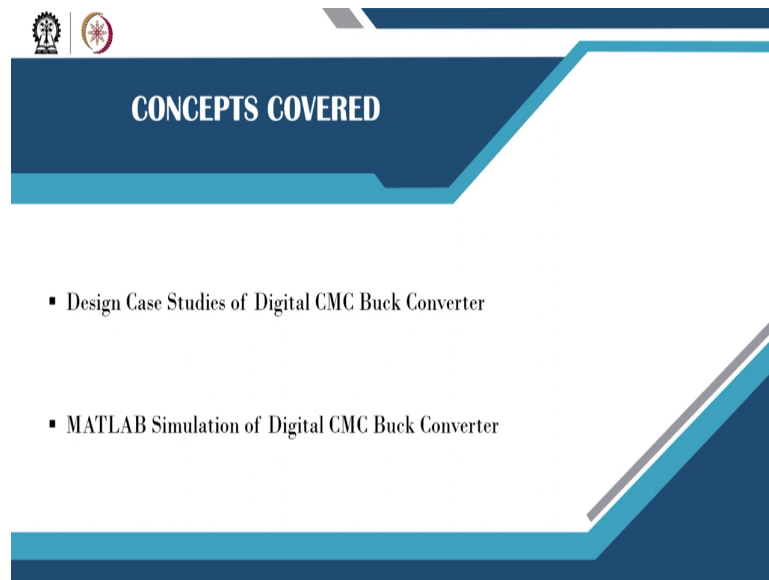


Digital Control in Switched Mode Power Converters and FPGA-based Prototyping
Prof. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Module - 05
Frequency and Time Domain Digital Control Design Approaches
Lecture - 46
Design Case Study and MATLAB Simulation of Digital Current Mode Control

Welcome, in this lecture we are going to talk about Digital Current Mode Control Design and MATLAB Case Study in a buck converter.

(Refer Slide Time: 00:33)



The slide features a dark blue header with the text "CONCEPTS COVERED" in white. Below the header, there are two bullet points in white text on a dark blue background. The slide is decorated with geometric shapes in shades of blue and grey.

- Design Case Studies of Digital CMC Buck Converter
- MATLAB Simulation of Digital CMC Buck Converter

So, we will talk about a design case study of the digital current mode control buck converter and then MATLAB simulation of digital current mode control.

(Refer Slide Time: 00:41)

Buck Converter Current Mode Control

NPTEL Online Certification Courses
IIT Kharagpur

So, here we want to recapitulate our analog current mode control structure and which we have discussed multiple times. So, this is the structure and this is you are considering peak current mode control fix frequency, that is the under trailing edge modulation.

(Refer Slide Time: 00:56)

Current Mode Control : Primary Loop Shaping Objectives

$$G_{vc}(s) = \frac{R \left(1 + \frac{s}{\omega_{ESR}} \right)}{\left(1 + \frac{s}{\omega_p} \right)}$$

where $\omega_{ESR} = \frac{1}{r_c C}$; $\omega_p = \frac{1}{(R + r_c)C}$

$$\Rightarrow K_{loop}(s) = \frac{R \left(1 + \frac{s}{\omega_{ESR}} \right)}{\left(1 + \frac{s}{\omega_p} \right)} \times G_c$$

Loop Gain of Practical Buck Converter

[For details, refer to Lecture~38, NPTEL "Control and Tuning Methods ..." course [\(Link\)](#)]

NPTEL Online Certification Courses
IIT Kharagpur

Then what is the primary loop-shaping objective in analog current mode control? So, we consider the control to output transfer function and this we have discussed in lecture number 36, 37; I think 38, yeah lecture number 38 in our earlier NPTEL course for current mode

control design in a buck converter. So, where yeah, so this is lecture number 38 in our NPTEL course.

So, where we know that current mode control can be now buck converter can be approximated, control to output transformation can be approximated by a first-order model, and then by stable pole-zero cancellation, we can design a compensator which.

(Refer Slide Time: 01:41)

Ideal Buck Converter CMC Controller Tuning : Summary

$$G_c = K_p + \frac{K_i}{s} = \frac{K_i \left(1 + \frac{s}{\omega_{cz}} \right)}{s}$$

where $\omega_{cz} = \frac{K_i}{K_p}$

Stable pole/zero cancellation

$$\omega_{cz} = \omega_p \Rightarrow K_p = K_i \times RC$$

Loop Gain of Practical Buck Converter

$$K_{loop}(s) = \frac{RK_i}{s}$$

[For details, refer to Lecture-44, NPTEL "Digital Control of Switched Mode ..." course]

NPTEL Online Certification Courses
IIT Kharagpur

So, if you take a PI controller, then this is the case when we are ignoring the effect due to ESR; assuming that ESR is very very low, which is typically the case for a practical converter. If ESR is here, we have shown that we need a type II compensator, where and if the ESR is negligible, then a PI controller is enough. So, what is between difference between PI and type II we have discussed; the type II compensator is a generalized version of a PI controller, where we can have an extra pole and if we keep that extra pole far at the left-hand side, then it can be approximated like a PI controller.

So, if we consider an ideal buck converter; that means a very very low ESR, then we can consider a PI controller. And what is the design objective? The omega c z; that means the PI controller can be represented by this form, where the 0 has one 0 and one pole at the origin, the zero is K i by K p. And then if we take the loop transfer function by stable pole-zero cancellation; then we will get the loop transfer function is a first-order system.

And this requires that controller zero should cancel the pole and it turns out the proportional gain should be K_i times RC ; because we are assuming the ESR to be negligible. So, it is R is the load resistance, and since C is the output capacitor, this we have discussed in lecture number 44.

(Refer Slide Time: 03:15)

Design Digital CMC based on Gain Crossover Frequency

Step 1: Select gain crossover frequency ω_c by setting $\omega_c = \frac{2\pi f_{sw}}{8}$

Step 2: Compute phase margin (PM) $PM = 90^\circ - \omega_c \tau_d$

Step 3: For given τ_d , verify whether PM meets the requirement, typically $PM = 60^\circ$

Step 4: If not, go to step~1, reduce ω_c and repeat the process till PM is met

Step 5: If step~3 is passed, find $K_i = \frac{\omega_c}{R}$

[For details, refer to [Lecture~44](#), NPTEL "Digital Control of Switched Mode ..." course]

NPTEL Online Certification Courses
IIT Kharagpur

Now, if we want to get the digital controller, then how to design it? So, we will start with the crossover frequency of one-eighth of the switching frequency and we know that in traditional current mode control by stable pole-zero cancellation, we want can achieve some of the loop transfer function some gain by s ; that means we will get 90 degree phase margin.

But due to the delay in the digital controller, the phase margin will be lower than the analog controller and that will be 90 degree minus ω_c times τ_d , where ω_c is the cross-over frequency. So, we suppose our desired phase margin is 60 degree and if you try to push the cross-over frequency and the delay is large. So, this term can be significant.

So, we need to check that for given τ_d what we need to meet a certain 60 degree phase margin, so that will pose a limit on your cross-over frequency; you cannot have a very large crossover frequency, due to the phase lag contribution which may reduce the effective phase margin.

Now, another aspect we cannot increase cross over frequency too high; because we know that if we try to cross in current mode control, the crossover frequency is beyond one eight or so,

then our model validation is an issue. So, here the small signal-based design may not be valid beyond that. So, keeping that two facts in mind, our desired objective is to achieve 60 degree phase margin without violating the model validation problem.

So, if the phase margin is lower than 60 degree, then may we have to reduce further this omega c and so that it passes this third requirement. And if it is passed, then we will go to the last step that our controller gain integral is simply cross-over frequency by R.

