Credit Hours System

ELCN351-FALL-2021

Industrial electronics

Cairo University

Faculty of Engineering

Assignment - Report

Submitted to: Dr. Ahmed Hussien

Student name: Hosam Motaz Hosam

Student ID: 1180534

Submission Date: 22.12.2021

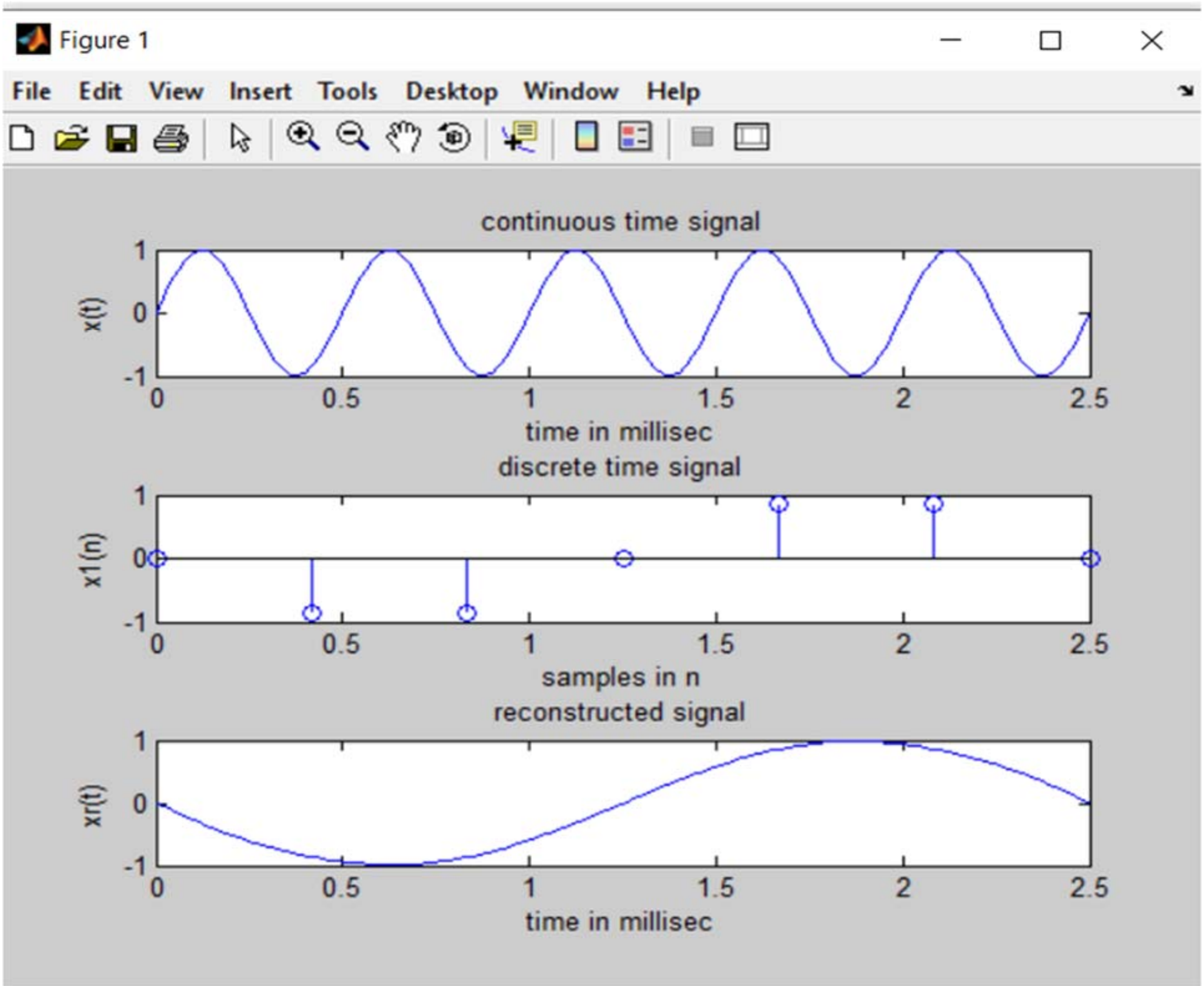
ASSIGNMENT-3

PART-A

We will apply sampling condition if only one single frequency was sent

If Fs less than 2Fm aliasing occur and we want be able to reconstruct original signal from observed sample. The reconstructed signal is a new signal of low frequency appears at fs-fm (information is lost)

