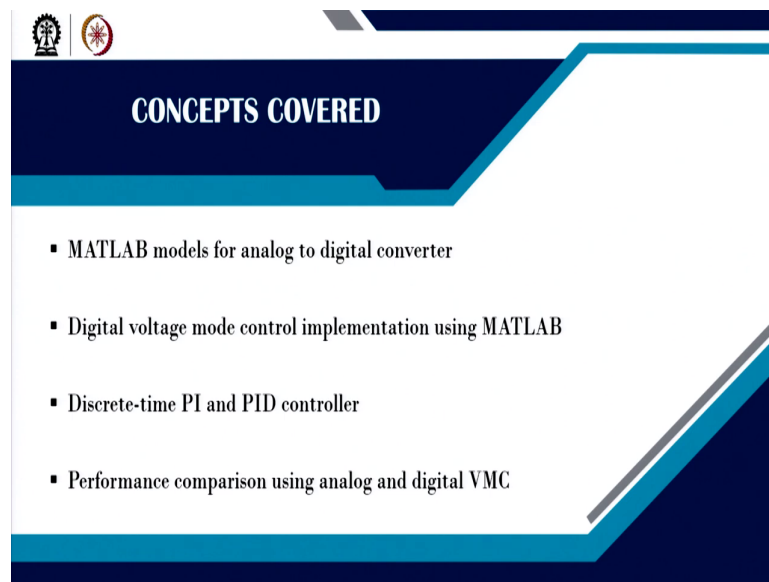


Digital Control in Switched Mode Power Converters and FPGA-based Prototyping
Prof. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Module - 03
MATLAB Custom Model Development under Digital Control
Lecture - 22
MATLAB Model Development for Basic Digital Control Blocks

Welcome back, so, in this lecture, we are going to talk about MATLAB Model Development for Basic Digital Control Block.

(Refer Slide Time: 00:31)

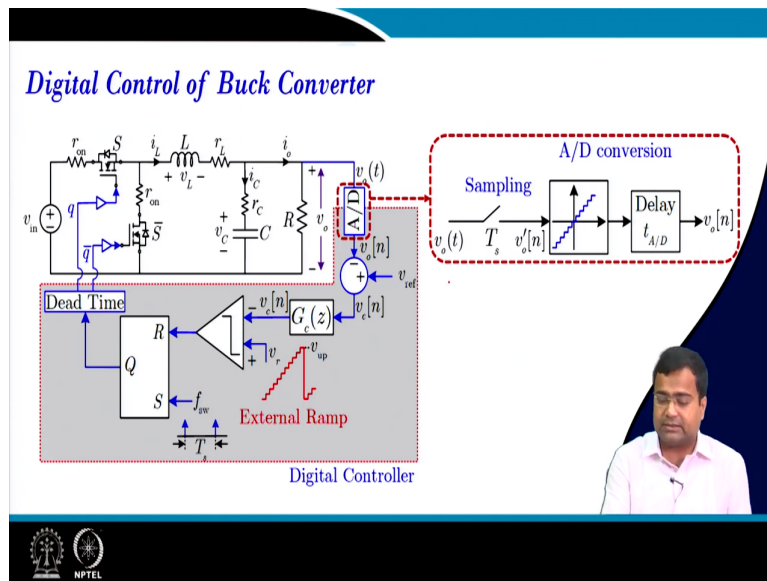


The slide features a dark blue background with a light blue diagonal stripe. At the top left, there are two small circular logos. The main title 'CONCEPTS COVERED' is centered in white. Below the title, a bulleted list of four items is presented in white text.

- MATLAB models for analog to digital converter
- Digital voltage mode control implementation using MATLAB
- Discrete-time PI and PID controller
- Performance comparison using analog and digital VMC

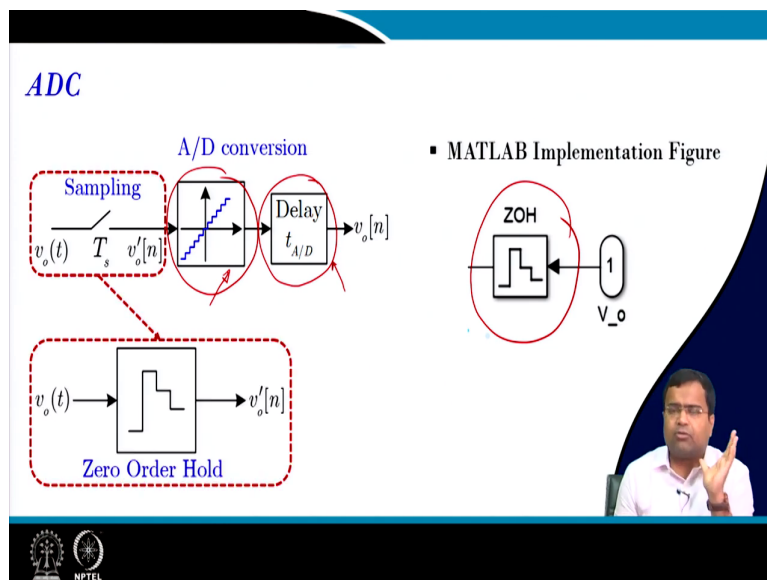
So, here we will first talk about model MATLAB Models for Analog to Digital Converter, then digital voltage mode control implementation using MATLAB. Then discrete-time PI and PID controller and the performance comparison using analog and digital voltage mode control.

(Refer Slide Time: 00:48)



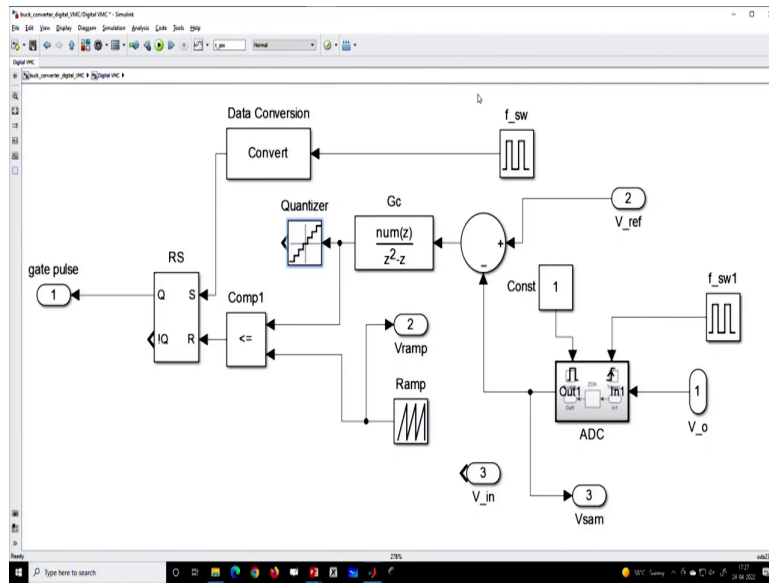
So, here first let us start with the digital control of a buck converter voltage mode control. So, here the A to D converter requires a sampler than a quantize, and delay.

(Refer Slide Time: 01:02)

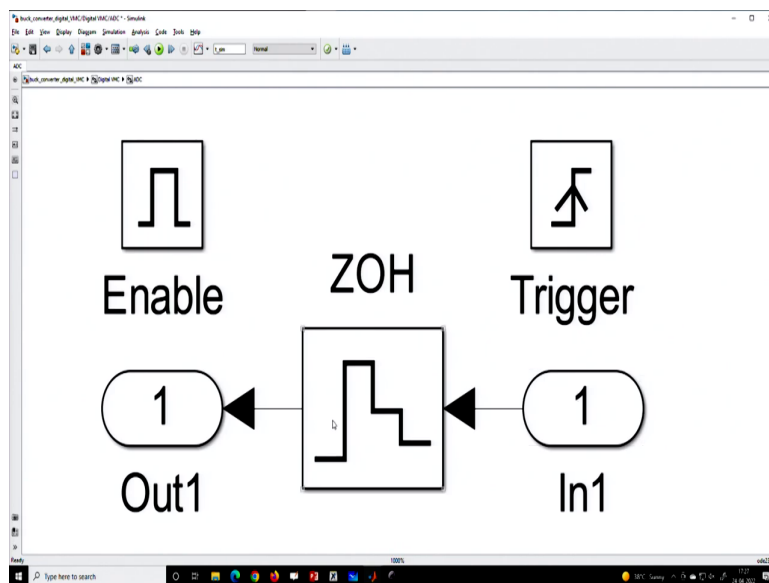


So, how do you implement this block? So, any sampling you know and hold; that means, in MATLAB you can easily get a zero-order hold circuit ok.

(Refer Slide Time: 01:12)



(Refer Slide Time: 01:17)



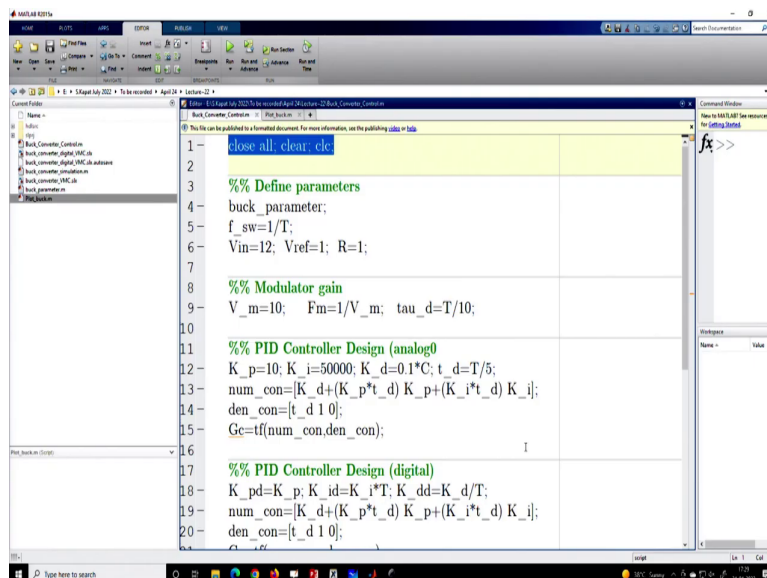
So, let us go to MATLAB and see; so, if you go inside I will say this zero-order hold block you can simply get this zero-order hold block and this block is a zero-order hold. If you go inside you need to provide some time, what is the sampling time ok? So, you can start with this sampling block you know I will come to this point what is this block?

voltage this is a sample voltage; that means if I consider this zero-order hold output voltage there is sample voltage and it is subtracted from the reference voltage and error goes.

Then you know we are using a discrete-time compensator; that means, you know depending upon what kind of controller you are using P PID PI all discrete time. Because it is we are going for digital control, the output is compared with the sawtooth. So, this sawtooth modulator latch they are the same as analog control; we are only considering this because we are not talking about the discretization in the sawtooth that will come later.

Assuming that the sawtooth has sufficient resolution then the compensator is the discrete-time compensator and this is the voltage sam.