(Refer Slide Time: 05:08)

Convert Analog PI to Digital PI Controller – Backward Difference


$$G_c = \frac{K_i \left(1 + \frac{K_p s}{K_i} \right)}{s}$$

$\omega_c = \frac{2\pi f_{sw}}{8}$

$$K_p = K_i \times RC \quad K_i = \frac{\omega_c}{R}$$

$$K_{pd} = K_p \quad K_{id} = K_i T_s$$

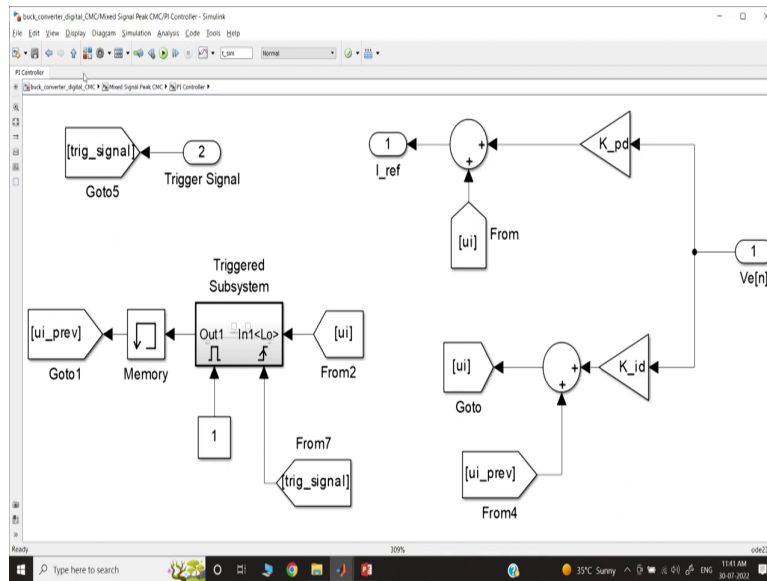
[For details, refer to [Lecture~44, NPTEL "Digital Control of Switched Mode ..."](#) course]



Once we get it, that means we can get the PI controller gain; so that means this is our analog PI controller and analog PI controller K p value we know, it is K i time RC and K i equal to cross over frequency by R and whatever the cross over frequency select, we select typically cross over frequency, we will take 2 pi the switching frequency by roughly you know 8 or so.

Because in current mode control, you can get somewhat higher than the voltage mode control and that divided by R is the K i. Once we get the continuous time proportional and continuous time integral gain; then the derivative digital PI K p gain will remain the same, but the digital integral gain will be the analog integral gain multiplied by the sampling time. So, we have discussed lecture number 44 in this course. Now, we want to see some MALAB case studies.

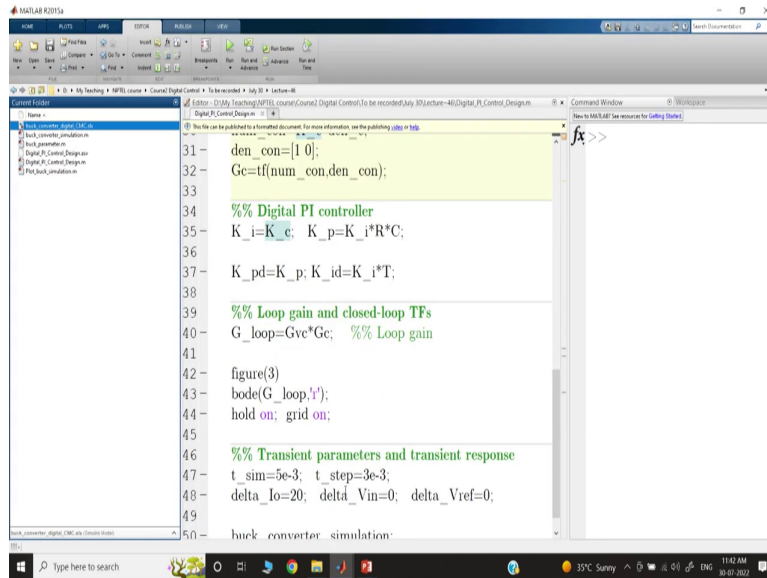
(Refer Slide Time: 06:24)



If you go inside, we have also discussed that mix signal current mode control with a custom PI controller block with a clock-synchronized operation which also I discussed in the third-week lecture. Then we need to set the right value of the proportional gain and the discrete-time integral gain. So, then there is another option is the load current feed-forward; we may or may not use and I will take both case studies.

So, initially, we will take the feed-forward load feed forward to be 0, then it will be the traditional current mode control peak current mode control; then if we consider k_{ff} to be 1, there is a perfect load feed-forward and we want to see what is the response. So, let us go back to the design methodology.

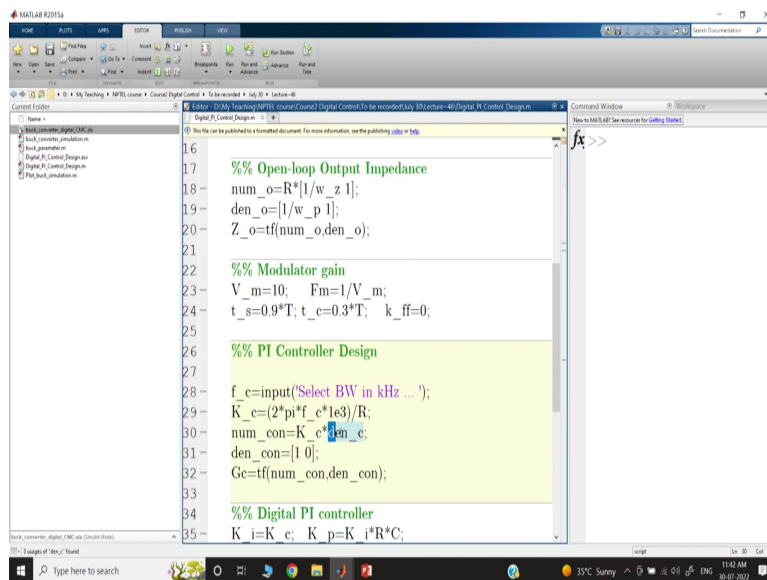
(Refer Slide Time: 07:06)



```
31 den_con=[1 0];
32 Ge=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;
36
37 K_pd=K_p; K_id=K_i*T;
38
39 %% Loop gain and closed-loop TFs
40 G_loop=Gve*Ge; %% Loop gain
41
42 figure(3)
43 bode(G_loop,'');
44 hold on; grid on;
45
46 %% Transient parameters and transient response
47 t_sim=5e-3; t_step=3e-3;
48 delta_lo=20; delta_Vin=0; delta_Vref=0;
49
50 buck_converter_simulation;
```

So, if we consider the traditional current mode control design and we are assuming that ESR is very small, we are not going to compensate ESR, so we are taking a simple PI controller.

(Refer Slide Time: 07:19)



```
16
17 %% Open-loop Output Impedance
18 num_o=R*[1/w_z 1];
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
21
22 %% Modulator gain
23 V_m=10; Fm=1/V_m;
24 t_s=0.9*T; t_c=0.3*T; k_ff=0;
25
26 %% PI Controller Design
27
28 f_c=input('Select BW in kHz ... ');
29 K_c=(2*pi*f_c*1e3)/R;
30 num_con=K_c*[1 den_c];
31 den_con=[1 0];
32 Ge=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;
```

As we have already discussed the PI is nothing, but the integral gain is nothing but 2π ; then your crossover frequency, that means it is crossover frequency by R . So, this whole term is the crossover frequency in radian. So, if we set the crossover frequency or the bandwidth is let us say 50 kilohertz, it is in kilohertz or 80 kilohertz, accordingly the gain will be

calculated. Then we are finding the numerator gain which is the denominator of this; because we are doing some stable pole-zero cancellation. And what is the denominator of c ?

(Refer Slide Time: 08:00)

```

7
8 %% Define zeros and poles
9
10 w_z=1/(r_C*C); w_p=1/((R+r_C)*C);
11
12 %% Control-to-output TF Gvd
13 num_c=R*[1/w_z 1];
14 den_c=[1/w_p 1];
15 Gvc=tf(num_c,den_c);
16
17 %% Open-loop Output Impedance
18 num_o=R*[1/w_z 1];
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
21
22 %% Modulator gain
23 V_m=10; Fm=1/V_m;
24 t_s=0.9*T; t_c=0.3*T; k_ff=0;
25
26 %% PI Controller Design

```

It is nothing but denominator c it is nothing, but it consists of the pole of the first-order approximated model of the under-of-the-buck converter under current mode control. So, that will be the numerator.

(Refer Slide Time: 08:17)

```

22 %% Modulator gain
23 V_m=10; Fm=1/V_m;
24 t_s=0.9*T; t_c=0.3*T; k_ff=0;
25
26 %% PI Controller Design
27
28 f_c=input('Select BW in kHz ... ');
29 K_c=(2*pi*f_c*1e3)/R;
30 num_con=K_c*den_c;
31 den_con=[1 0];
32 Gc=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;
36 K_pd=K_p; K_id=K_i*T;
37
38 %% Loop gain and closed-loop TFs
39 G_loop=Gvc*Gc; %% Loop gain
40
41

```

And derivative is nothing but a simple integrator at origin; that means a pure integrator. So, in this case, the K_i is this, K_p is we have discussed K_i time R into C ; R is the load resistance the discrete time, K_p gain will remain as the proportional to the analog controller gain. But the discrete-time integral gain is the continuous time integral gain into the sampling time and then these parameters will be set into the digital controller and we want to compare. So, let us first consider and we are initially considering no feed forward, of the load current, so it is 0.