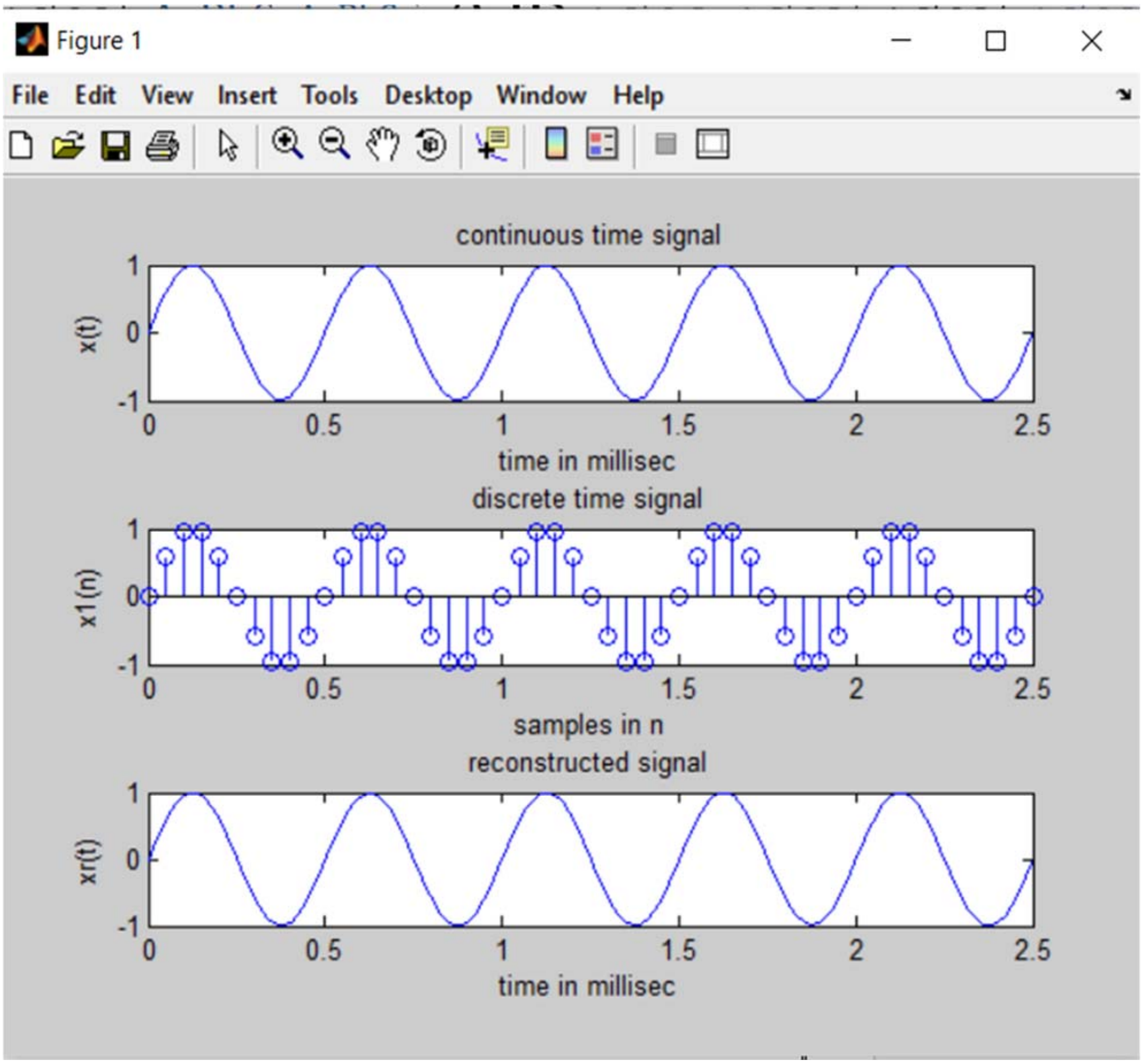
If Fm=2khz and Fs=1.2Fm therefore aliasing occurs resulting in reconstructed signal of frequency=0.4khz



In order to reconstruct original shape of signal therefore Fs>=2Fm (the bigger Fs the better to be able to reconstruct the shape accurately) and so no information is lost during sampling

For Fm=2khz if Fs=10Fm therefore original shape is reconstructed and no loss of information

As shown blow:



