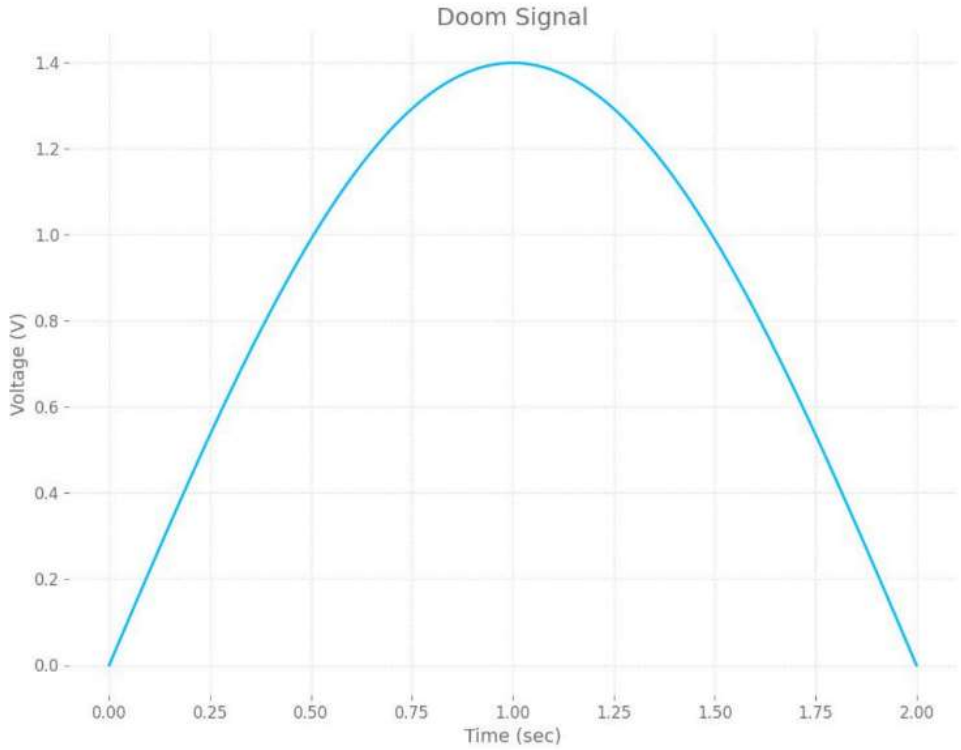


if we need to convert this analog to digital signal what is the digital sequence if we use ADC with :

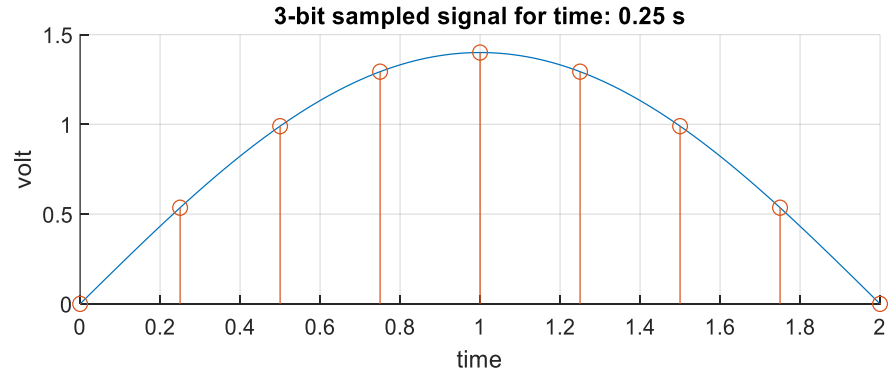
- 3-bit encoder with Sampling Time = 0.25sec,
- 3-bit encoder with Sampling Time = 0.5sec,
- 3-bit encoder with Sampling Time = 1sec,
- 2-bit encoder with Sampling Time = 0.25sec,

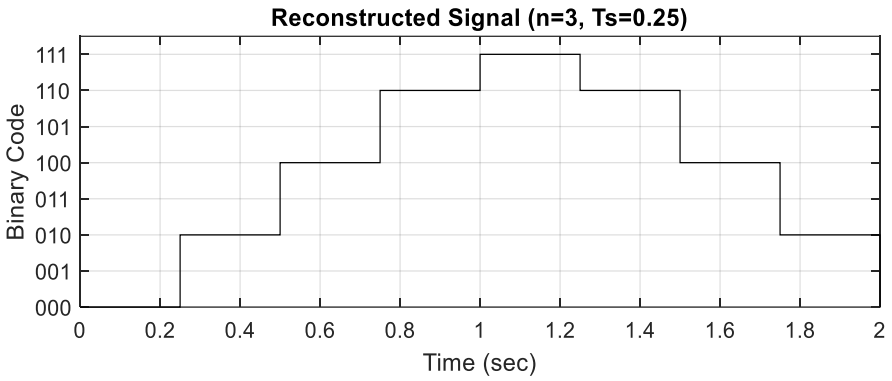
at each point draw the discrete signal also (the step after time sampling and quantization)