(Refer Slide Time: 03:22)

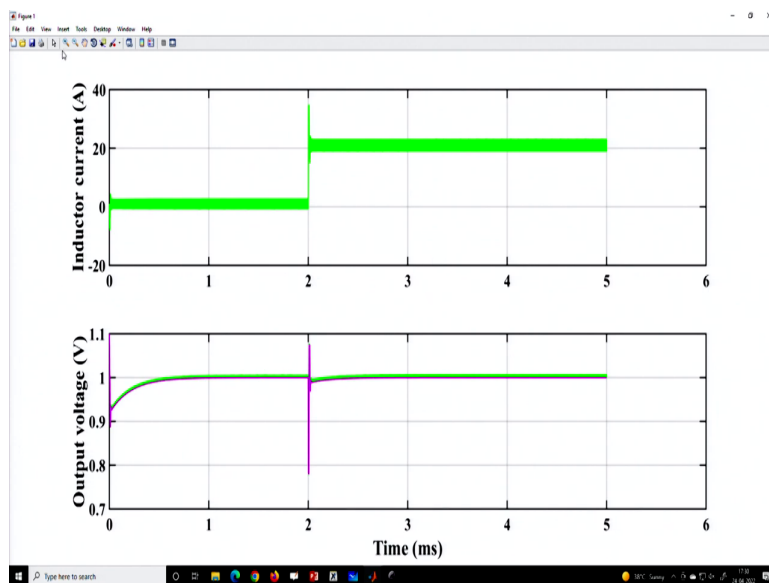


```
1- close all; clear; clc
2-
3- %% Define parameters
4- buck_parameter;
5- f_sw=1/T;
6- Vin=12; Vref=1; R=1;
7-
8- %% Modulator gain
9- V_m=10; Fm=1/V_m; tau_d=T/10;
10-
11- %% PID Controller Design (analog)
12- K_p=10; K_i=50000; K_d=0.1*C; t_d=T/5;
13- num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14- den_con=[t_d 1 0];
15- Gc=tf(num_con,den_con);
16-
17- %% PID Controller Design (digital)
18- K_pd=K_p; K_id=K_i*T; K_dd=K_d/T;
19- num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
20- den_con=[t_d 1 0];
```

(Refer Slide Time: 03:27)

```
10
11 %% PID Controller Design (analog)
12 K_p=10; K_i=50000; K_d=0.1*C; t_d=T/5;
13 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14 den_con=[t_d 1 0];
15 Gc=tf(num_con,den_con);
16
17 %% PID Controller Design (digital)
18 K_pd=K_p; K_id=K_i*T; K_dd=K_d/T;
19 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
20 den_con=[t_d 1 0];
21 Gc=tf(num_con,den_con);
22
23 %% Control method - option
24 op1='buck_converter_VMC.slx';
25 op2='buck_converter_digital_VMC.slx';
26
27 enter_file_name=op2;
28
29 %% Transient parameters and plots
```

(Refer Slide Time: 03:33)

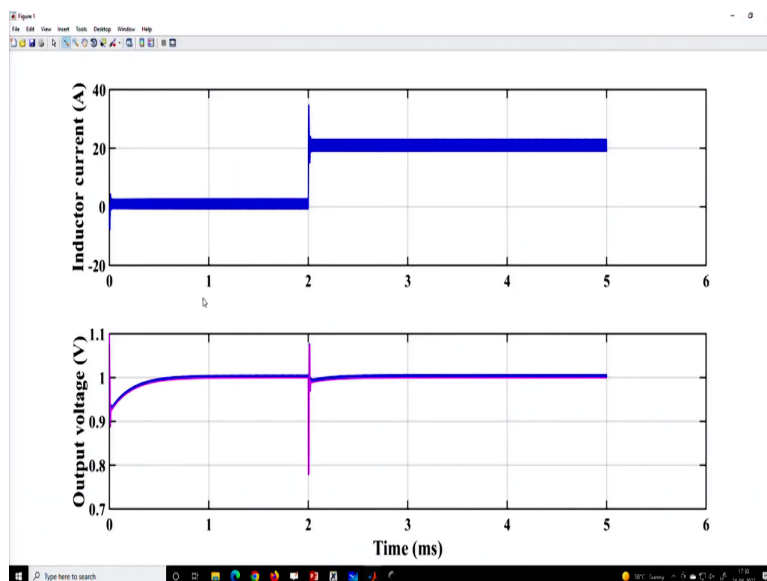


So, if you want to run this converter; so, again we are talking about digital control, and just let us ramp. So, we are running a load transient case study and I am showing it here.

(Refer Slide Time: 03:46)

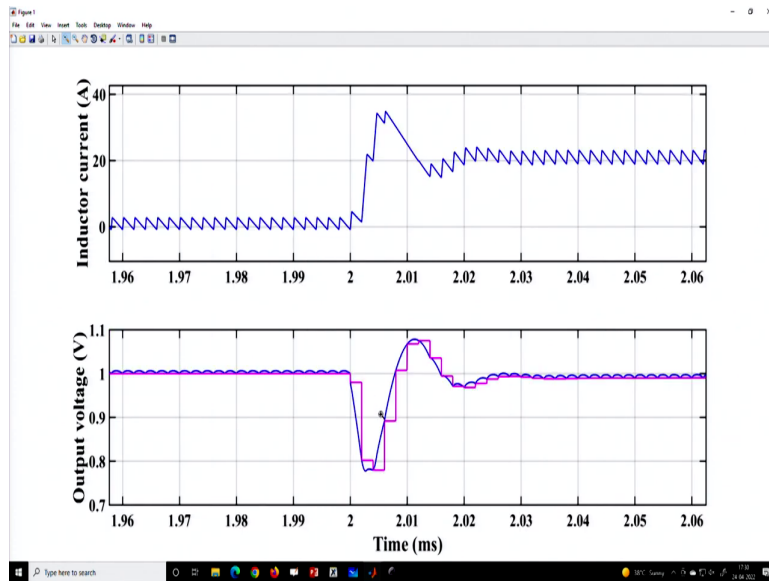
```
1- figure(1)
2-
3- plt1=subplot(2,1,1);
4- plot(t_scale,i_L,'b','Linewidth', 2); hold on;
5- set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','linewidth',2,'FontName',
6- ylabel('Inductor current (A)','FontWeight','bold','FontSize',30,'FontName','Times N
7- grid on;
8-
9- plt2=subplot(2,1,2);
10- plot(t_scale,V_o,'b','Linewidth', 2); hold on;
11- plot(t_scale,Vcon,'m','Linewidth', 2); hold on;
12- set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','linewidth',2,'FontName',
13- xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontName','Times New Roma
14- ylabel('Output voltage (V)','FontWeight','bold','FontSize',30,'FontName','Times N
15- grid on;
16-
17- linkaxes([plt1,plt2],'x')
```

(Refer Slide Time: 03:50)

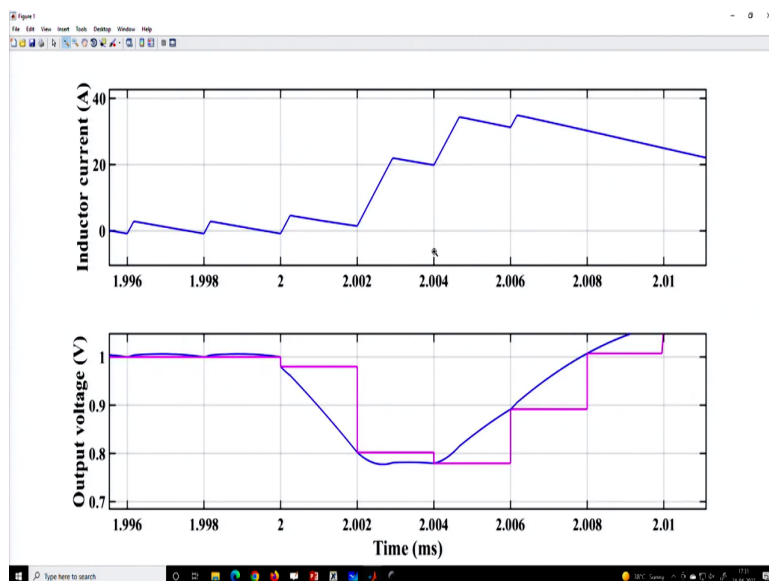


So, let us use another color let us use blue color, yeah, I am showing here a load transient case study of a buck converter and here I am using a discrete-time PID control. In the previous lecture, I talked about continuous time PID controllers with some values and I have exactly done the digital version of their continuous time. That means how k_I in the discrete domain is related to that in the continuous domain, how derivative in the discrete domain is related to that in the continuous domain using that mapping only I have used the coefficient.

(Refer Slide Time: 04:27)



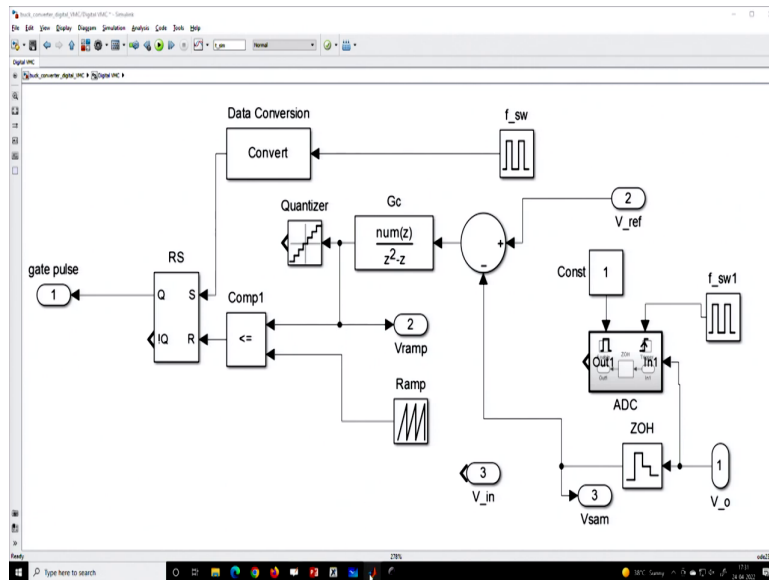
(Refer Slide Time: 04:40)



And you will see this is the transient response and this magenta color shows the bottom stress as the voltage output voltage and the magenta color as a sample voltage. And you can see they are sampled at every because this is the edge of switching. After all, we are talking about two microsecond time periods. So, at every rising edge of the clock actually, the output voltage is a sample and the same sample voltage is used to compute the discrete-time compensator what is because this is used the sample voltage is used?