Here is the code for this part

```
clc
clear
%single frequency
%assume frequency =2Khz therefore t=0.5ms
%5aleeh step very small 34an signal 7lwa
fm=2;%2khz
dt=1/50;
tm=1/fm;
t=0:dt:5*tm;%nrsmha mrtain period b dt
x=sin(2*pi*fm*t);
subplot(5,1,1)%3 rows 1 column and first location
plot(t,x)
xlabel('time in millisec')
ylabel('x(t)')
title('continuous time signal')

y = fft(x);
ft = 50;
n = length(x);
fshift = (-n/2:n/2-1)*(ft/n);
subplot(5,1,2)
yshift = fftshift(y);
plot(fshift,abs(yshift))%lma t reconstruct use lpf of amplitude 1/50 therefore dont change
amplitude
xlabel('Frequency (KHz)')
ylabel('Magnitude')


%sampling means we will discretize x-axis
fs=10*fm;
ts=1/fs;
n1=0:ts:5*tm;%ta5od sample kol ts
x1=sin(2*pi*fm*n1);
%we will use stem command to show the sampled signal
subplot(5,1,3)
stem(n1,x1);
xlabel('samples in n')
ylabel('x1(n)')
title('discrete time signal');



%we would like to reconstruct the signal
%first reconstruct time vector then we will reconstruct continuous signal
%from the sampled values
tr=linspace(0,max(n1),(max(n1)/dt));%n1 is discrete time vector and line space command is used
to create the linearly space vector from 0 to maximum of na
%we will use interpolation command and spline method to reconstruct
%the,,,,tr is reconstructed time vector
%signal as shown below
xr=interp1(n1,x1,tr,'spline');%or linear
subplot(5,1,4);
plot(tr,xr)
xlabel('time in millisec')
ylabel('xr(t)')
title('reconstructed signal');

%we can also check using fourier transform
yrec = fft(xr);
ftrec = 50;
nrec = length(xr);
fshift = (-nrec/2:nrec/2-1)*(ftrec/nrec);
subplot(5,1,5)
yshift = fftshift(yrec);
plot(fshift,abs(yshift))%lma t reconstruct use lpf of amplitude 1/50 therefore dont change
amplitude
xlabel('Frequency (KHz)')
ylabel('Magnitude')
```

PART-B

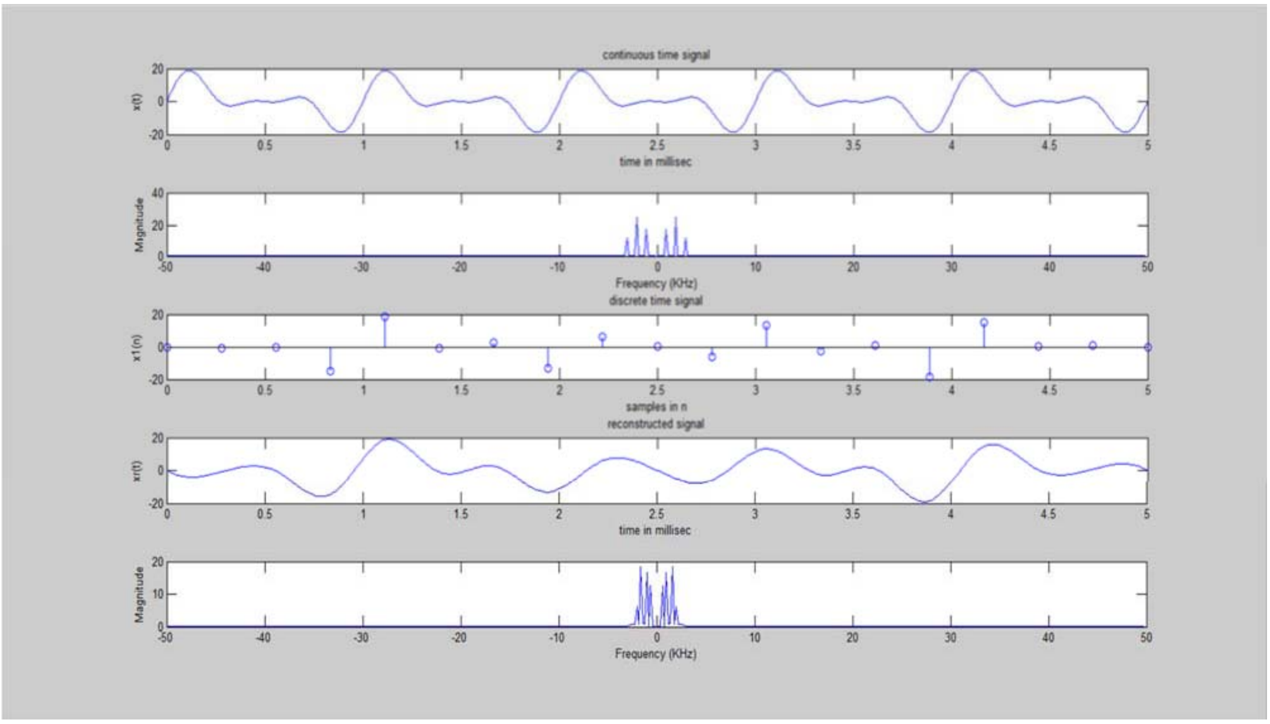
We will apply sampling condition if more than one signal with multiple frequencies was sent