(Refer Slide Time: 08:54)

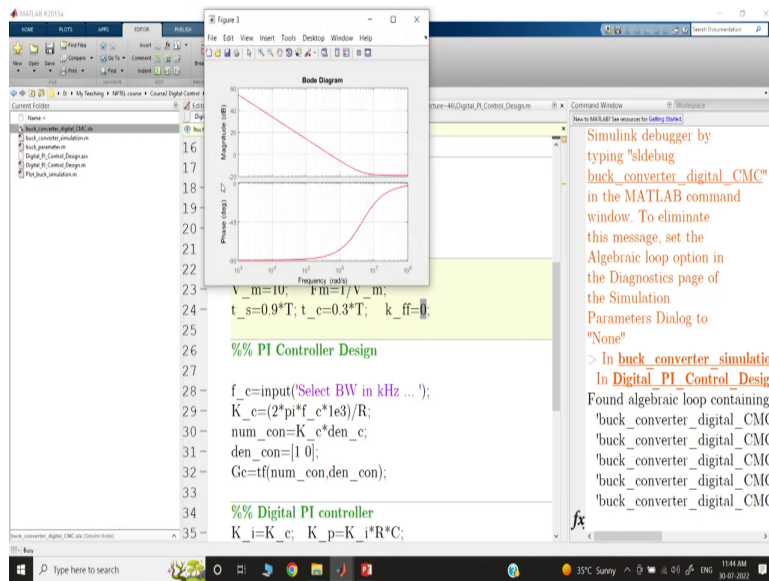
```

16
17 %% Open-loop Output Impedance
18 num_o=R*[1/w z 1];
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
21
22 %% Modulator gain
23 V_m=10; F_m=1/V_m;
24 t_s=0.9*T; t_c=0.3*T; k_ff=0;
25
26 %% PI Controller Design
27
28 f_c=input('Select BW in kHz ... ');
29 K_c=(2*pi*f_c*1e3)/R;
30 num_con=K_c*den_c;
31 den_con=[1 0];
32 Ge=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;

```

So, let us consider. So, it will ask for the bandwidths. So, let us set the bandwidth to be let us say 80 kilohertz.

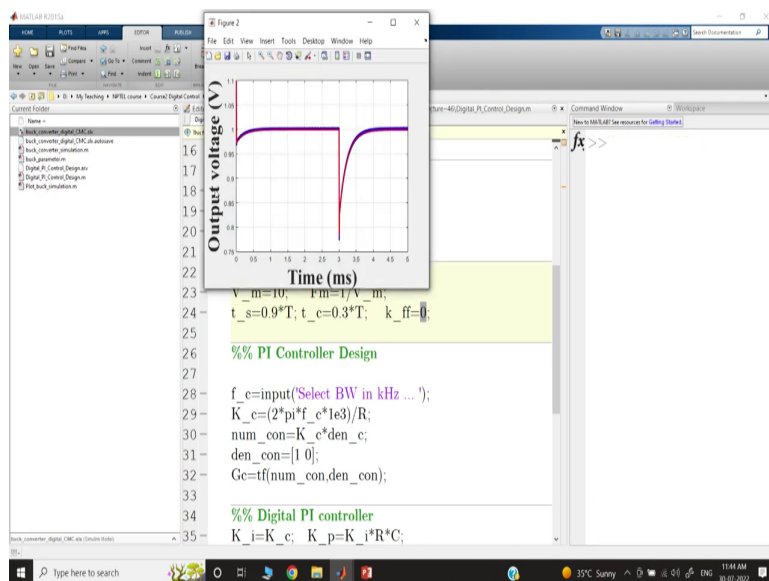
(Refer Slide Time: 09:08)



Simulink debugger by typing "sdebug buck_converter_digital_CMC" in the MATLAB command window. To eliminate this message, set the Algebraic loop option in the Diagnostics page of the Simulation Parameters Dialog to "None"
> In buck_converter_simulation In Digital_PI_Control_Design Found algebraic loop containing 'buck_converter_digital_CMC' 'buck_converter_digital_CMC' 'buck_converter_digital_CMC' 'buck_converter_digital_CMC' 'buck_converter_digital_CMC'

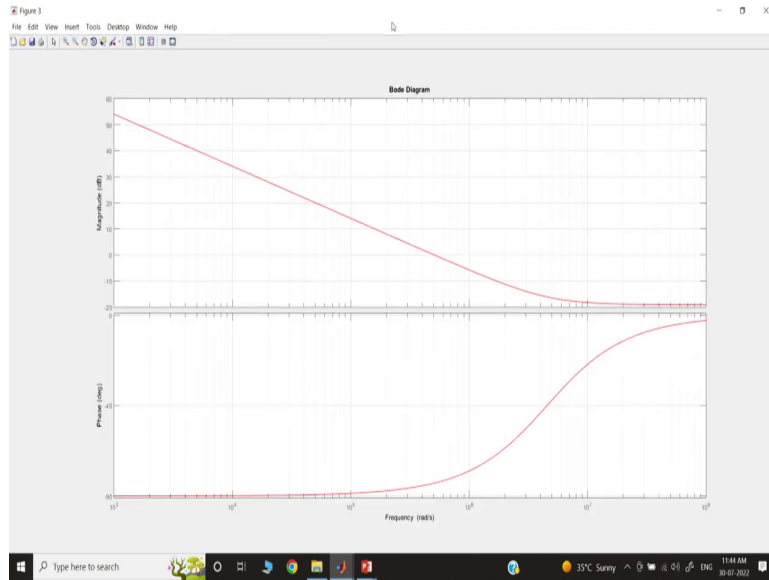
Then it will show the response.

(Refer Slide Time: 09:09)



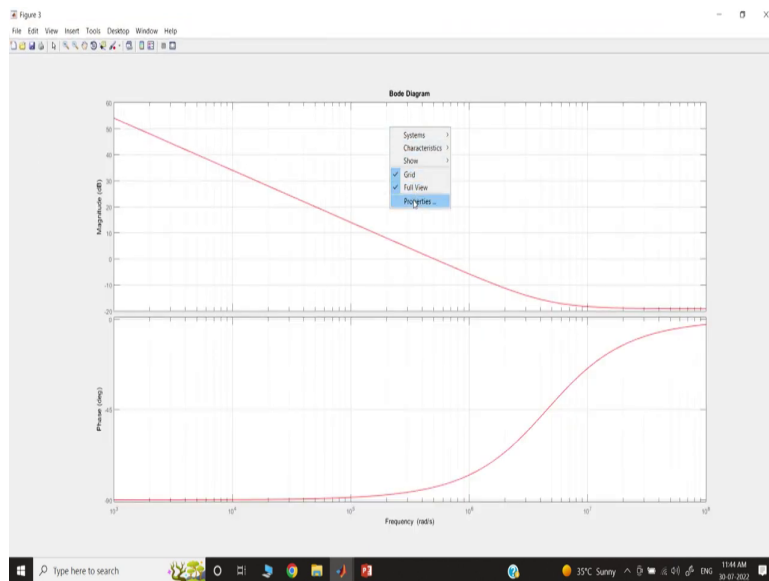
And before going to that, we will go to MATLAB.

(Refer Slide Time: 09:13).

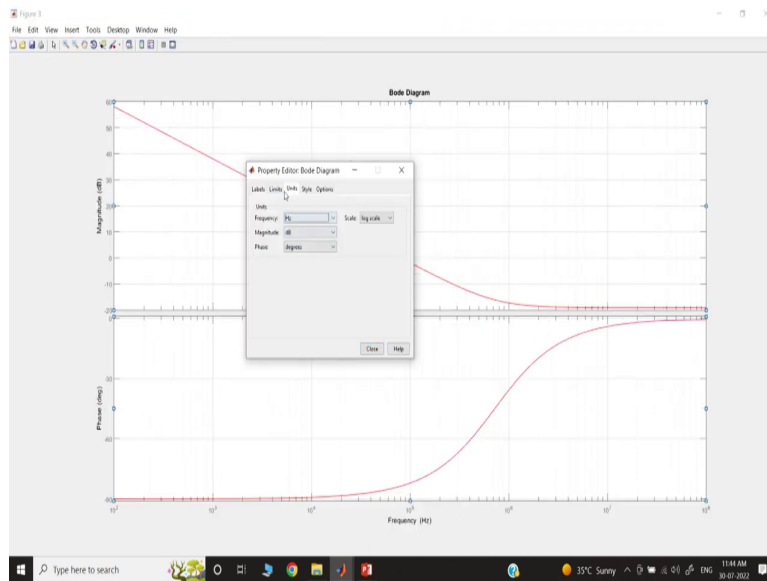


And see the Loop transfer function.