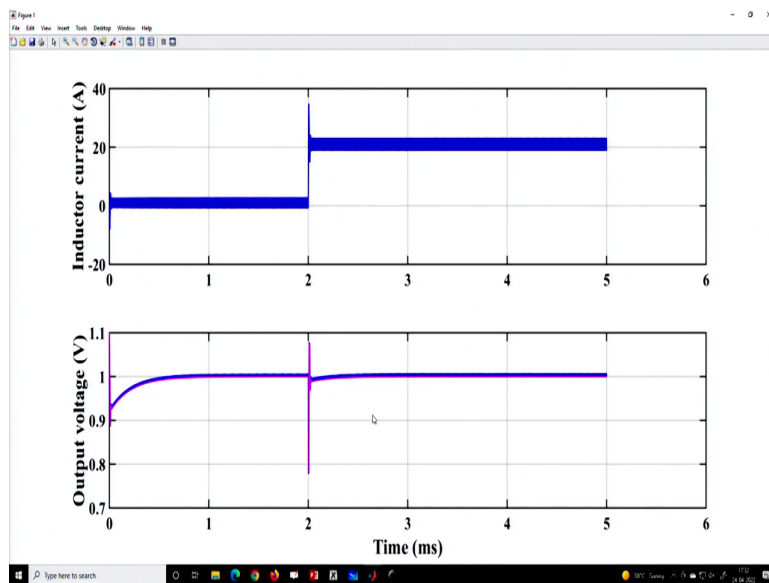
And this is the discrete-time compensator and this is the output of the compensator that is compared to the sawtooth.

(Refer Slide Time: 05:15)



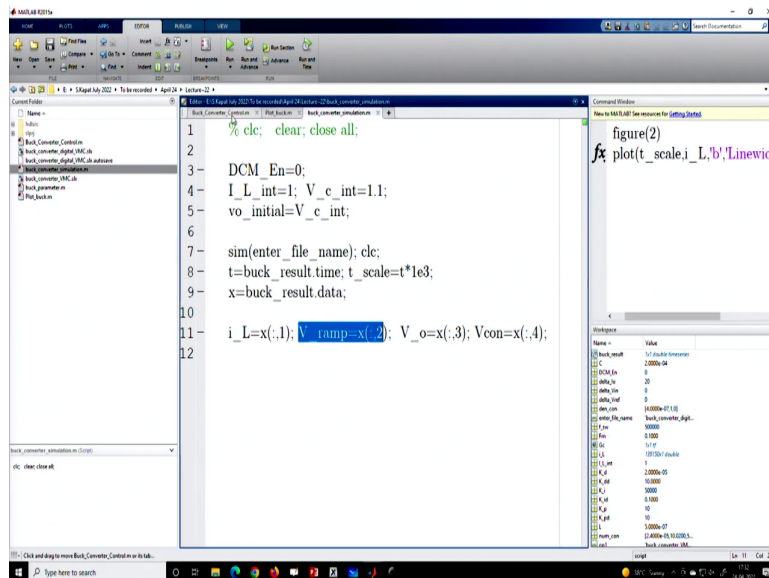
So, this compensator value; means, if we take you to know if we want to see let us say this response how does it look like; that means if I want to plot this. And if I rerun this simulation; that means, you know if I rerun this simulation I want to show how does the control voltage update.

(Refer Slide Time: 05:33)

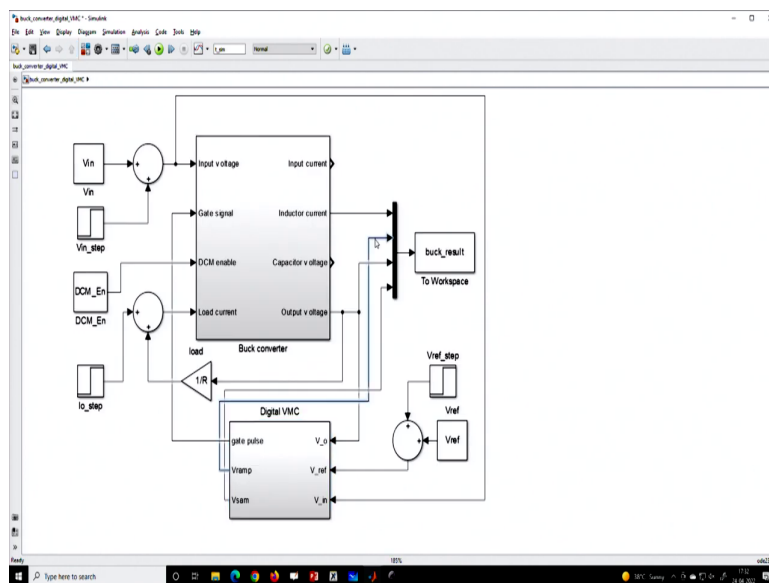


So, we can plot separately; that means if you go to the plot command let us plot; so, what are where it is stored buck converter simulation; so, this is our ramp ok.

(Refer Slide Time: 05:50)

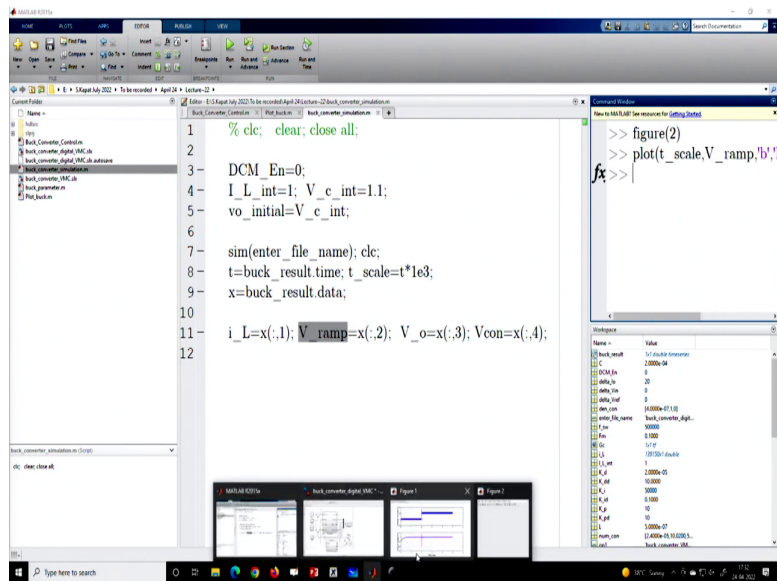


(Refer Slide Time: 05:55)

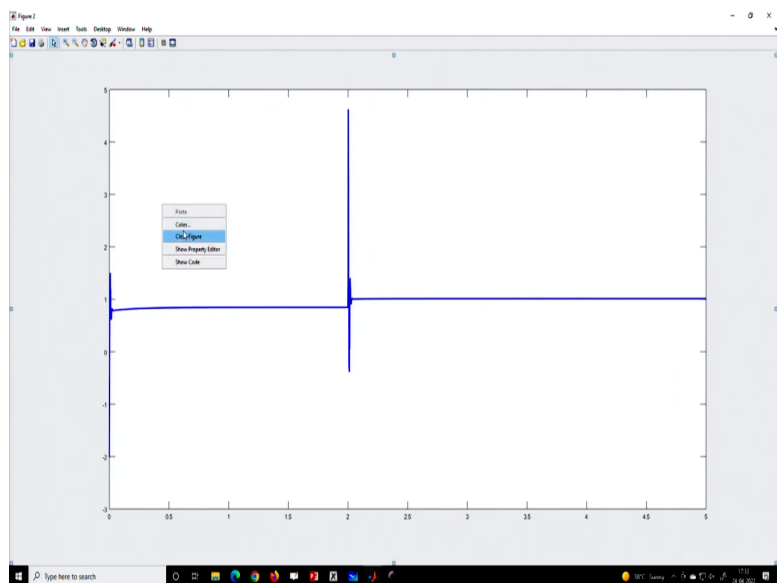


So, if you go to the ramp, we will see yes this is connected to the ramp ok all right? So, here we will figure two because we do not want to disturb the current figure and we want to plot what are we going to plot V ramp; that means, we want to plot this particular signal; so, this particular signal.

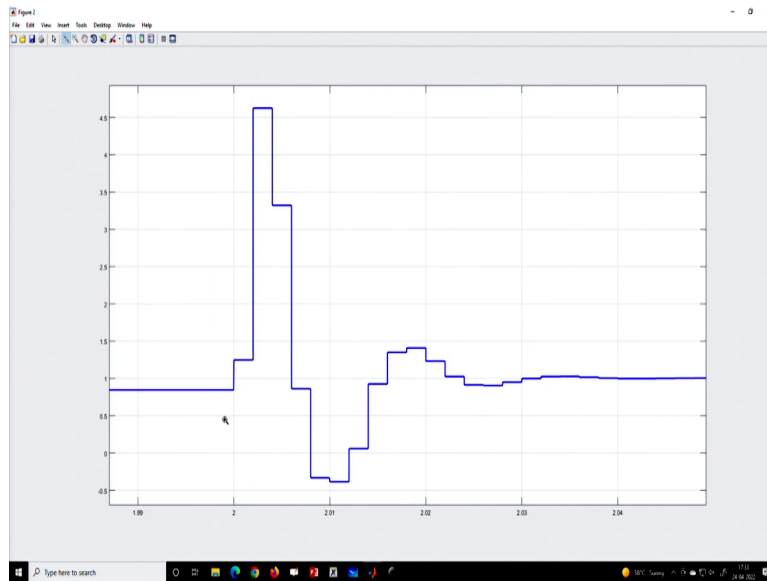
(Refer Slide Time: 06:20)



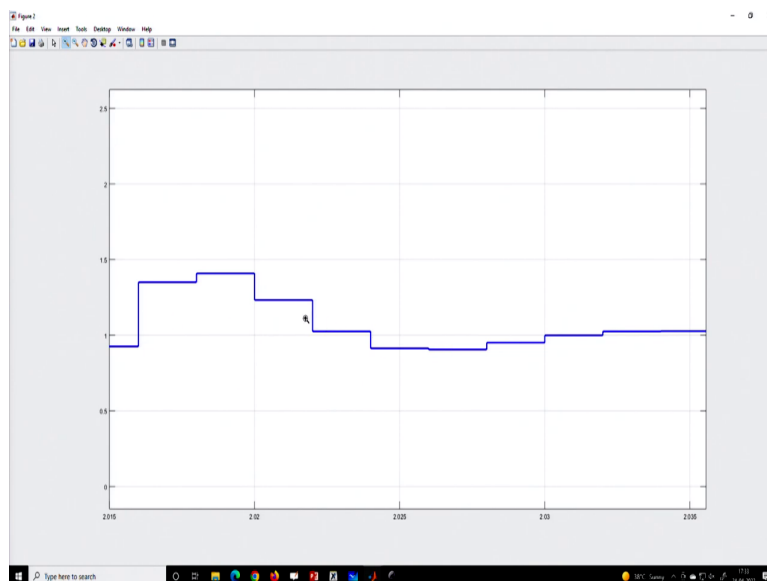
(Refer Slide Time: 06:26)



(Refer Slide Time: 06:40)



(Refer Slide Time: 06:51)



And I want to show that yes, it is coming and if I grade it yeah. So, you can see these are getting updated according to the sample; that means, these are every switching edge; that means, edge and every switching edge it is getting updated ok. That means what we understood that right now we are not incorporating any delay; that means, we can incorporate delay right now we are not incorporating any delay. How to incorporate delay we will discuss in the subsequent lecture.

In today's lecture, we are just talking without delay, but we can add the quantization block. Because; so, far we are using the sample, but when you pass through a 2D converter either you take 8-bit ADC or 10-bit ADC. So, there should be quantization; that means, the voltage will be quantized. So, now let us go to the simulation and; so, in our simulation case study first run without quantization effect; that means, we are talking about without quantizer this is the response.