If for example we have 3 frequencies f1,f2,f3 for 3 different signals therefore in order to be able to reconstruct shape and don't lose information it is requires that the sampling frequency Fs is at least twice the maximum frequency of either f1or f2 or f3

Therefore Fs>=2max(f1,f2,f3)

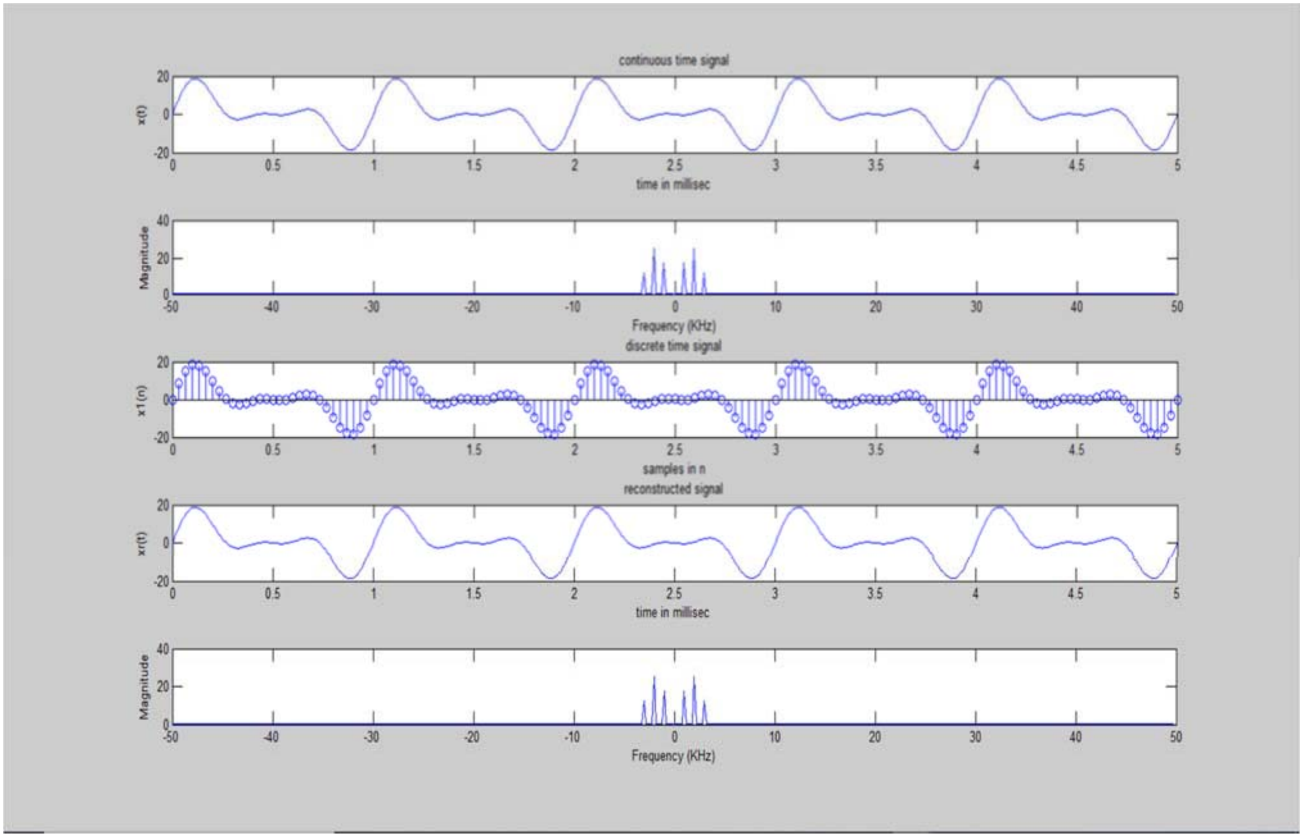
If this condition is not satisfied aliasing occur ,information is lost and we won't be able to reconstruct the shape of the signal

For eg: if f1=2khz,f2=1khz,f3=3khz and Fs=3.6khz(less than 6khz) therefore nyquist condition is not fulfilled



Now assume Fs=30khz:-nyquist condition is fulfilled therefore the shape of signal can be reconstructed