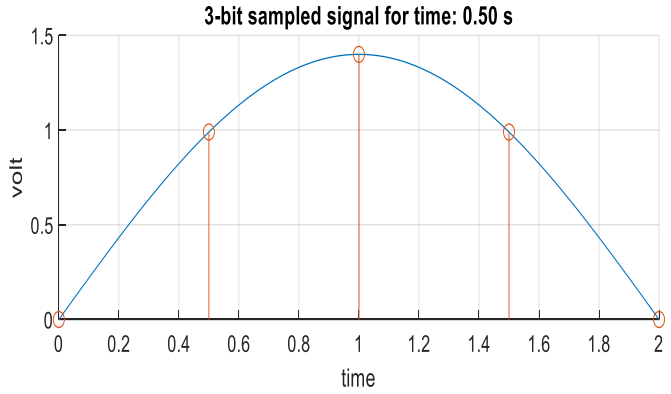
what is your conclusion from this problem

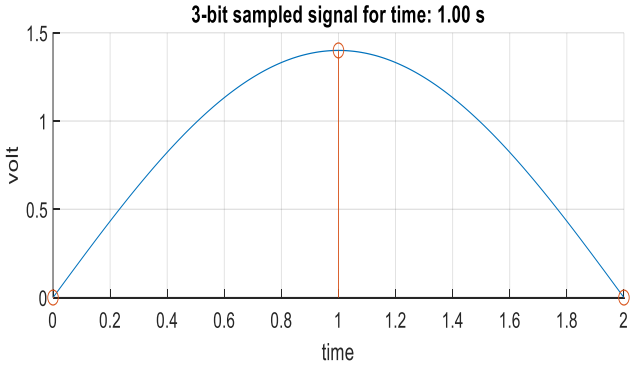
Solution:

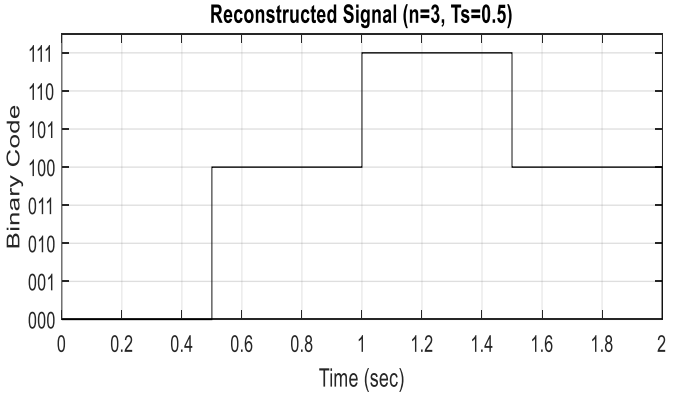
The following graphs show the 3-bit quantization of a half sin wave at different sampling time :

