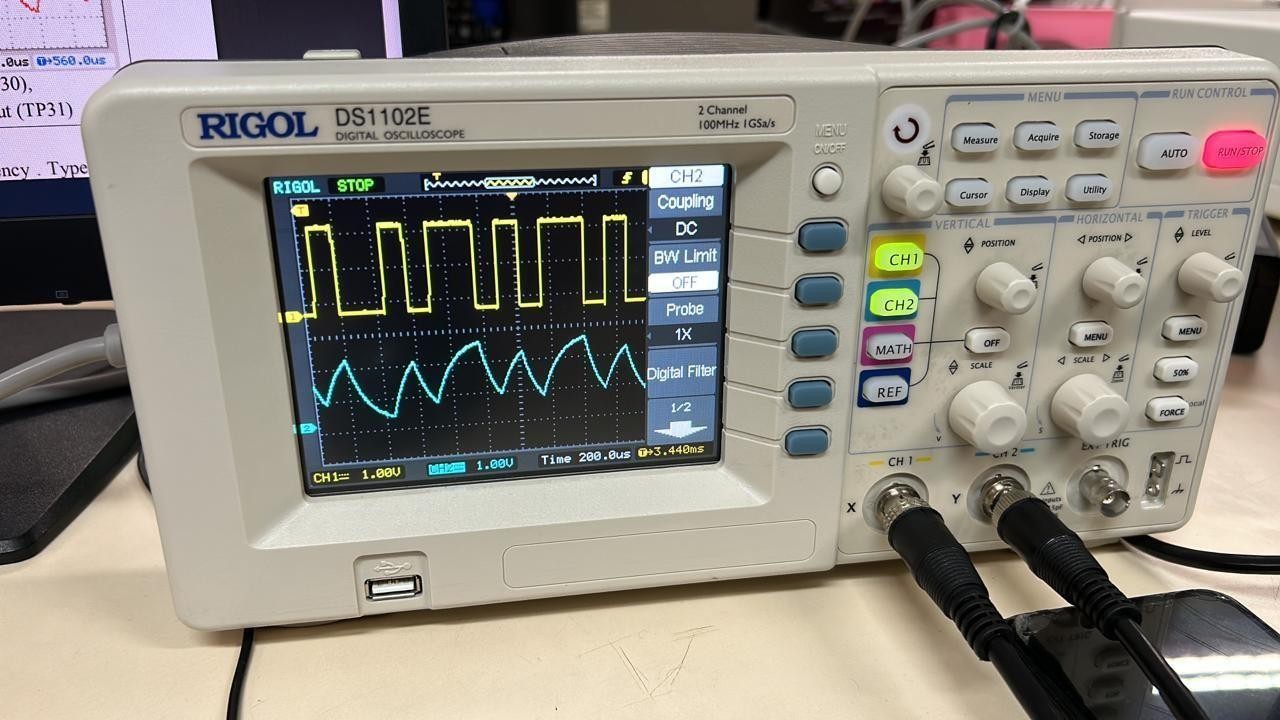
7.6



7.7



7.8



7.9

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

%original sampling rate 500 Hz

% t ime interval of 2 seconds xsig=cos (10\*pi\*t) +cos(16\*pi\*t); Lsig=length(xsig);

ts=0.02;

Nfactor=ts / td;

%new sampl ing rate = 50Hz .

% s end the signal through a 16-leve l uni form quant i z er

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

% rece ive 3 signal s:

% 1. sampled signal s\_out

% 2. sampled and quant ized signal sq\_out

% 3. sampled , quant i zed , and zero- order hold signal sqh\_out

%

% calculate the Fouri er transforms 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 ) );

% Examples of sampl ing and recons truc tion us ing

% a) ideal impulse train through LPF

% b) flat top pul se recons truc tion through LPF

% plot the original signal and the sample signals in t ime

% and frequency domain figure (1) ;

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

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

%set(sfig1a , 'LineWidth' ,2) ; 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])