As shown below



Here is the code for this part

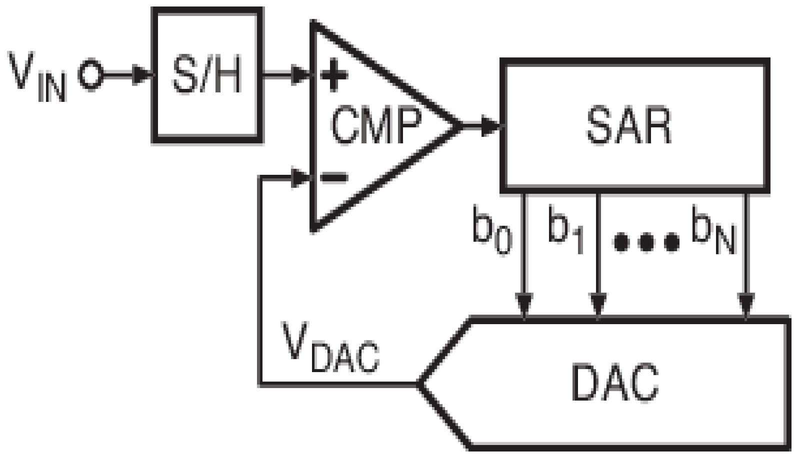
```
clc
clear
%multiple frequency
%assume signals with different frequencies =2Khz,1Khz and 3Khz were sent
%5aleeh step very small 34an signal 7lwa
f1=2;%2khz
f2=1;%1khz
f3=3;%3khz
dt=1/100;
t=0:dt:5/f2;%nrsmha mrtain period 1/f2
y1=10*sin(2*pi*f1*t);
y2=7*sin(2*pi*f2*t);
y3=5*sin(2*pi*f3*t);
x=y1+y2+y3;
subplot(5,1,1)%3 rows 1 column and first location
plot(t,x)
xlabel('time in millisec')
ylabel('x(t)')
```

```
title('continuous time signal')
f=[f1,f2,f3];
fm=max(f);


%fourier transform of the transmitted signal
y = fft(x);
ft = 100;
n = length(x);
fshift = (-n/2:n/2-1)*(ft/n);
subplot(5,1,2)
yshift = fftshift(y);
plot(fshift,abs(yshift)/ft)
xlabel('Frequency (KHz)')
ylabel('Magnitude')



%sampling means we will discretize x-axis
fs=10*fm;
ts=1/fs;
n1=0:ts:5/f2;%ta5od sample kol ts
x1=10*sin(2*pi*f1*n1)+7*sin(2*pi*f2*n1)+5*sin(2*pi*f3*n1);
%we will use stem command to show the sampled signal
subplot(5,1,3)
stem(n1,x1);
xlabel('samples in n')
ylabel('x1(n)')
title('discrete time signal');


%we would like to reconstruct the signal
%first reconstruct time vector then we will reconstruct continuous signal
%from the sampled values
tr=linspace(0,max(n1),(max(n1)/dt));%n1 is discrete time vector and line
space command is used to create the linearly space vector from 0 to maximum
of na
%we will use interpolation command and spline method to reconstruct
%the,,,,tr is reconstructed time vector
%signal as shown below
xr=interp1(n1,x1,tr,'spline');%we can also use linear but spline is better
subplot(5,1,4);
plot(tr,xr)
xlabel('time in millisec')
ylabel('xr(t)')
title('reconstructed signal');


%fourier transform to observe the difference
yrec = fft(xr);
ftrec = 100;
nrec = length(xr);
fshiftt = (-nrec/2:nrec/2-1)*(ftrec/nrec);
subplot(5,1,5)
yshiftt = fftshift(yrec);
plot(fshiftt,abs(yshiftt)/ftrec)
xlabel('Frequency (KHz)')
ylabel('Magnitude')
```