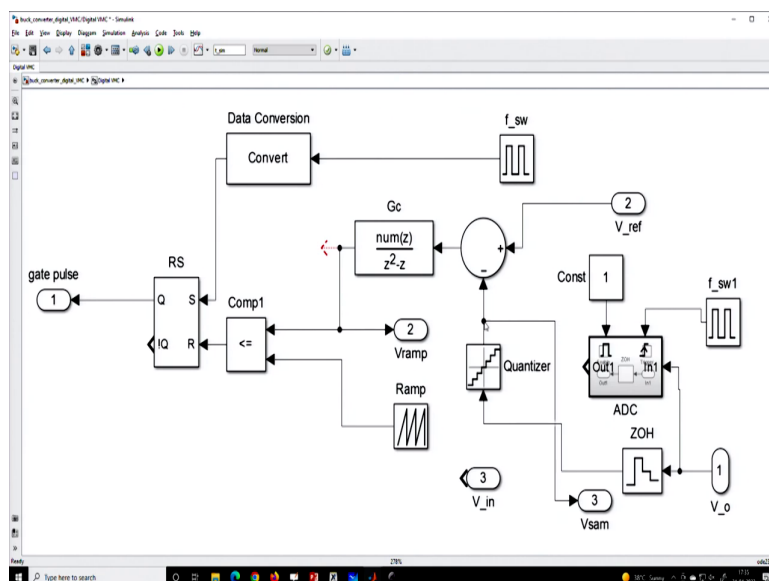
(Refer Slide Time: 07:51)

```

1- figure(1)
2-
3- plt1=subplot(2,1,1);
4- plot(t_scale_i_L,'g','Linewidth',2); hold on;
5- set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','line
6- ylabel('Inductor current (A)','FontWeight','bold','FontSize',30,'Fo
7- grid on;
8-
9- plt2=subplot(2,1,2);
10- plot(t_scale_V_o,'g','Linewidth',2); hold on;
11- plot(t_scale_Vcon,'m','Linewidth',2); hold on;
12- set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','line
13- xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontName'
14- ylabel('Output voltage (V)','FontWeight','bold','FontSize',30,'Fo
15- grid on;
16-
17- linkaxes([plt1,plt2],x)
18-
19-

```

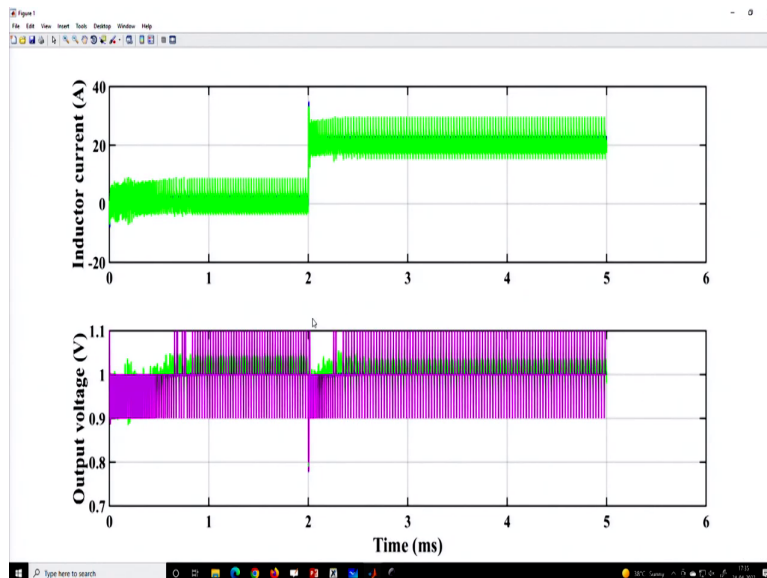
(Refer Slide Time: 08:02)



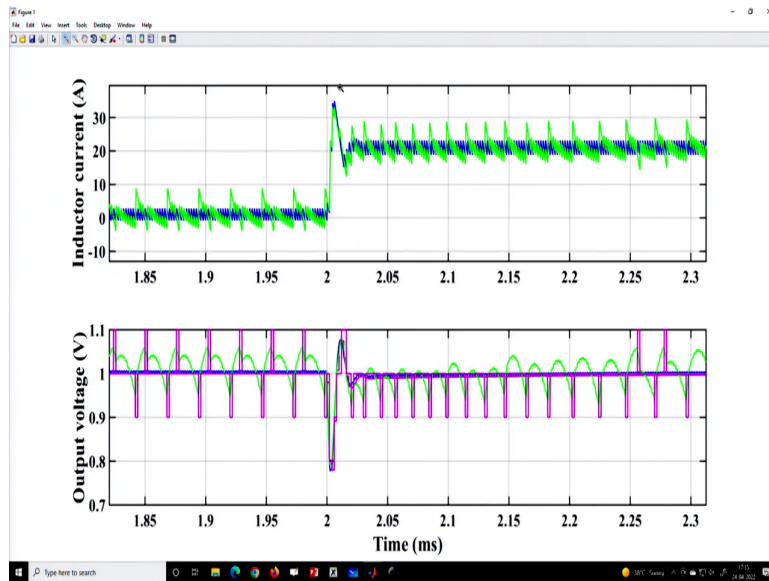
Now, next, we want to rerun it with a quantizer and we want to change the color to green color what should we change, if you go inside the digital controller? Now, you can see there is a quantizer; so, you can use this. So, I can I think you can use this quantizer before this compensator and the compensator also have a coefficient quantization ok.

So, you can whatever you use let us go and use this quantizer block here; that means, sorry yeah let us use this quantizer block. So, I am using the quantizer here and this should come here; that means, here; so, this is after quantization. So, let us run what will happen with the feedback controller with the quantization block. Now, you have added the quantizer to the feedback path.

(Refer Slide Time: 08:42)

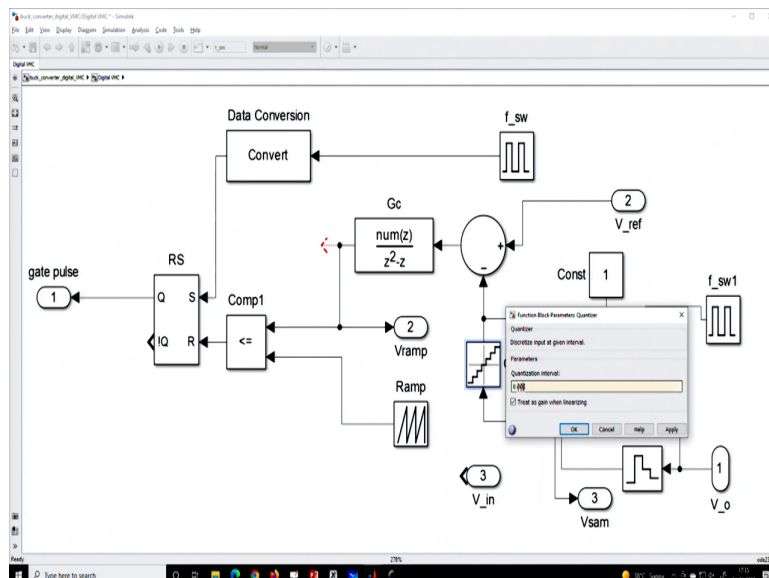


(Refer Slide Time: 08:51)



Now, since we have to take care of many things; so, it is the quantization level may be too poor and that is why it is leading to a lot of non-linear phenomena. And as a result, you can see the quantized values are different because I think I have used only 0.1 volts as a quantize; so, it should increase.

(Refer Slide Time: 09:07)

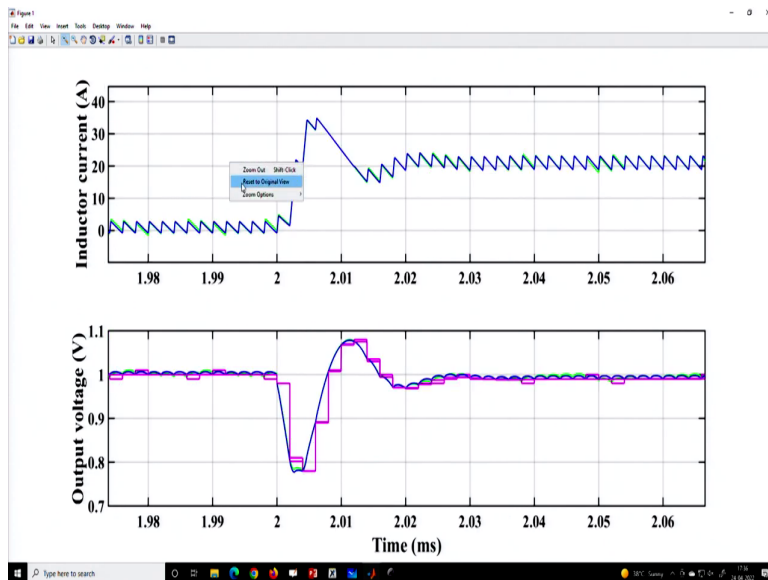


So, you should go for 10 maybe 1 mV or 10 mV quantization level. So, if you improve this quantization then it should be further improved yeah; so, let us rerun this simulation.

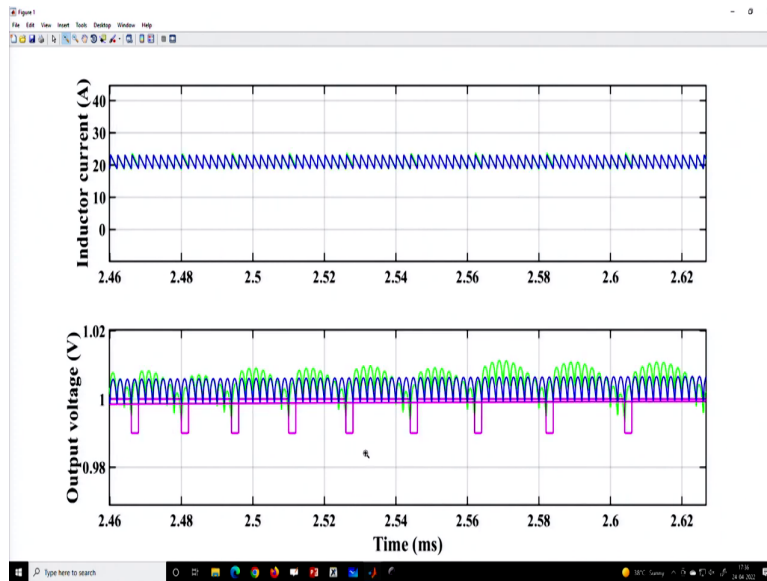
(Refer Slide Time: 09:54)

```
1 figure(1)
2
3 plt1=subplot(2,1,1);
4 plot(t_scale,i_L,'b','Linewidth', 2); hold on;
5 set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','line
6 ylabel('Inductor current (A)','FontWeight','bold','FontSize',30,'F
7 grid on;
8
9 plt2=subplot(2,1,2);
10 plot(t_scale,V_o,'m','Linewidth', 2); hold on;
11 plot(t_scale,Vcon,'m','Linewidth', 2); hold on;
12 set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','line
13 xlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontName'
14 ylabel('Output voltage (V)','FontWeight','bold','FontSize',30,'F
15 grid on;
16
17 linkaxes([plt1,plt2],'x')
```