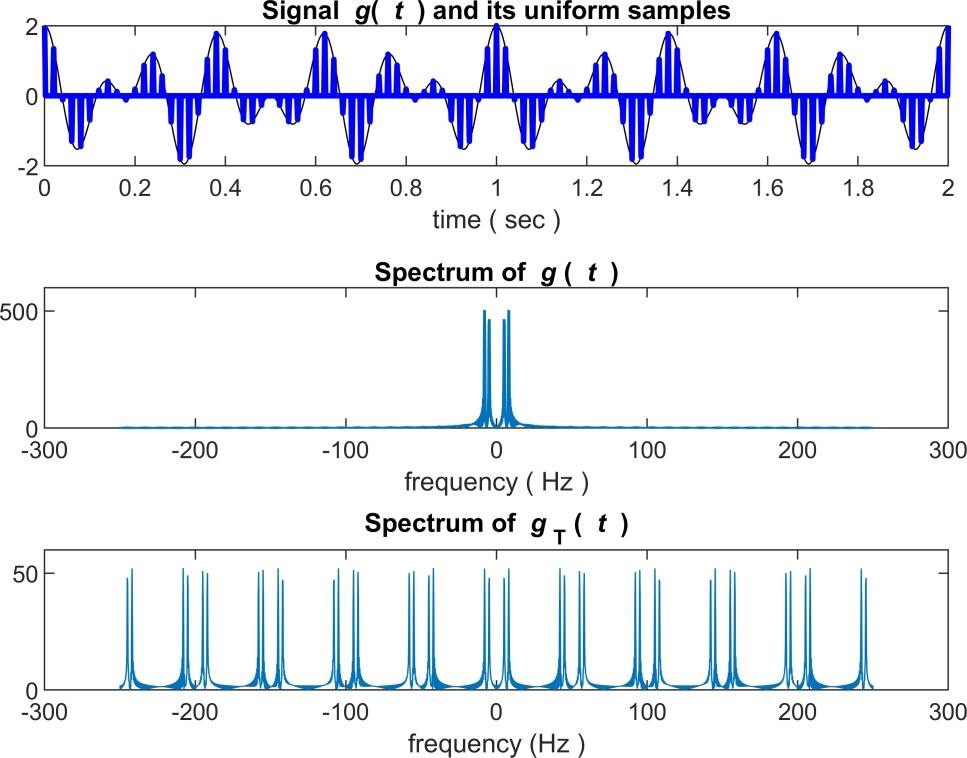
set(sfig1c , 'LineWidth' , 1 ) ; title ( ' Spectrum of {\it g} ( {\it t } ) '); subplot(313) ; sfig1d=plot(Faxis , abs( S\_out ) );

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

% ideal LPF

% Maximum LPF bandwidth equal s to BW= f l oor ( ( Lfft /Nfactor ) / 2 ) ; BW=100 ; %Bandwidth is no larger than 100Hz .

H\_lpf= zeros(1, Lfft); H\_lpf(Lfft/2-BW : Lfft/2+BW-1) = 1 ; %ideal LPF S\_recv=Nfactor\* S\_out .\* H\_lpf ; % ideal fil tering s\_recv=real(ifft(fftshift(S\_recv))) ; % recons tructed £ -domain s\_recv= s\_recv(1:Lsig) ; % reconstructed t-domain

% plot the ideal ly reconstructed signal in t ime

% and frequency domain figure(2)

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

axis ( [ -300 300 0 600] );

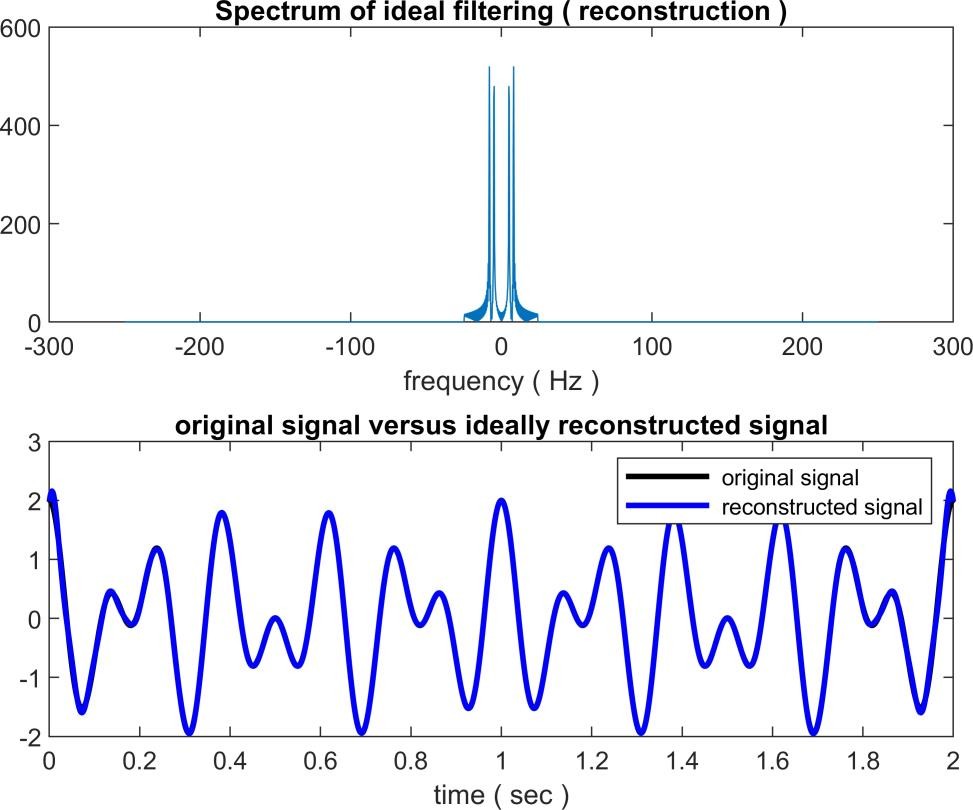
title ( ' Spectrum of ideal filtering ( reconstruction ) ');