ASSIGNMENT-1

Required to design the circuit for SAR type ADC

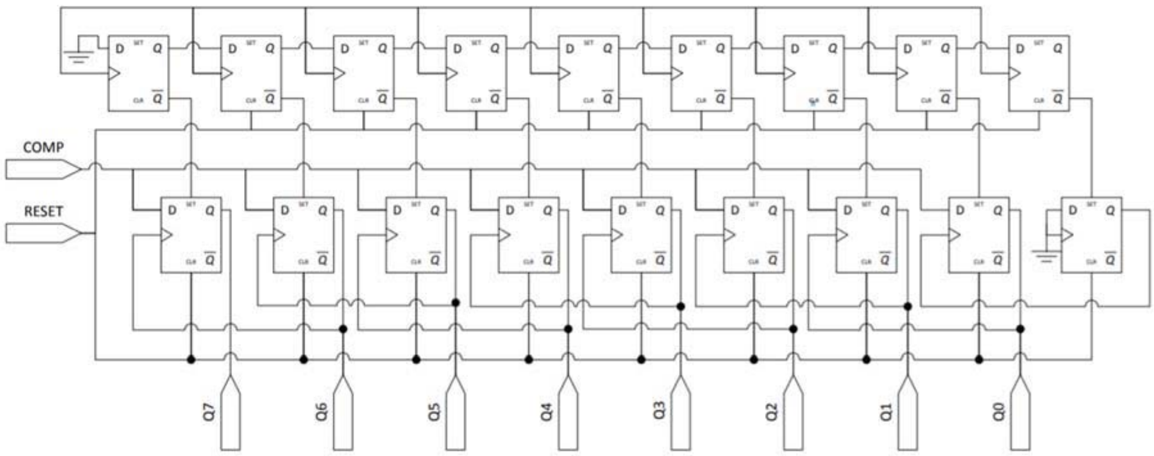
This is the full block diagram of the SAR-type ADC and our attempt is to design the control logic (of SAR) of this type of ADC (with the help of counter type ADC)



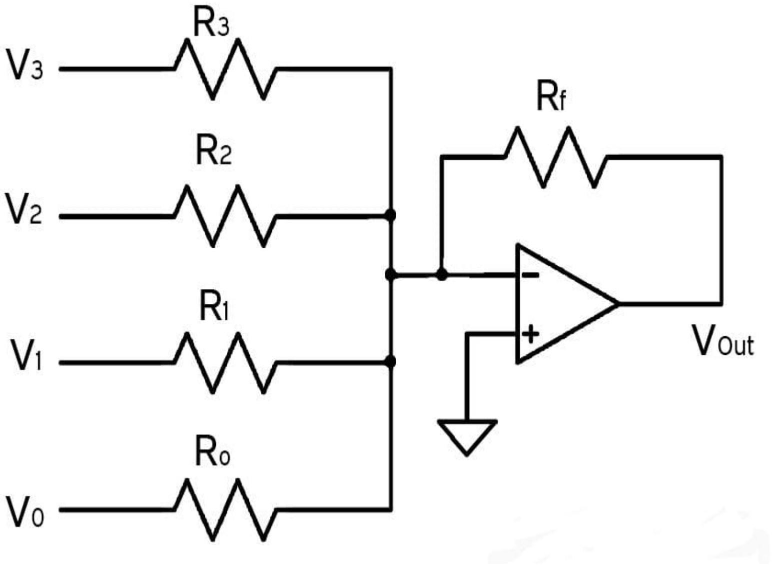
Here is the what SAR is made of it is made of many registers if we will construct 4 bit SAR we will require 10 d flip flops(2n+2)

The top 5 flip flops act as a shift register, while the bottom 5 flip flops act as code register which depends upon the top register where the output of the top registers enables the bottom register and updates its value according to the output of the comparator.

The below figure shows an example of 8 bit SAR control logic (we will implement similar design but for 4 bit)



We will also require 4 bit DAC in order to compare the result with the input. This is the design used for our DAC