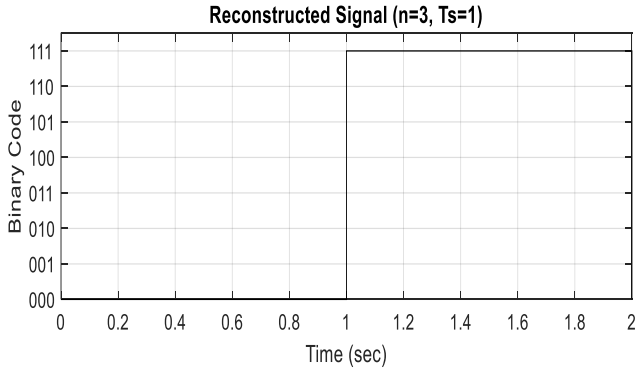










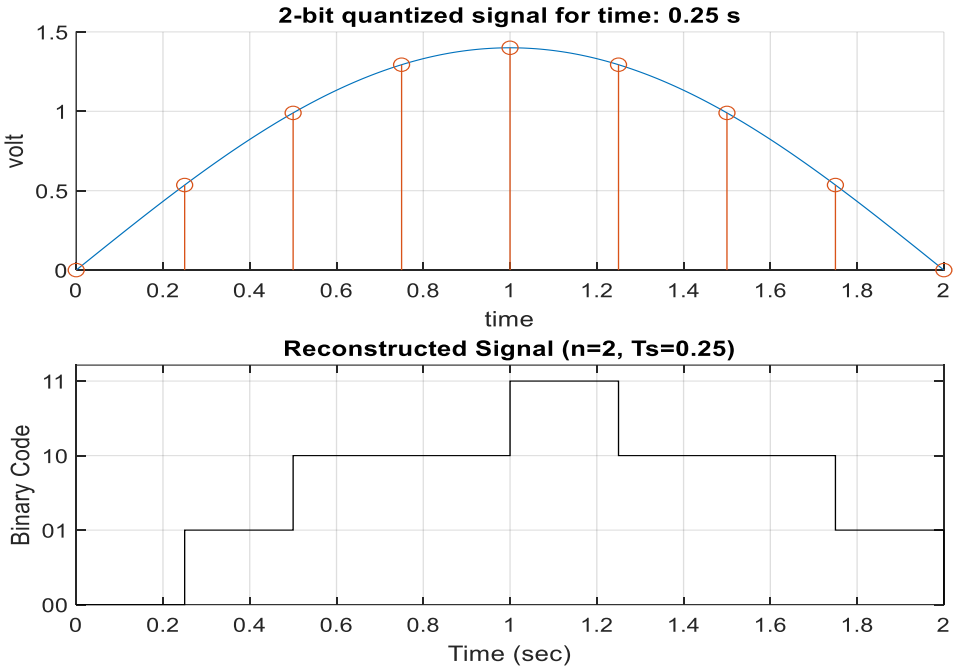


Code used for 3bit generation:

```
%% generating an analog sin wave
t=0:0.01:2;
amp=1.4;
v=amp*sin(pi*t/2);
```

```
%% 3 bit encoder
samplings=[0.25,0.5,1];
for i=1:length(samplings)
    t_sample=0:samplings(i):2;  %time sampling with 0.25,0.5,1 step
    v_sampled=zeros(1,length(t_sample(i)));  % sampled voltage empty array
    quantization=zeros(1,length(t_sample(i))); %qauntized data empty array
    q = amp / (2^3 - 1);
    for j =1:length(t_sample)
    v_sampled(j)=amp*sin(pi*t_sample(j)/2);
    quant=(v_sampled(j)*(2^3-1)/amp);
    quantization(j)=quant*(amp/(2^3-1));
    a = fix(v_sampled / (amp / (2^3 - 1)));  % rounds the data
    yq=a*(amp / (2^3 - 1));
    end
    %plotting
    figure;
    subplot(2,1,1)
    hold on
    plot(t,v)
    stem(t_sample,quantization)
     grid on
    xlabel("time");
    ylabel("volt");
    title(sprintf('3-bit sampled signal for time: %.2f s',samplings(i)))
    hold off
    %plotting the data in binary steps
    subplot(2, 2, 3:4);
   stairs(t_sample, yq, 'black');
   title(['Reconstructed Signal (n=', num2str(3), ', Ts=', num2str(samplings(i)), ')'])
   xlabel('Time (sec)');
   ylabel('Binary Code');
   yticks((0:2^3-1) * q);
   yticklabels(dec2bin(0:2^3-1, 3));
   grid on
end
hold off
```

The following graph shows the 2-bit quantization of a half sin wave at 0.25 sampling time:



The code used:

```
%% 2 bit encoder
t_sample=0:0.25:2;
v_sampled=zeros(1,length(t_sample));
quantization=zeros(1,length(t_sample));
q = amp / (2^2 - 1);
for j =1:length(t_sample)
    v_sampled(j)=amp*sin(pi*t_sample(j)/2);
    quantization(j)=(v_sampled(j)*(2^2-1)/amp)*(amp/(2^2-1));
    a = fix(v_sampled / q);
    yq=a*q;
end
figure;
subplot(2,1,1)
hold on
 plot(t,v)
 grid on
 stem(t_sample,quantization)
 xlabel("time");
 ylabel("volt");
title('2-bit quantized signal for time: 0.25 s')
hold off
subplot(2, 2, 3:4);
stairs(t_sample, yq, 'black');
title(['Reconstructed Signal (n=', num2str(2), ', Ts=', num2str(0.25), ')']);
xlabel('Time (sec)');
ylabel('Binary Code');
yticks((0:2^2-1) * q);
yticklabels(dec2bin(0:2^2-1, 2));
grid on
```

Conclusion:

1. **Higher Sampling Rates**: A shorter sampling time represents more details of the signal but requires more data points,(more memory).
2. Quantization error is reduced by increasing the resolution.
3. In the 0.1 s sampling time it didn't represent the data which should be solved by using the Nyquist theorem where (sampled freq>2 data freq) .
4. By increasing the number of bits more data can be represented over the same time where in the 2bit there are about 4 data points represented by 10 only although they are at different voltages

Example Plots

1. **3-bit Encoder with 0.25 sec Sampling Time**:
    - Sample points: [0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2]
    - Quantized values: [000, 011, 100, 110, 111, 110, 100, 011, 000]
2. **2-bit Encoder with 0.25 sec Sampling Time**:
    - Sample points: [0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2]
    - Quantized values: [00, 01, 10, 10, 11, 10, 10, 01, 00]