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

% plot the ideally reconstructed signal in time

% and frequency domain figure (3)

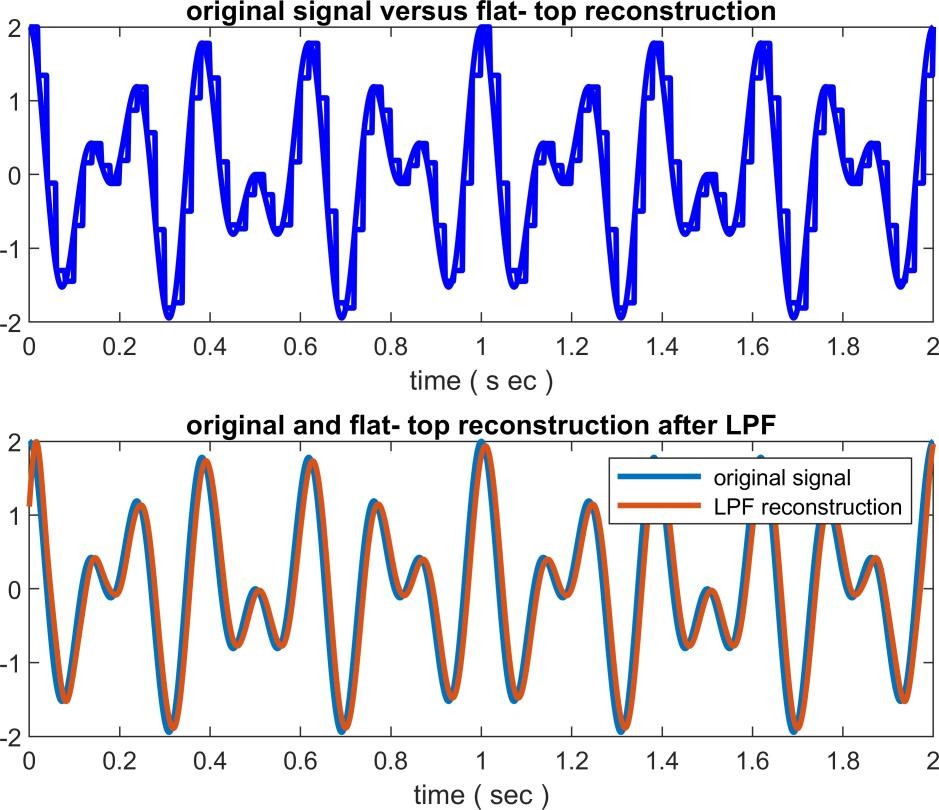
subplot(211); sfig3a=plot(t,xsig , 'b' , t , s\_ni (1:Lsig) , ' b '); xlabel( ' time ( s ec ) ');

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 )

% Usage

% [ q out , Delta , SQNR] =uniquan ( s ig\_in , L )

% L number of uni form quant i zation l evels

% sig\_in - input signal vec tor

% Func tion outputs:

% q out - quant i zed output

% Del ta - quantization interval

% SQNR

sig\_pmax=max( sig\_in ) ;

%actual signal to quant i zation noise ratio

% f inding the pos i t ive peak

sig\_nmax=min( sig\_in ) ; % finding the negative peak Delta= ( sig\_pmax- sig\_nmax ) /L; % quantization interval

q\_level=sig\_nmax+Delta/2 : Delta : sig\_pmax-Delta / 2 ; % de f ine Q-levels L\_sig=length( sig\_in ) ; % find signal l ength

sigp=(sig\_in- sig\_nmax ) / Delta+1/2 ; % convert into 1/2 to L+ l/2 range qindex=round( sigp ) ; % round to 1, 2, ... L levels

qindex=min ( qindex , L ) ; % eleminate L+l as a rare possibility q\_out =q\_level(qindex) ; % use index vec tor to generate output SQNR=20.\*log10(norm(sig\_in)/norm( sig\_in-q\_out )); %actual SQNR value end

% ( sampandquant .m)

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

% Usage

4

% [ s\_out , sq\_out , sqh\_out , Delta , SQNR] =sampandquant ( s ig\_in , L , td , f s )

% L number of uni form quantization l evels

% sig\_in - input signal vec tor

% td original signal s ampl ing period of sig\_in

% ts new sampl ing period

% NOTE : td\* fs must be a pos itive integer ;

% Func tion output s:

%s out - sampled output

%6. 9 MATLAB Exercises 315

%

%

%

%sq\_out - sample-and-quantized output

%sqh\_out - sample , quantize , and ho ld output

%

%

%Delta - quant ization interval

%SQNR - ac tual signal to quanti zation noise ratio 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

end

5