(Refer Slide Time: 09:14)

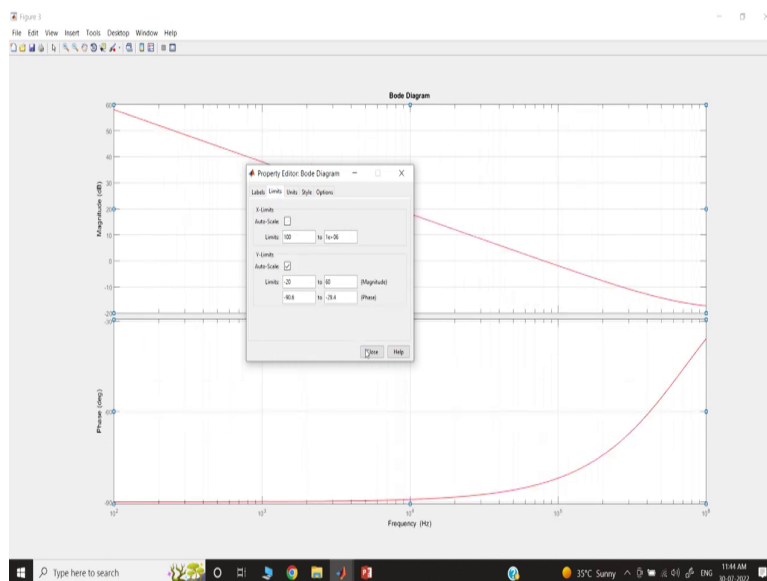


(Refer Slide Time: 09:15)



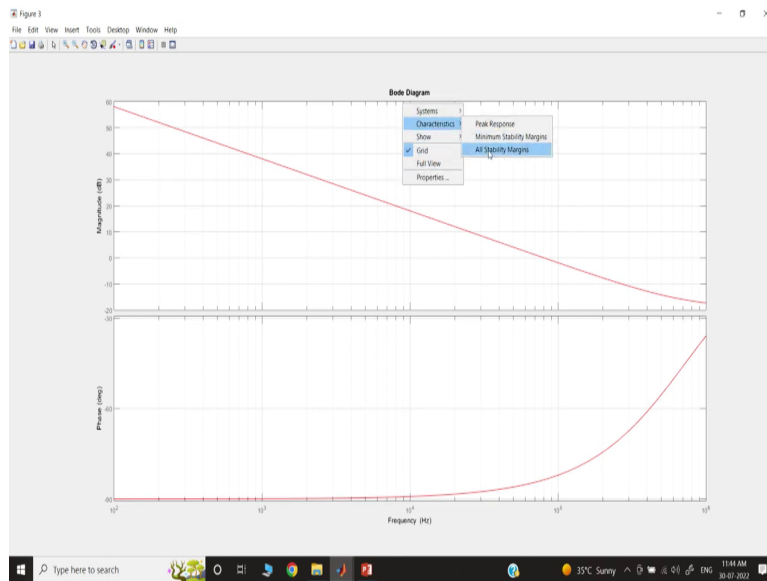
So, here we will set the loop transfer function.

(Refer Slide Time: 09:18)



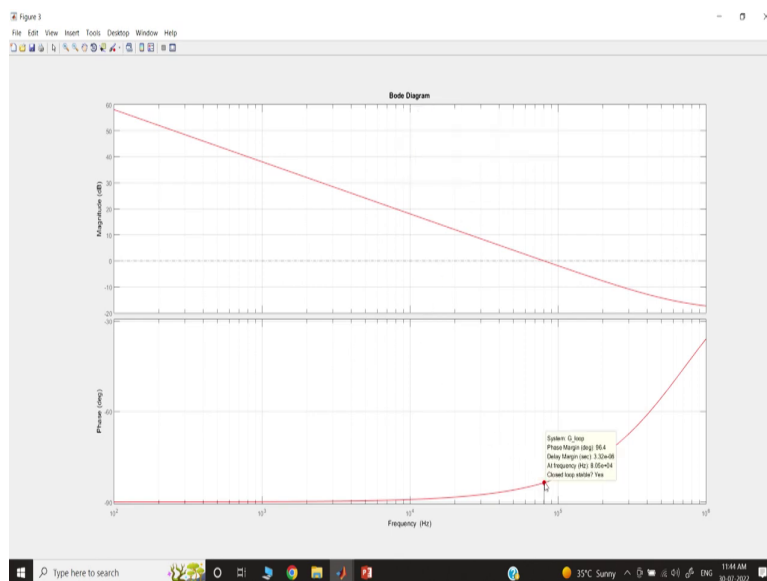
And the limit we set.

(Refer Slide Time: 09:23)



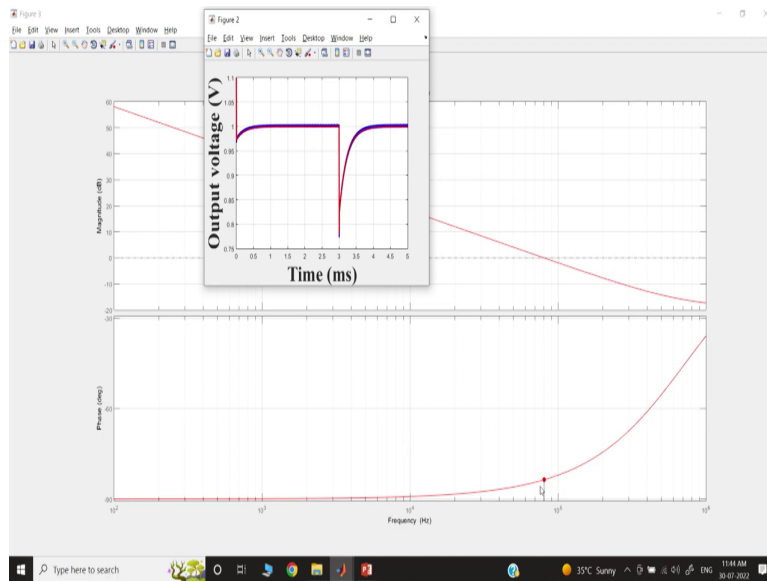
Then we want to see all the stability margins.

(Refer Slide Time: 09:27)



So, you can see, we are getting 80 kilohertz as your crossover frequency and the phase margin is 96.4 much higher phase margin; because ESR is there though we have ignored it, it is at a high frequency. So, due to that, it will give a phase boost ok; we are not exactly canceling because we are not anticipating ESR, but that is fine. So, now, we are getting phase margin.

(Refer Slide Time: 09:56)



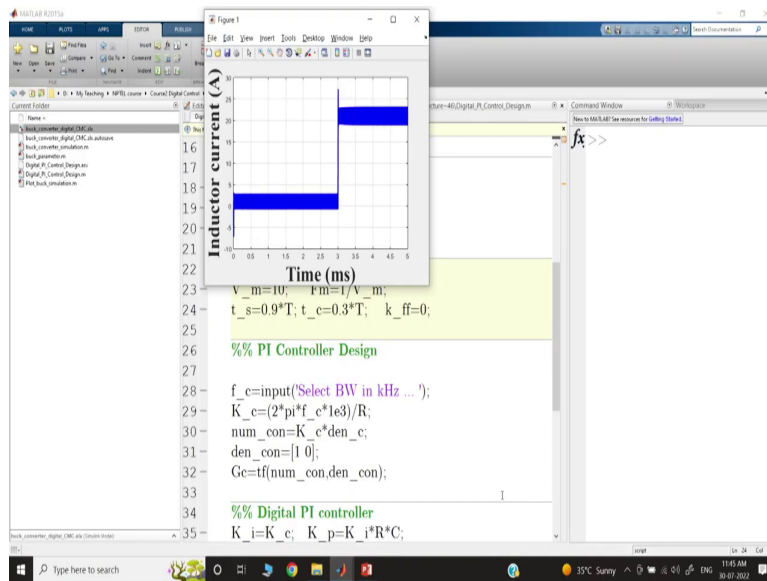
Then, when you go to digital control, we will have an additional lag. And the phase delay is how much?