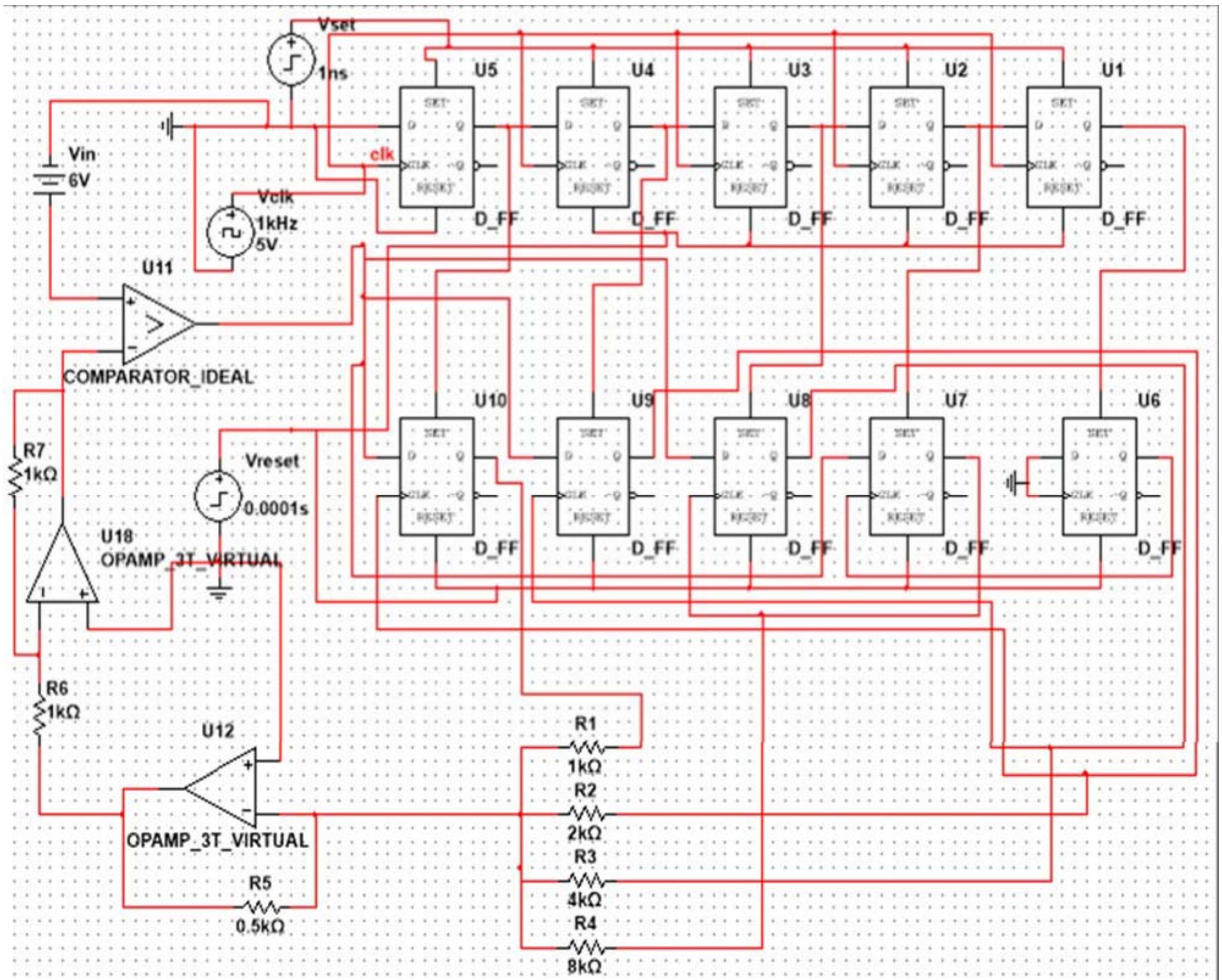
$$V_{out} = -\left\{ \frac{R_f}{R_0} V_0 + \frac{R_f}{R_1} V_1 + \frac{R_f}{R_2} V_2 + \frac{R_f}{R_3} V_3 \right\}$$

Since the result of Vout is negative we will need a simple inverter op-amp . we would like to make use of this DAC so we will justify the values of the resistor so as to obtain a maximum Vout of 4.6875 with an error of 0.3125

R0=1K,R1=2K,R2=4K,R3=8K WHILE RF=0.5K

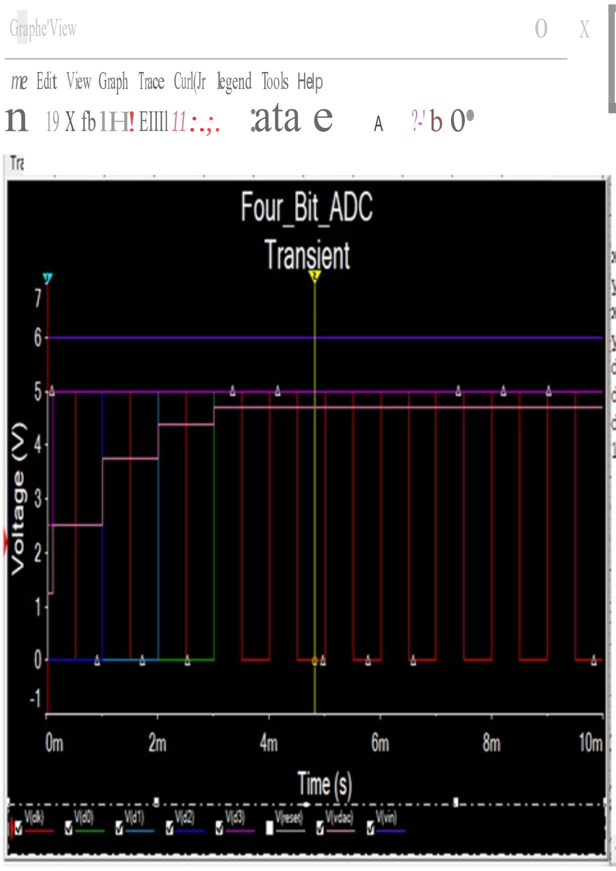
(we can also obtain maximum Vout of 9.375 simply by letting RF=1K instead of 0.5K)