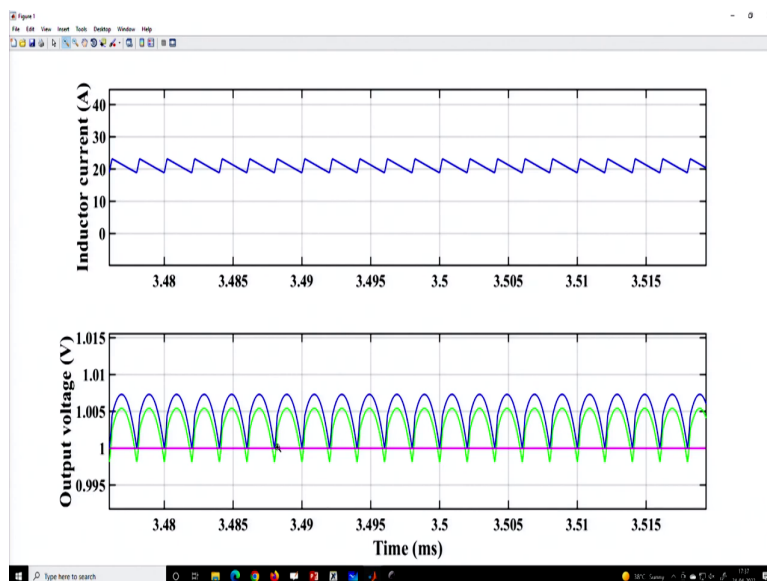
(Refer Slide Time: 10:01)



(Refer Slide Time: 10:21)



(Refer Slide Time: 10:43)



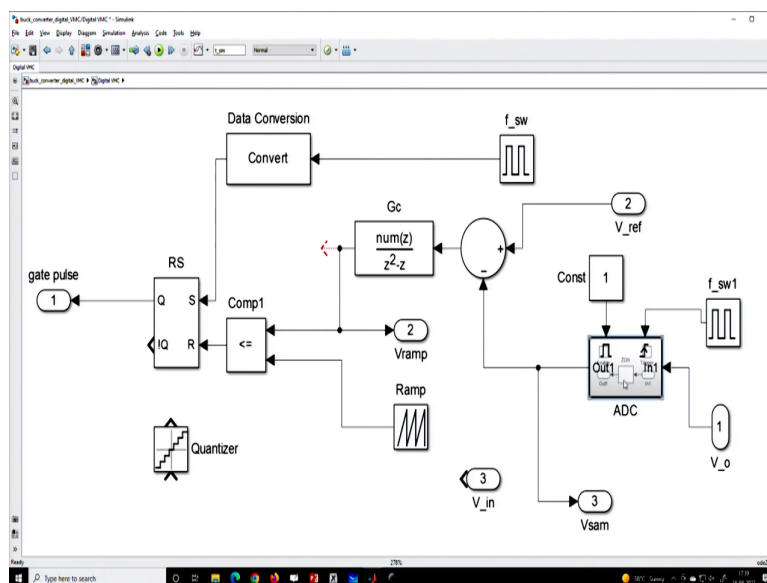
So, one with quantization another without quantization, and the quantization we have taken 10 milli volt. So, we will see the difference in the blue color without quantization and the response is not significantly different they are more or less the same right? But, there is something that is there. That means if you compare up to this point you will find a kind of non-linear behavior is happening for the quantized scale. The green color is the quantized one and you can see because of the magnitude quantization.

But after sufficient time, actually, now the quantized gets settled and then it reaches the final value, but there is a slight difference in their un-quantized and the quantized value because of the finite resolution. But steady state they reach without any further, but in light load conditions you see there is a periodic oscillation. So; that means, there because of the quantization you can see the effect is visible in the green; so, it looks like a non-linear behavior.

That means and sometimes we call it a limit cycle oscillation; that means, there is a slow scale oscillation coming. And we will discuss some of these aspects once you go to the actual implementation, but the point I am trying to make is that you can incorporate this quantization effect. So, the quantization will not have a very large or significant effect on the transient performance, but it might have; in fact, it would have an effect in the steady state behavior ok; so, 0; so, start with a zero-order hold.

But what is the problem here, suppose I want to shift the sampling time, here I have used a just zero-order hold and this will not enable me to decide my point of sampling. We can only set the sampling rate, but we cannot change the sampling point. Even though we use the same sampling rate because we need to accommodate the sampling delay; that means, the conversion time and the computational time; so, you need to provide some delay.

(Refer Slide Time: 12:07)



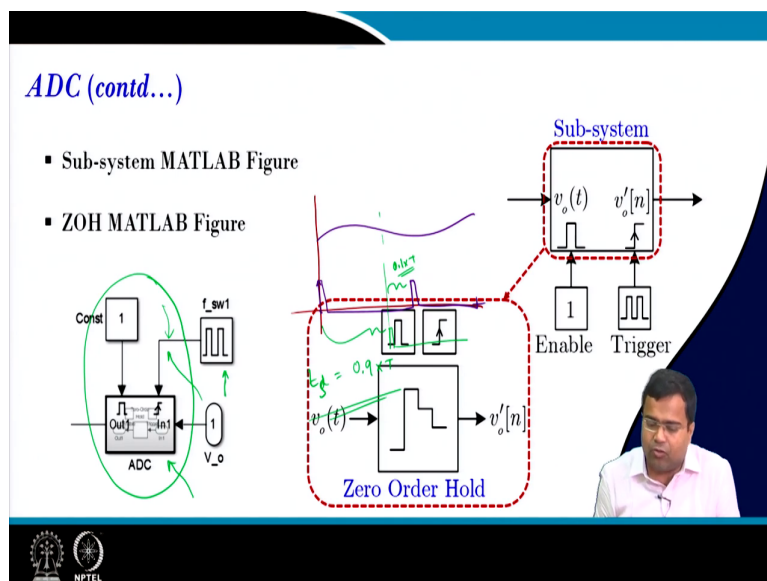
So; that means, this block will not be sufficient then we need to take some other step; so, what to do? So, right now we are not going to consider quantization; so, this is the block that

is designed to take into account this; so, I will explain what is this block. So, first, let us connect and see. So, in this block this is like an ADC, without quantization if you go inside there is zero order hold and along with that, there will be an enable and trigger block.

That means you should be able to if you enable then only it will sample otherwise it will hold. And you see outside there is a register; that means, the initial output you have to define V_0 initial to start with. But afterward, if the enable signal sorry if you enable then only the zero-order hold the sampling effect will start, if you disable the zero-order hold will not at all work. Once you enable then the sample will be updated only at the trigger edge.

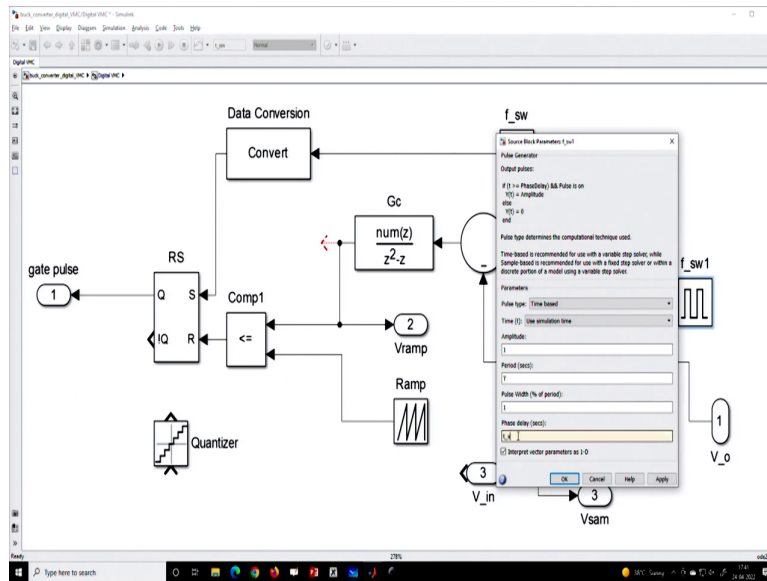
So, you cannot; that means, this zero-order hold will be only passed or it will only update the value input only at the trigger edge otherwise it will just hold it and this output like a register will just be like a storing device. So, this storing device will be updated at the trigger edge depending on whether the input signal is changing or not ok. So, this block I am explaining here; that means if you glow; so, this is what we need to move for.

(Refer Slide Time: 13:44)



So; that means, it will have enabled as well as the trigger. So, you can see in the MATLAB simulation, I have used one here; that means, it is permanently enabled we are not disabling it at all. But, suppose you know if you are going for a light load you want to stop the ADC; that means, you do not want to enable the ADC to save power you can simply take it 0; so, this feature can be linked with the practical ADC ok. Now; that means, here this clock will decide where to sample ok; so, we will take a case study ok.

(Refer Slide Time: 14:22)



(Refer Slide Time: 14:44)

```

1 - close all; clear; clc;
2
3 %% Define parameters
4 buck_parameter;
5 f_sw=1/T;
6 Vin=12; Vref=1; R=1; t_s=0*0.9*T;
7
8 %% Modulator gain
9 V_m=10; Fm=1/V_m; tau_d=T/10;
10
11 %% PID Controller Design (analog)
12 K_p=10; K_i=50000; K_d=0.1*C; t_d=T/5;
13 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14 den_con=[t_d 1 0];
15 Gc=tf(num_con,den_con);
16
17 %% PID Controller Design (digital)
18 K_pd=K_p; K_id=K_i*T; K_dd=K_d/T;
19 num_con=[K_dd+(K_p*t_d) K_p+(K_i*t_d) K_i];
20 den_con=[t_d 1 0];

```

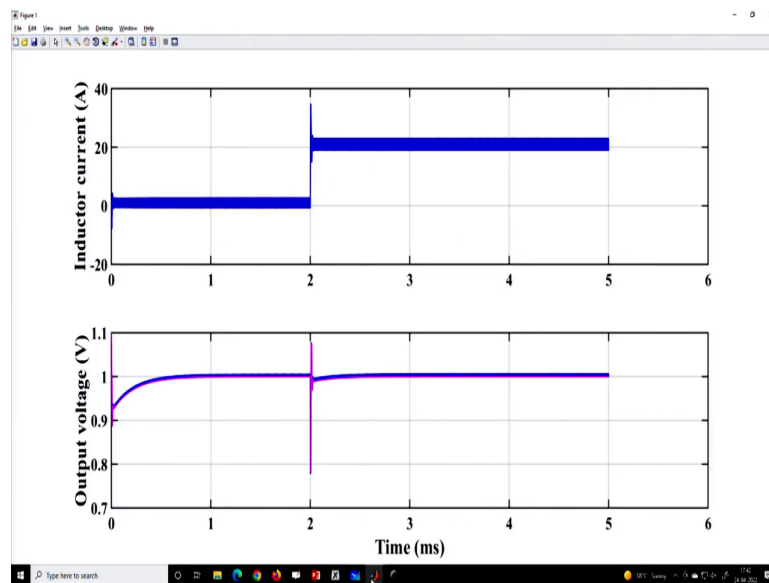
Let us say we are taking the sample before the switch start; that means, I can introduce a delay here. So, let us say we are introducing a delay and this delay I will define this in my script file; that means, I will define let us say I am defining this script file t s. And I am taking 0.9; that means, why I am taking that large value if you go to this suppose if I draw because if you go here ok.