(Refer Slide Time: 10:06)

```
16
17 %% Open-loop Output Impedance
18 num_o=R*(1/w_z 1);
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
21
22 %% Modulator gain
23 V_m=10; Fm=1/V_m;
24 t_s=0.9*T; t_c=0.3*T; k_ff=0;
25
26 %% PI Controller Design
27
28 f_c=input('Select BW in kHz ... ');
29 K_c=(2*pi*f_c*1e3)/R;
30 num_con=K_c*den_c;
31 den_con=[1 0];
32 Ge=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;
```

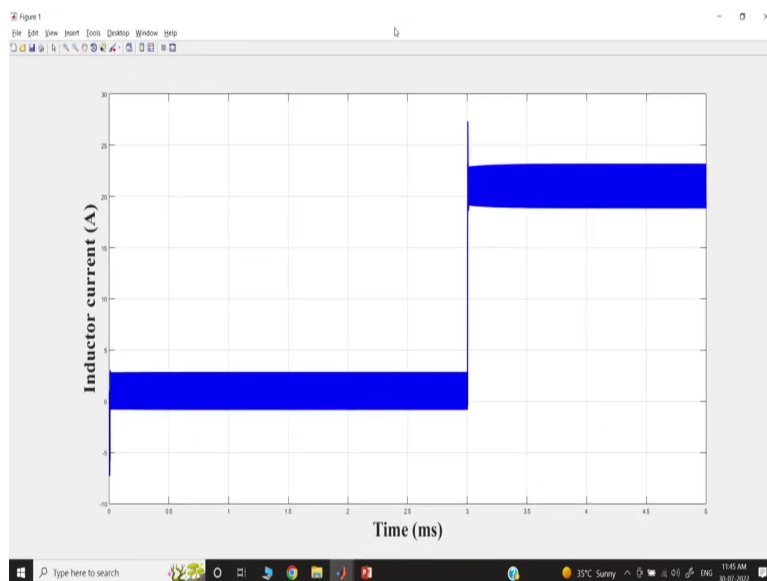
Here 0.9 time T; that means we are taking the sample first and then switching and this duration is 0.1 t, which means t by 10. So, we are taking a sampling delay of t by 10; because it is fast sampling then switching and that actually will give rise to some kind of phase lag.

(Refer Slide Time: 10:30)

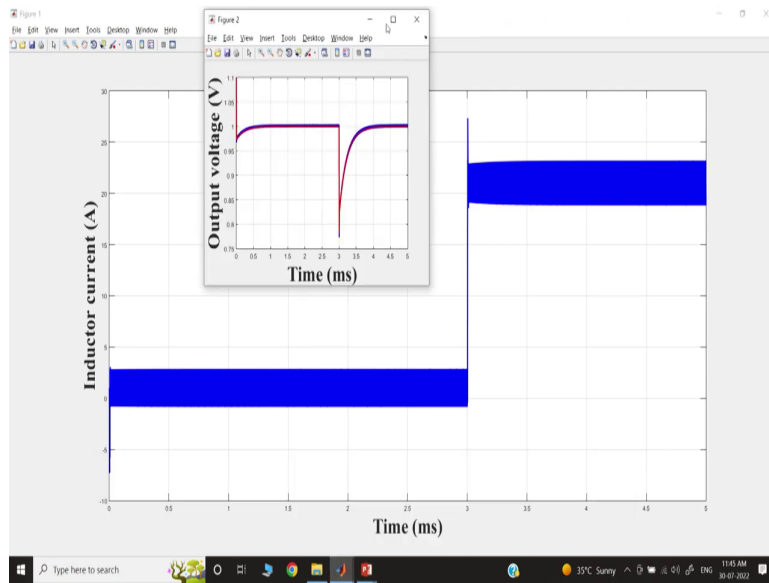


So, then if you go to the response, this is the inductor current.

(Refer Slide Time: 10:34)

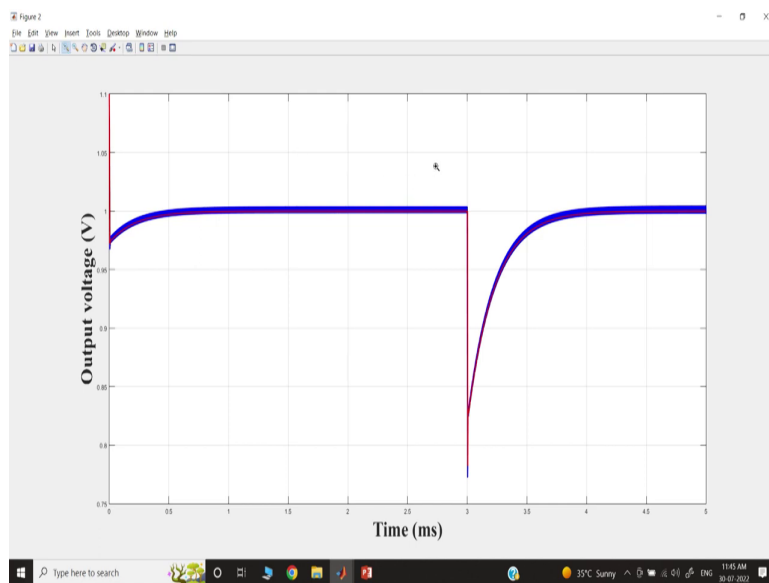


(Refer Slide Time: 10:36)



And this is an output voltage.

(Refer Slide Time: 10:38)



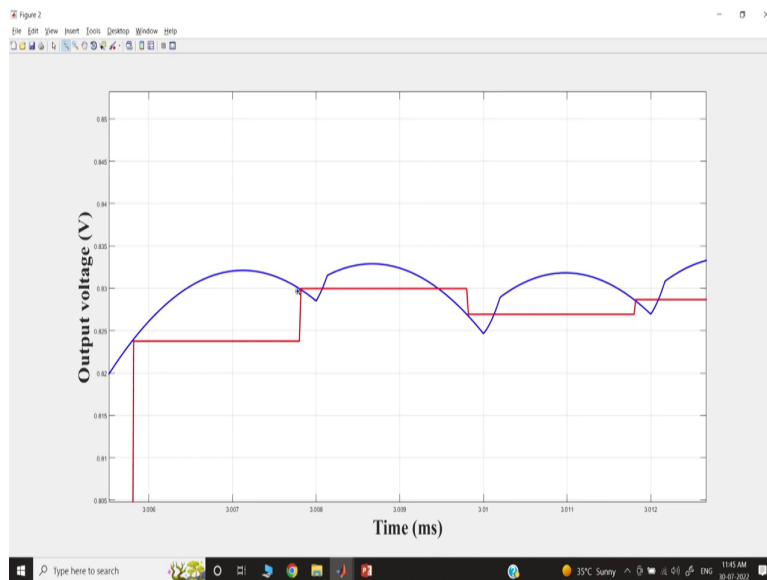
So, here you can see the output voltage.

(Refer Slide Time: 10:46)



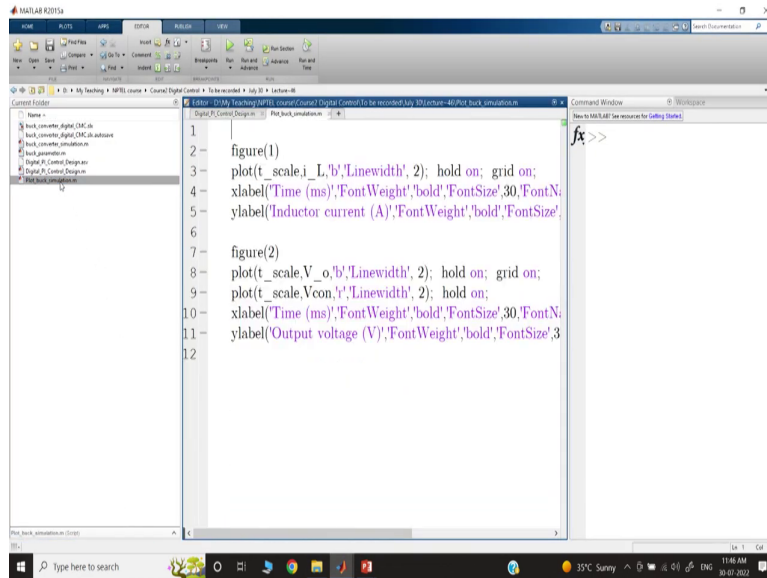
The red one is the sample voltage and I want to show you if you consider a sample voltage; you can see we are taking the sample first and then switching.

(Refer Slide Time: 10:53)



So, there is a $t_{by 10}$ delays and this delay will cause some additional phase lack compared to analog control. Now, the next task once you do; we want to compare what will happen if we add a loaded feed forward.