And here is our design



Lets start by assuming Vin=6V

Therefore all the 4 bits will be '1' and the output of DAC is 4.687SV



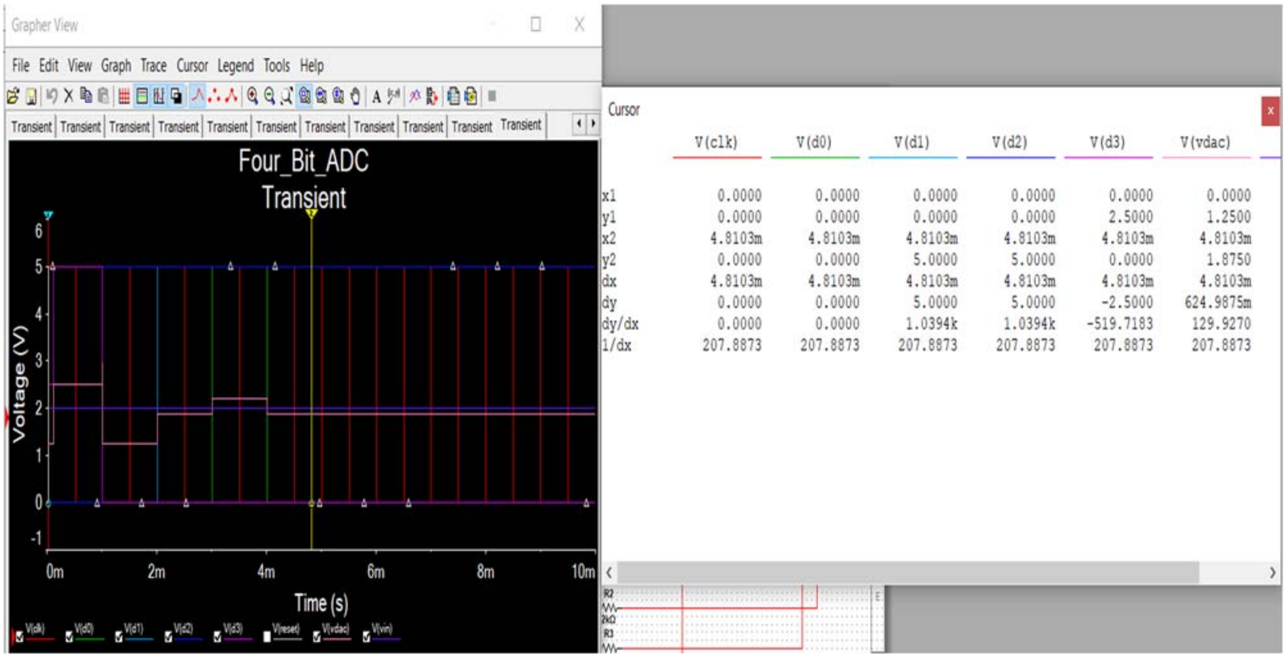
	V(clk)	V(dO)	V(d1)	V(d2)	V(d3)	V(vdac)
x1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
y1	0.0000	0.0000	0.0000	0.0000	2.5000	1.2500
x2	4.8103:!	4.8103m	4.81033	4.8103m	4.8103:!	4.8103m
y2	0.0000	5.0000	5.0000	5.0000	5.0000	4.6814
dx	4.8103::1	4.8103m	4.8103=t	4.8103m	4.8103=t	4.8103m
dy	0.0000	5.0000	5.0000	5.0000	2.5000	3.4374
dy/dx	0.0000	1.0394k	1.039n	1.0394k	519.7183	714.5983
1/dx	201.8813	201.8813	201.8813	201.8813	201.8813	201.8813

03 is the MSB while DO is the LSB

Lets try another example by assuming Vin=2V:

Therefore the result of the DAC is 1.875 Volt

Where the result of the 4 bit is 0110 (level 6)



In this type of ADC the quantization error is VFS/(2^n-1)
Vin=2v
Illustration:
In the first clock cycle the vin is compared to 1000 which is 2.5v and in the second clock cycle if Vin is less than 2.5v it is compared to 0100 which is 1.25v, in the third clock cycle if the Vin is greater than 1.25v it is compared to 0110 which is 1.875v .In the fourth cycle if the Vin is larger than 1.875v therefore it is compared to 0111 which is 2.1875v and since we are using this type of ADC with error of complete level therefore 2v will appear to the down level so it appears as 1.875v
(Each clock in this type of ADC process one bit)
Tc (conversion time) worst case is equal to n*Tclk
Where n is the number of bit
In our case maximum conversion time is 4Tclk