So, suppose I am talking about a signal, and let us say we are talking about the output voltage waveform. Let us say these are the switching point, my switching point like the switching

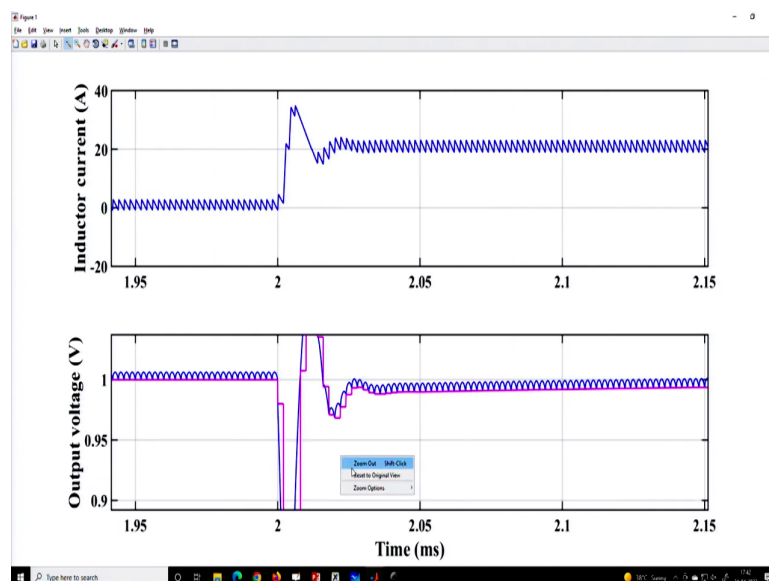
point. Now, I want to sample here; that means, I need to generate my sampling clock here; so, concerning this clock, it is delayed. So, we are setting that is why tau d we are setting or t s delay 0.9 time T; so, it is closed ok.

So, this time we will provide it is provided for ADC computation ADC conversion, and computation. So, we are providing 0.1 T as the conversion time and computation time ok; so, let me go back to the MATLAB block and if we simulate this. So, first, we will set the delay to 0, there is no quantization anymore ok; so, let us run it closes everything.

(Refer Slide Time: 16:18)

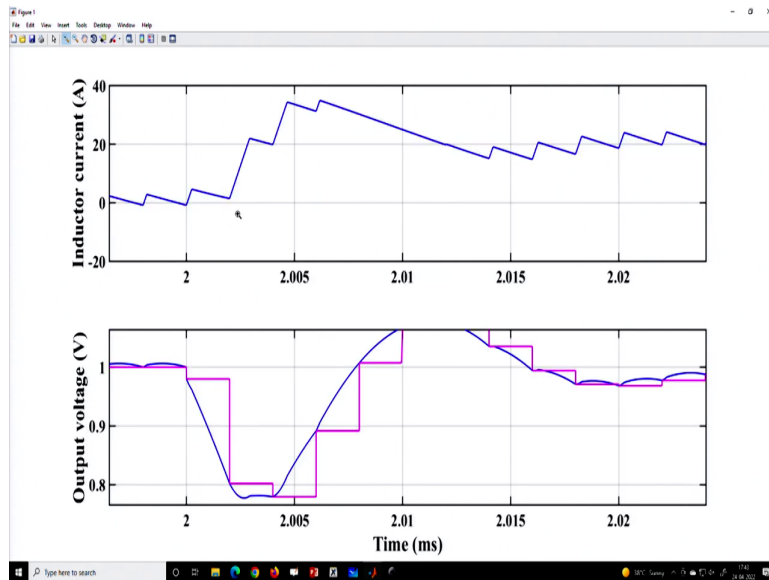


(Refer Slide Time: 16:25)



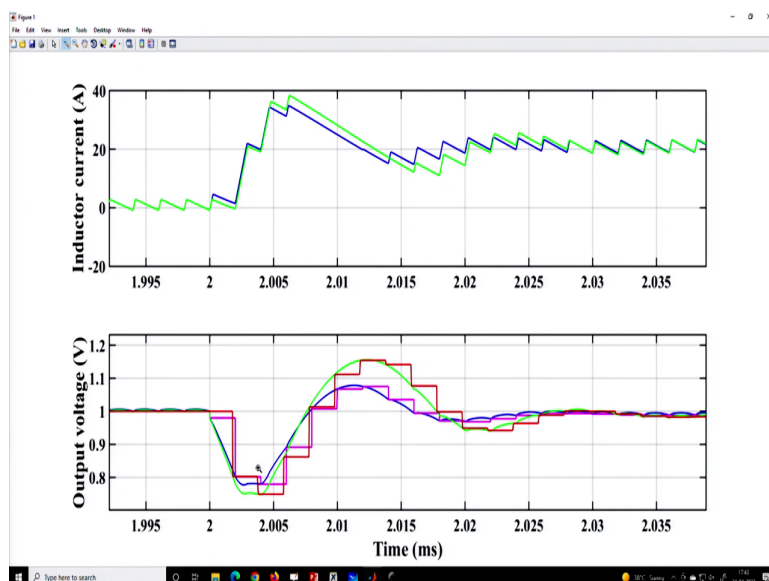
So, this is a load transient performance without delay and you can see how can you visualize whether there is no delay, because you can visualize from this waveform itself.

(Refer Slide Time: 16:31)

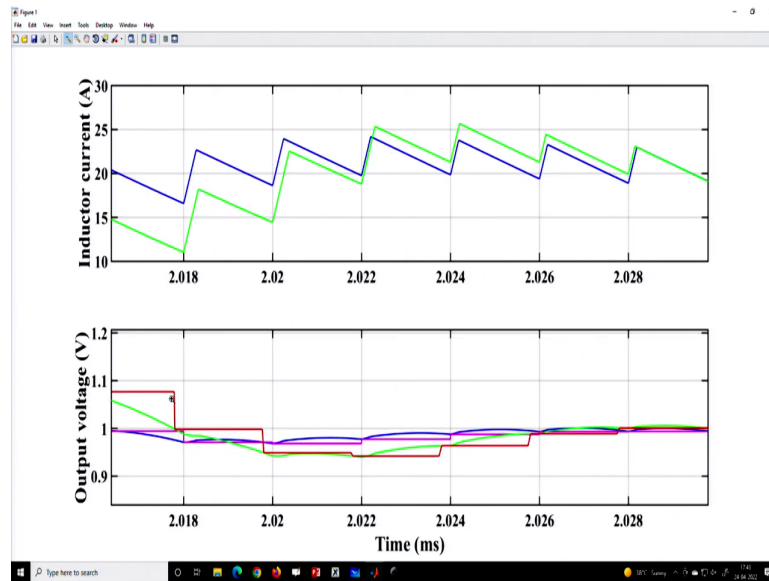


Because, if you see at every switching point the data are getting updated; that means, this is a point, data are getting upgraded at every switching instant. But now, on top of that, I want to introduce this delay and I want to give a different color. So, here I am using green color let us say green and here I am using red color as the sample box let us run, it and we want to compare; that means, with delay.

(Refer Slide Time: 17:10)



(Refer Slide Time: 17:22)



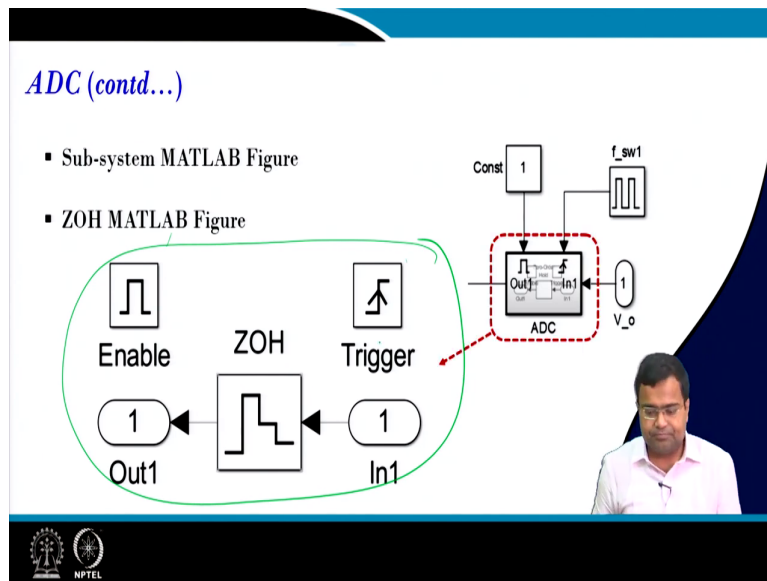
So, now; so, now, we are showing a delayed version of this you see the red one if you zoom it; that means, if you zoom this portion of this waveform, you see the sample is captured before the switching action you see this is the red one. But without delay, the magenta shows it is captured at the switching instant every switching instant, but the red one shows it is captured before the switching instant; so, we are introducing a delay.

So; that means, by using this particular block; that means, what we are discussing this block will enable us not only to set the sampling rate but also to select the point of sampling and which is of very very important that is very important. Because, if you are talking about how much delay you want to accommodate one cycle delay all these things are possible by playing with this clock ok.

Similarly, we want to go for event-based sampling because, in the previous week 2, we will talk about a lot of event-based sampling for control on time off time control. Such a clock if we cannot simply use a zero-order hold, because a zero-order hold it will ask for a sampling time and it is used for uniform sampling.

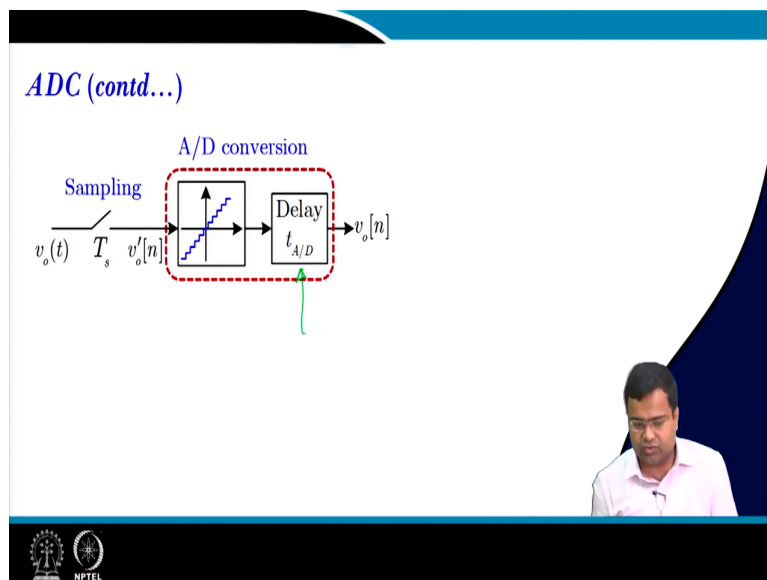
But this block can be used for event-based sampling; that means, we can use an event base sampling based on the edges; which means, the event you can take the same. And this can be the event clock ok; so; that means, inside this block what are they we have discussed?