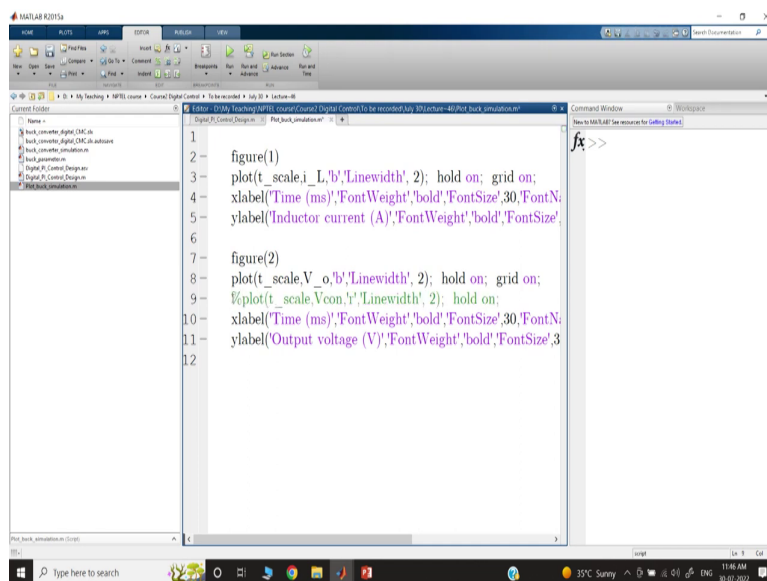
(Refer Slide Time: 11:09)



```
1 figure(1)
2 plot(t_scale,i_L,'b','Linewidth',2); hold on; grid on;
3 xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontN
4 ylabel('Inductor current (A)','FontWeight','bold','FontSize',
5
6
7 figure(2)
8 plot(t_scale,V_o,'b','Linewidth',2); hold on; grid on;
9 plot(t_scale,Vcon,'r','Linewidth',2); hold on;
10 xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontN
11 ylabel('Output voltage (V)','FontWeight','bold','FontSize',3
12
```

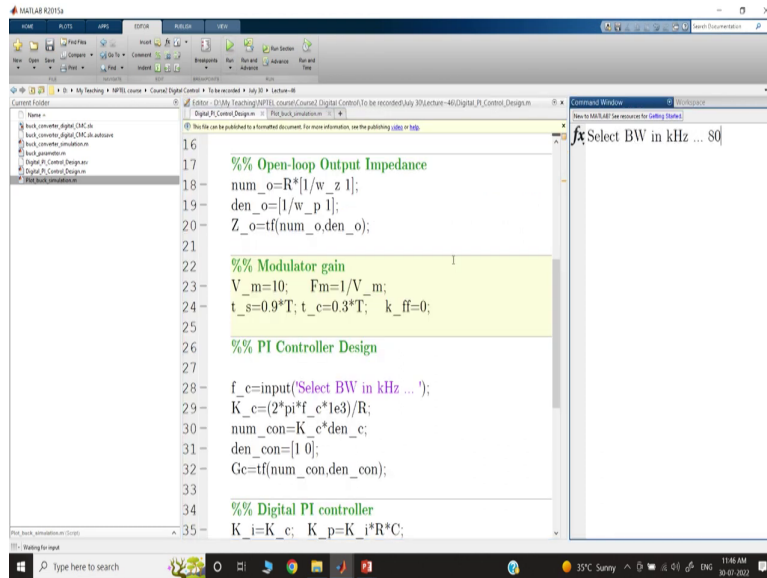
So, in that case, we will simply not show the sample.

(Refer Slide Time: 11:14)



```
1 figure(1)
2 plot(t_scale,i_L,'b','Linewidth',2); hold on; grid on;
3 xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontN
4 ylabel('Inductor current (A)','FontWeight','bold','FontSize',
5
6
7 figure(2)
8 plot(t_scale,V_o,'b','Linewidth',2); hold on; grid on;
9 %plot(t_scale,Vcon,'r','Linewidth',2); hold on;
10 xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontN
11 ylabel('Output voltage (V)','FontWeight','bold','FontSize',3
12
```

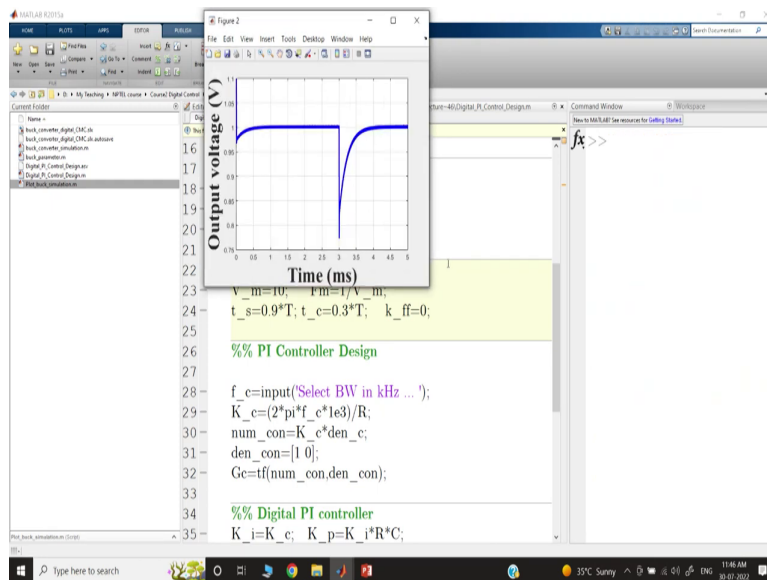
(Refer Slide Time: 11:16)



```
16
17
18 %% Open-loop Output Impedance
19 num_o=R*(1/w_z 1);
20 den_o=|1/w_p 1|;
21 Z_o=tf(num_o,den_o);
22
23 %% Modulator gain
24 V_m=10; Fm=1/V_m;
25 t_s=0.9*T; t_c=0.3*T; k_ff=0;
26
27 %% PI Controller Design
28 f_c=input('Select BW in kHz ... ');
29 K_c=(2*pi*f_c*1e3)/R;
30 num_con=K_c*den_c;
31 den_con=|1 0|;
32 Ge=tf(num_con,den_con);
33
34 %% Digital PI controller
35 K_i=K_c; K_p=K_i*R*C;
```

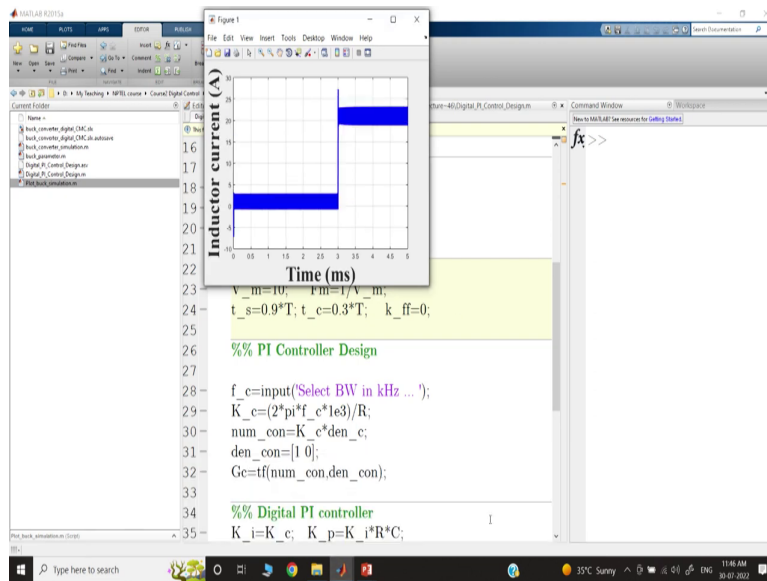
So, let us first simulate once more; that means 80 kilohertz; now we do not want to show the sample signal.