(Refer Slide Time: 18:44)



So, this is the inside of the block I have shown, and I have used a permanent one as a enable block. So, the ADC is always on enable then the trigger clock is used to take the same.

(Refer Slide Time: 18:55)



Next so, I will discuss this delay later part because right now ok. We have already discussed delay; which means, we can introduce a delay in the sampling point; that means, right now we are sampling earlier than the switching event; so, that is also discussed.

(Refer Slide Time: 19:13)

Voltage Error

▪ MATLAB Implementation

Figure

Digital Controller

MPTEL

Now, this voltage mode control implementation is, a digital compensator; so, here is the error voltage now this is a compensator.

(Refer Slide Time: 19:21)

Digital Compensator

Digital Compensator $G_c(z)$

- P
- PI
- PID

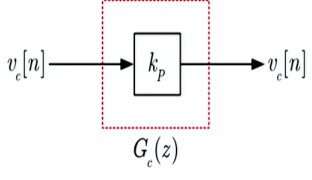


Digital Controller

MPTEL

(Refer Slide Time: 19:26)

Digital Compensator (contd...)

- Proportional

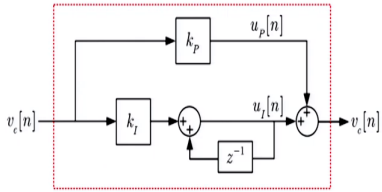

$$G_c(z) = k_p$$


So, digital compensators what are? P PI PID; so, proportional control is very simple you can simply it k P for the z transform of the compensator.



(Refer Slide Time: 19:30)

Digital Compensator (contd...)

- Proportional-Integral
- MATLAB Implementation Figure


$$G_c(z) = k_p + \frac{k_i}{1 - z^{-1}}$$

Discrete-time PI controller



So, in proportional control if you use a PI controller, this is a discrete-time PI controller it is a discrete-time PI controller ok this is a discrete-time PI controller and this is z inverse and this is a discrete-time equivalent similarly. So, the overall transfer function of the discrete-time PI controller looks like this; so, this is a discrete-time PI controller ok, and this is the overall transfer function.

(Refer Slide Time: 20:23)

Digital Compensator (contd...)

- Discrete-time PID controller

MATLAB Implementation Figure

$$G_c(z) = k_p + \frac{k_i}{1-z^{-1}} + k_d(1-z^{-1})$$

$$= k_p + \frac{k_i z}{z-1} + k_d \frac{(z-1)}{z}$$

Analog PID

$$G_c(z) = \frac{z^2(k_p + k_i + k_d) - z(2k_d + k_p) + k_d}{z(z-1)}$$

NPTEL

Next, we can also have a discrete time PID controller; so, where the equation will look like this. And in this particular lecture, we have considered this discrete-time PID controller how do we do that? If we summarize this block; that means, if we multiply by z here both numerator denominators, what we will get? We will get k P plus k i z z minus 1 plus k d z minus 1 divided by z. And if you simplify you get this transfer function z into z.

(Refer Slide Time: 21:05)

Comparator

External Ramp

Digital Controller

External Ramp

Reset

NPTEL

And in this particular case study then we can also have a comparator external ramp. So, in this case, a study we have considered if you go to MATLAB; so, this is the analog you know analog controller.

(Refer Slide Time: 21:15)

```

4- buck_parameter;
5- f_sw=1/T;
6- Vin=12; Vref=1; R=1; t_s=0.9*T;
7
8- %% Modulator gain
9- V_m=10; Fm=1/V_m; tau_d=T/10;
10
11- %% PID Controller Design (analog)
12- K_p=10; K_i=50000; K_d=0.1*C; t_d=T/5;
13- num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14- den_con=[t_d 1 0];
15- Ge=tf(num_con,den_con);
16
17- %% PID Controller Design (digital)
18- K_pd=K_p; K_i=K_i*T; K_dd=K_d/T;
19- num_con=[K_dd+(K_p*t_d) K_pd+(K_i*t_d) K_i];
20- den_con=[t_d 1 0];
21- Ge=tf(num_con,den_con);
22
23- %% Control method - option

```

So, I have chosen K_p equal to 10, K_i to be 50000, and K_d equal to 0.1 C and there is you know we know the derivative filter. Now, in discrete time the k_p remains the same so; that means, let us go back to our understanding. So, compared to analog; that means if you talk about analog PID and you know let us remove the whole thing.

(Refer Slide Time: 21:56)

Digital Compensator (contd...)

- Discrete-time PID controller

$G_c(z) = \frac{z^2(k_p + k_i + k_d) - z(2k_d + k_p) + k_d}{z(z-1)}$

MATLAB Implementation Figure
 $G_c(z) = k_p + \frac{k_i}{1-z^{-1}} + k_d(1-z^{-1})$

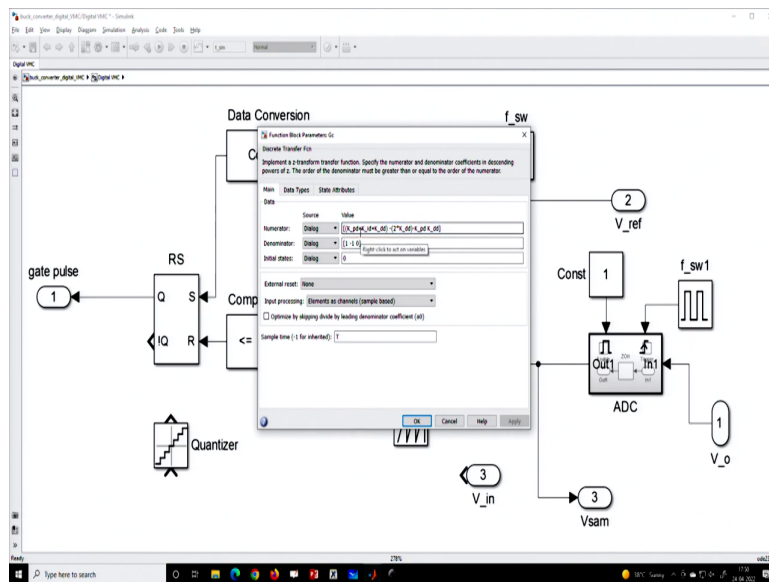
Analog PID \rightarrow Digital PID
 $K_{pa} \rightarrow K_p$
 $K_{ia} \rightarrow K_i = T_s \times K_{ie}$
 $K_{da} \rightarrow K_d = \frac{K_{de}}{T_s}$

So, if we talk about analog PID and if we talk about digital PID; so, analog PID k p will remain k P d in the digital domain. Then I will say I am using here just to avoid any confusion in analog k P I am using k P analog here it is simply k P for digital. k I analog whatever it will be in digital k I it will be sampling time multiplied by analog k i. And here the sampling time we are taking is the switching time. k derivative in analog whatever will be the value in discrete time k derivative will be k d a divided by T s and in this case, T s equal to T.

So; that means, we are just plugging this value from our analog value by suitably mapping this. And if you go to the MATLAB code the discrete-time K p is simply the same as analog K p. Discrete time integral gain is simply continuous time integral gain multiplied by the sampling time which is a switching period. And the discrete-time derivative gain is simply the continuous time derivative gain divided by T that is it.

And then this is the numerator polynomial corresponding to this equation this is a numerator polynomial and this is the denominator polynomial ok. So, that is why I am showing the numerator polynomial is K d this coefficient and denominator coefficients are this.

(Refer Slide Time: 24:01)

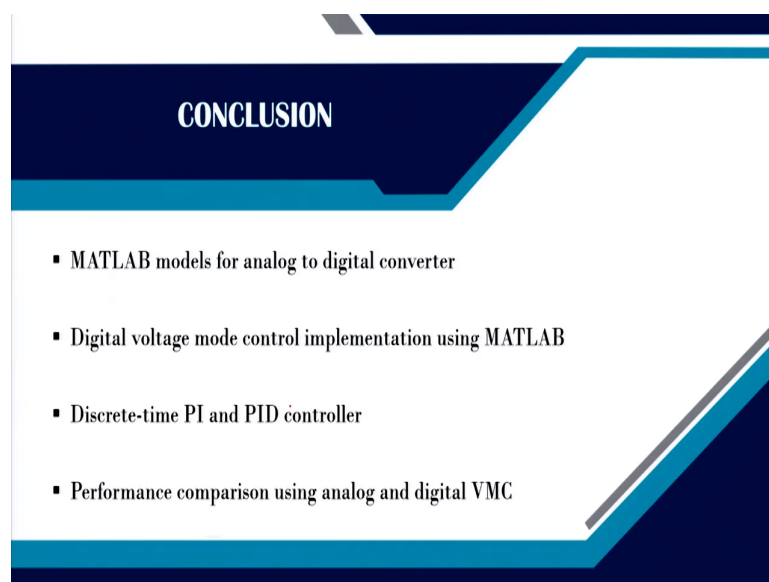


And these are going to our where it is going? We do not need it because, if you go to MATLAB, I will just show you that in this numerator we are only using this value K p d plus all these expressions whatever we got a plug-in here. And it is taking the value from the

MATLAB's key file that is it; so, with this comparison, we can simulate the digital PID controller ok.

So, then digital controllers are these blocks we have already discussed very simply. And external ramp; that means, if you use a sawtooth you can use a counter in MATLAB, I am just for sake of simplicity I am just using a sawtooth waveform simply sawtooth ok. But, you can add quantization with this sawtooth to generate this or you can add a counter to generate the number and that can be used. So, these possibilities are there and as we move forward in the subsequent lecture, we will slowly make the design more and more MATLAB model realistic ok.

(Refer Slide Time: 24:58)

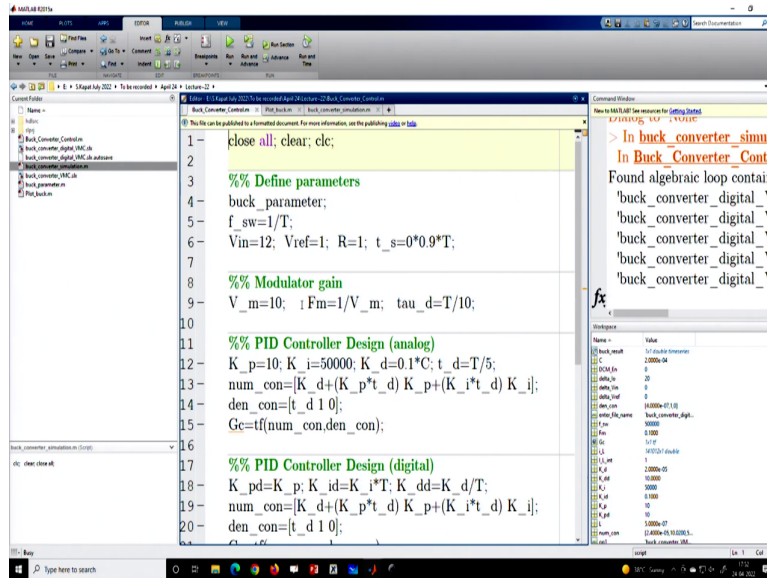


CONCLUSION

- MATLAB models for analog to digital converter
- Digital voltage mode control implementation using MATLAB
- Discrete-time PI and PID controller
- Performance comparison using analog and digital VMC

So, in summary, we have discussed MATLAB models for analog to digital converters then we have talked about digital voltage mode control implementation using MATLAB. We have discussed discrete-time PI and PID controller and then we want to compare performance using analog and digital control; so, this part is left. So, before we leave let us say; so, if we talk about analog; that means, right now we are not comparing; that means, there is no delay; so, let us say we are not delaying this signal.