(Refer Slide Time: 11:22)



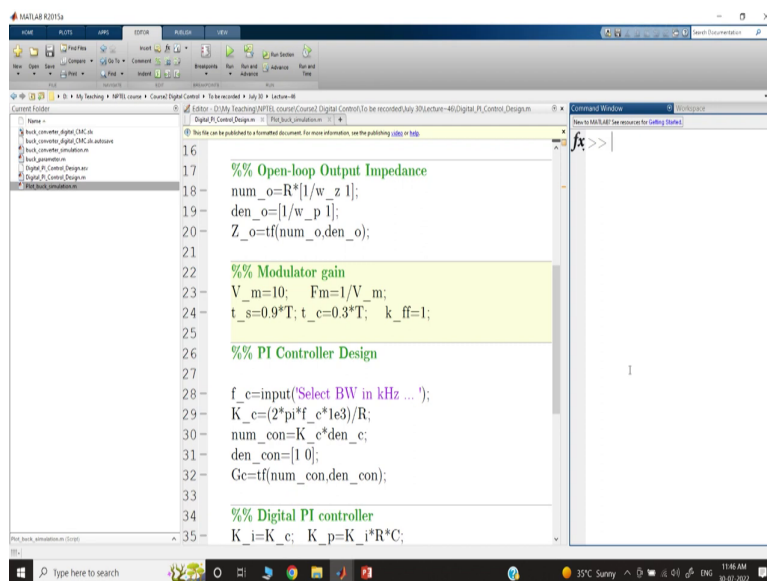
So, this is the output voltage without load feed forward.

(Refer Slide Time: 11:27)



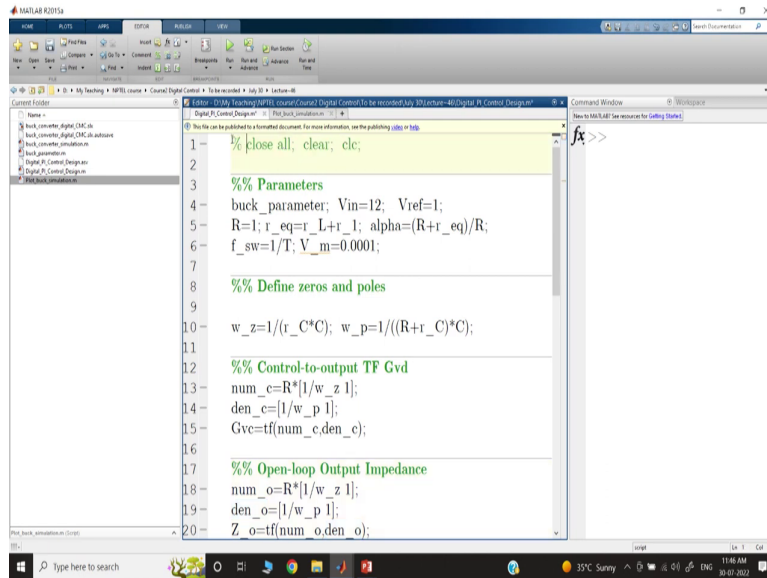
And this is the inductor current. Now, we want to go for load feed forward.

(Refer Slide Time: 11:30)



So, let us use 1 and we have used 80 kilohertz as the cross-over frequency.

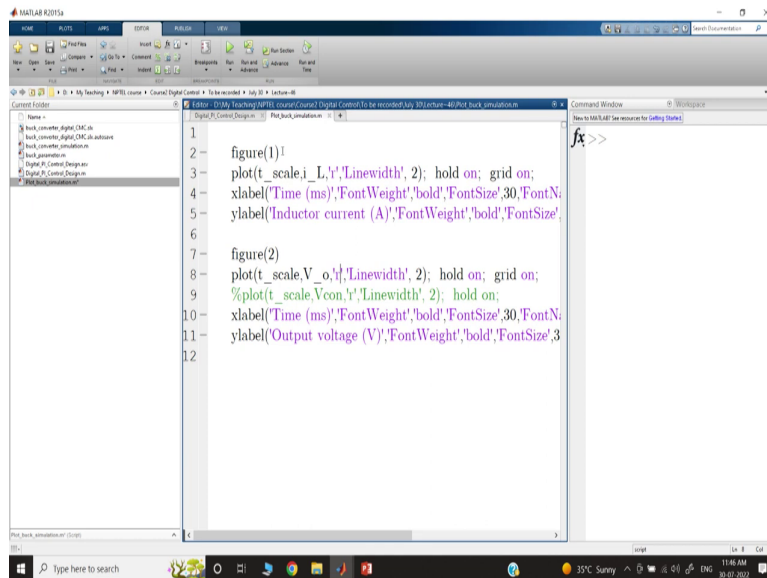
(Refer Slide Time: 11:40)



```
1 % close all, clear, clc;
2
3 %% Parameters
4 buck_parameter; Vin=12; Vref=1;
5 R=1; r_eq=r_L+r_1; alpha=(R+r_eq)/R;
6 f_sw=1/T; V_m=0.0001;
7
8 %% Define zeros and poles
9
10 w_z=1/(r_C*C); w_p=1/((R+r_C)*C);
11
12 %% Control-to-output TF Gvd
13 num_c=R*[1/w_z 1];
14 den_c=[1/w_p 1];
15 Gvc=tf(num_c,den_c);
16
17 %% Open-loop Output Impedance
18 num_o=R*[1/w_z 1];
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
```

Now, we want to rerun this simulation and we want to hold the previous value and change the color of the plot.

(Refer Slide Time: 11:48)



```
1
2 figure(1);
3 plot(t_scale,i_L,'r',Linewidth, 2); hold on; grid on;
4 xlabel('Time (ms)',FontWeight,'bold',FontSize,30,FontN;
5 ylabel('Inductor current (A)',FontWeight,'bold',FontSize,
6
7
8 figure(2);
9 plot(t_scale,V_o,'r',Linewidth, 2); hold on;
10 %plot(t_scale,Vcon,'r',Linewidth, 2); hold on;
11 xlabel('Time (ms)',FontWeight,'bold',FontSize,30,FontN;
12 ylabel('Output voltage (V)',FontWeight,'bold',FontSize,3
```

So, we are using red color for plotting the current and voltage for this case, now we have added the load feeds forward.

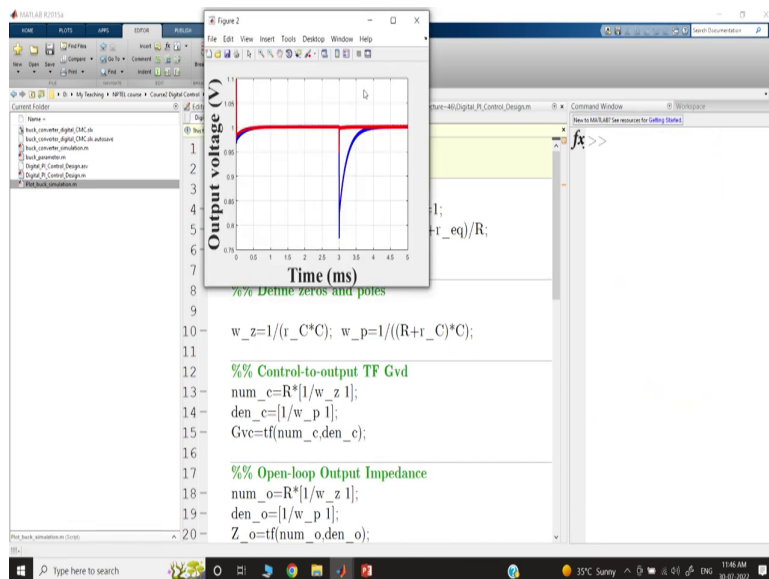
(Refer Slide Time: 11:57)

```
1 % close all; clear; clc;
2
3 %% Parameters
4 buck_parameter; Vin=12; Vref=1;
5 R=1; r_eq=r_L+r_1; alpha=(R+r_eq)/R;
6 f_sw=1/T; V_m=0.0001;
7
8 %% Define zeros and poles
9
10 w_z=1/(r_C*C); w_p=1/((R+r_C)*C);
11
12 %% Control-to-output TF Gvd
13 num_c=R*[1/w_z 1];
14 den_c=[1/w_p 1];
15 Gvc=tf(num_c,den_c);
16
17 %% Open-loop Output Impedance
18 num_o=R*[1/w_z 1];
19 den_o=[1/w_p 1];
20 Z_o=tf(num_o,den_o);
```

```
>> Digital_PI_Control_Design
fx Select BW in kHz ... 80
```

So, again we are giving 80 and see what happened.

(Refer Slide Time: noon)

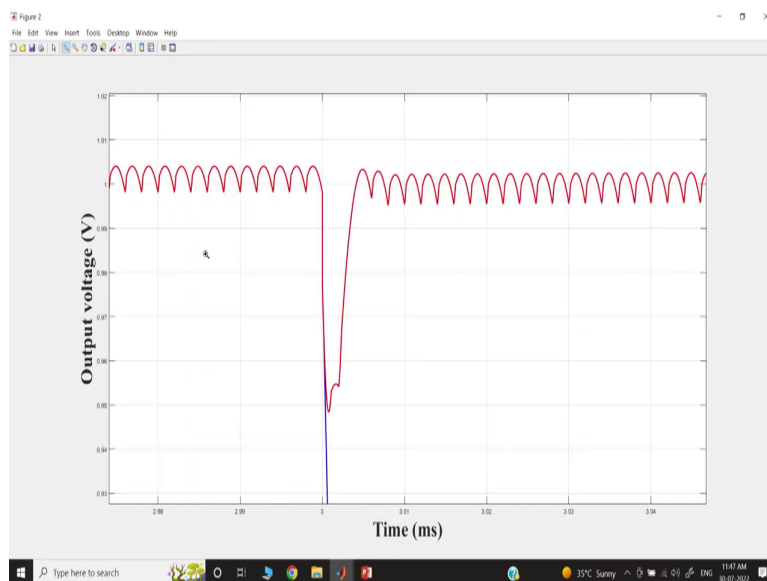


(Refer Slide Time: 12:01)



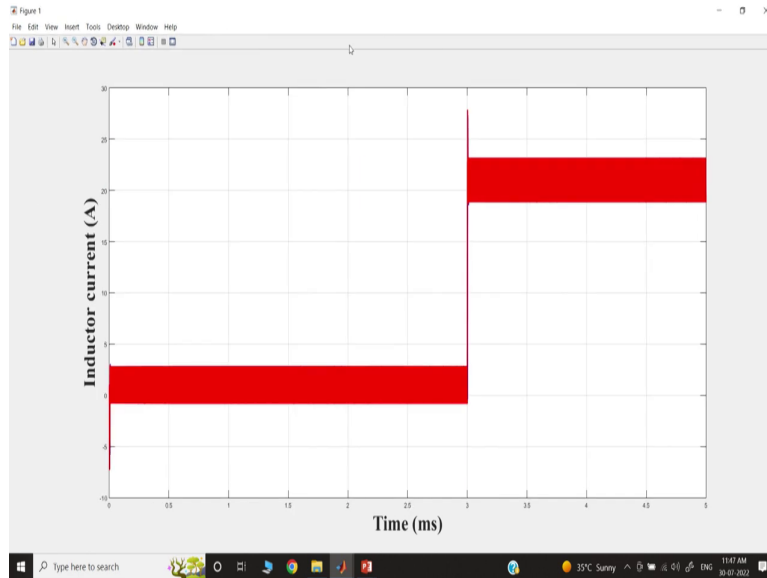
So, you can see because of the load feedforward, the response is extremely fast.

(Refer Slide Time: 12:08)



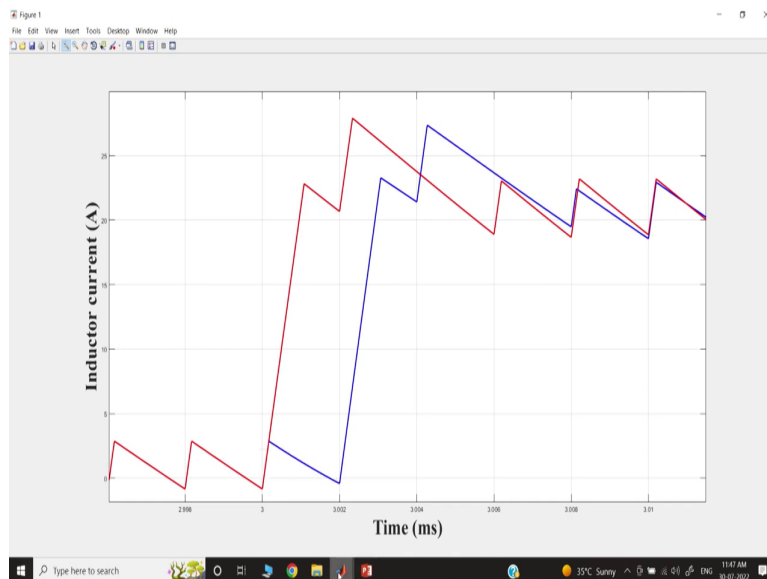
You can see this is extremely fast is like almost like time optimal.

(Refer Slide Time: 12:12)



That means you can see the inductor current will come back red color.

(Refer Slide Time: 12:18)



It is not exactly time optimal, but. So, because there is no load feedforward, there was an initial delay in the voltage and that is causing one of the issues with the addition undershot. So, this is because we have applied a load step at the point of switching and a sample for that signal was captured here. So, it did not recognize the delay; which means it was showing almost one full cycle delay, and the inductor current has not responded to the transient.

So, the transient is applied here, but the sample is captured somewhere at this point. So, that is why the sample after sampling the peak current does not know that there is a transient here, so it is almost like a one-cycle delay and that is why you will get an additional undershoot. Whereas, if you add a load feed, the peak referential automatically get set; because it is actually as if we are subtracting the load current from the inductor current or we can add the load current with the reference current. So, the reference current will get automatically shifted by this load feed forward.

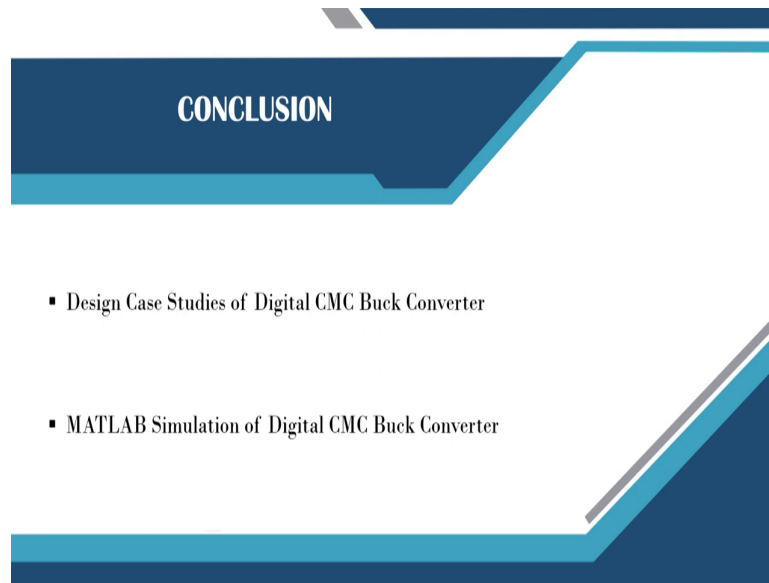
But it may be difficult to sense load current in practice because it is not permissible; but in many cases, if you take an LED driving case study or maybe if you can interact with the processor information, you can make the output impedance as well as the loop transient response extremely fast or output impedance to be very low, it can make it using load feed forward.

So, in summary, what we want to discuss is that you know we want to go back. So, we have considered that because of the delay; you will get some additional because it will take time to respond. So, that is why in many commercial products, they try to increase the sampling frequency during transient to speed up. Another way they can use an analog comparator is using a voltage stasis, so that if the voltage goes beyond this band; then they can either increase the sampling rate or they can you know suddenly create a step reference to anticipate the load tangent effect.

So, there are many ways off, because in the subsequent lecture, in the next lecture; we will be discussing how to go beyond this small signal design so that you can get a very first transient. Because the digital control if you look at the small signal-based design compared to the continuous time control, actually it gives rise to additional delay. So, digital control may not be a useful solution if you go by a small signal design.

But the digital platform offers you to change the controller gain based on the operating point, that is the only advantage and adapted dead time; but unless you implement some more advanced logic, I mean this digital control may not be so useful and that is why the majority of the commercial product is going with some non-linear or some other large signal based design, where you can actually capitalize this digital controller and you can also introduce some tuning as well as an advance algorithm.

(Refer Slide Time: 15:23)



CONCLUSION

- Design Case Studies of Digital CMC Buck Converter
- MATLAB Simulation of Digital CMC Buck Converter

In summary, we have discussed the design case study of a digital current mode control buck converter and we have also considered some simulation case studies. And we have shown how to design using small signals, and digital current mode control, but this is this lecture we will show that this is not the superior way of digital control design. So, in a subsequent lecture, we will slowly tend towards the large signal-based design and show that the performance can be much faster by utilizing the trajectory information as well as some advanced control algorithm. We can achieve close to time optimal solutions.

And that is the future trend of digital control, but here we want to finish the small signal-based design methodology. We have already discussed the small signal-based design for voltage mode and in this lecture, we have discussed the small signal design case study of current mode control. Now, we will go for a large signal-based design in the subsequent lecture; that is it for today.

Thank you very much.