(Refer Slide Time: 25:33)

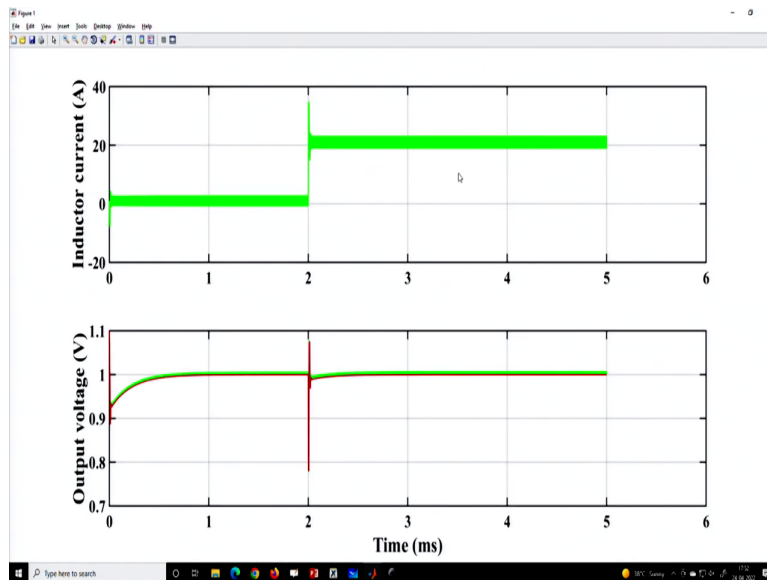


```
1 close all; clear; clc;
2
3 %% Define parameters
4 buck_parameter;
5 f_sw=1/T;
6 Vin=12; Vref=1; R=1; t_s=0*0.9*T;
7
8 %% Modulator gain
9 V_m=10; i_Fm=1/V_m; tau_d=T/10;
10
11 %% PID Controller Design (analog)
12 K_p=10; K_i=50000; K_d=0.1*C; t_d=T/5;
13 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14 den_con=[t_d 1 0];
15 Gc=tf(num_con,den_con);
16
17 %% PID Controller Design (digital)
18 K_pd=K_p; K_id=K_i*T; K_dd=K_d/T;
19 num_con=[K_dd+(K_pd*t_d) K_p+(K_id*t_d) K_id];
20 den_con=[t_d 1 0];
```

The screenshot shows a MATLAB script for a buck converter simulation. The code defines parameters like switching frequency, input voltage, and reference voltage. It then designs both analog and digital PID controllers. The digital controller parameters are derived from the analog ones. The script includes comments and variable assignments for the controller transfer functions.

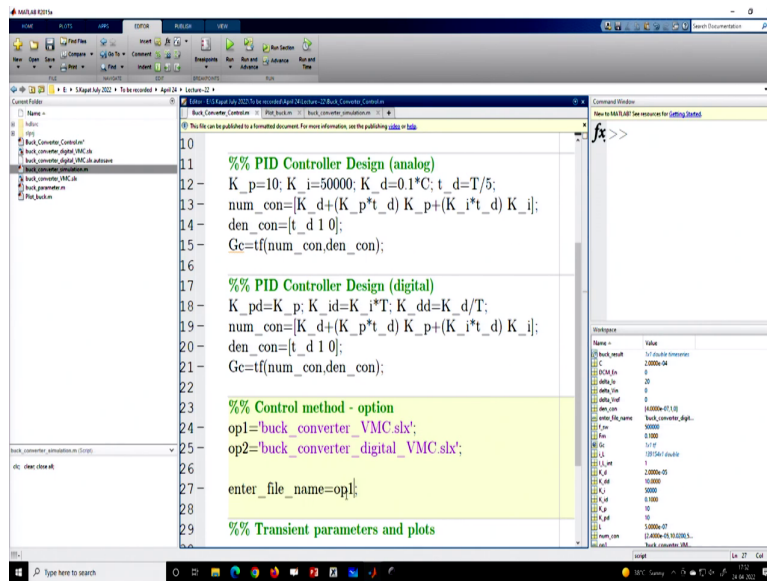
So, let us say delay there is no delay, 0 delay here multiplied by 0, and let us run for initially for this is using digital controller ok.

(Refer Slide Time: 25:51)



This is using a digital controller and this is the waveform.

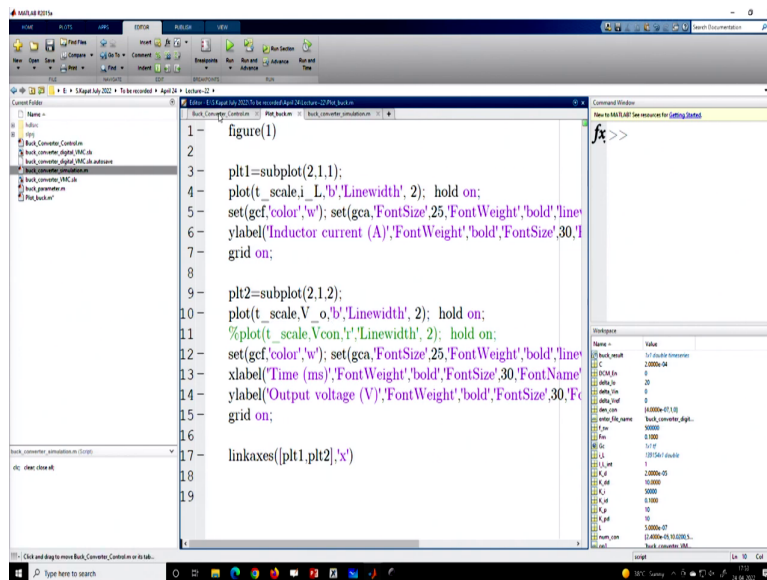
(Refer Slide Time: 26:01)



```
10
11 %% PID Controller Design (analog)
12 K_p=10; K_i=50000; K_d=0.1*T; t_d=T/5;
13 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
14 den_con=[t_d 1 0];
15 Gc=tf(num_con,den_con);
16
17 %% PID Controller Design (digital)
18 K_pd=K_p; K_id=K_i*T; K_dd=K_d/T;
19 num_con=[K_d+(K_p*t_d) K_p+(K_i*t_d) K_i];
20 den_con=[t_d 1 0];
21 Gc=tf(num_con,den_con);
22
23 %% Control method - option
24 op1='buck_converter_VMC.slx';
25 op2='buck_converter_digital_VMC.slx';
26
27 enter_file_name=op1;
28
29 %% Transient parameters and plots
```

Now, we want to show the same thing in analog so; that means, we have to change this option to 1, same voltage controller only their analog version.

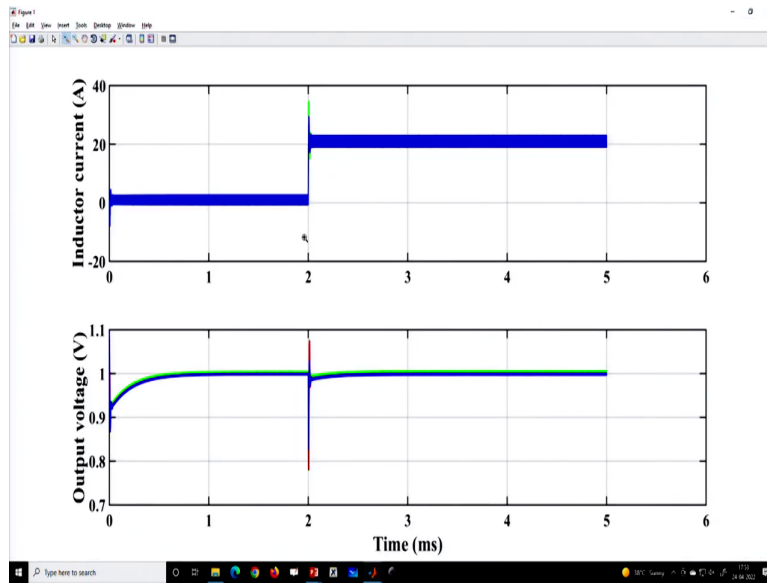
(Refer Slide Time: 26:10)



```
1 figure(1)
2
3 plt1=subplot(2,1,1);
4 plot(t_scale,i_L,'b','Linewidth',2); hold on;
5 set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','lineylabel('Inductor current (A)','FontWeight','bold','FontSize',30,'grid on;
6
7
8
9 plt2=subplot(2,1,2);
10 plot(t_scale,V_o,'b','Linewidth',2); hold on;
11 %plot(t_scale,Vcon,'r','Linewidth',2); hold on;
12 set(gcf,'color','w'); set(gca,'FontSize',25,'FontWeight','bold','linexlabel('Time (ms)','FontWeight','bold','FontSize',30,'FontName'ylabel('Output voltage (V)','FontWeight','bold','FontSize',30,'Fgrid on;
13
14
15
16
17 linkaxes([plt1,plt2],'x')
```

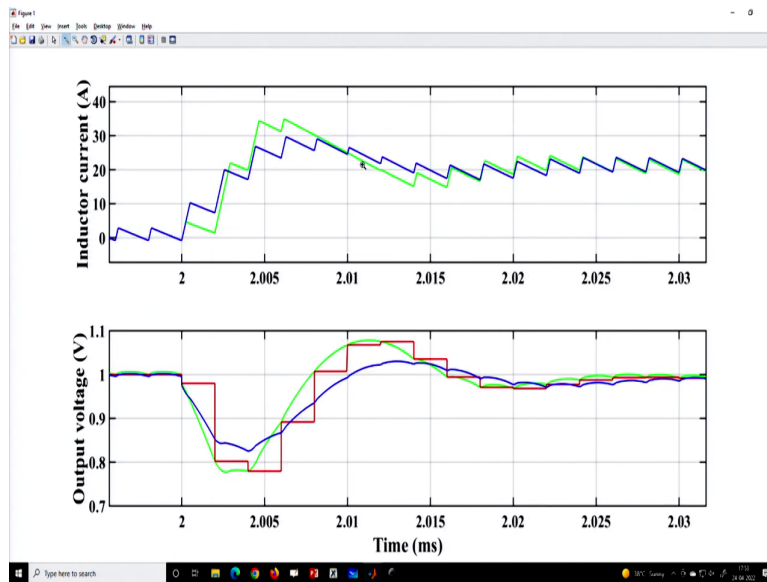
And in the plot command since there is no sample; so, we can drop this term we will change the color to blue to that is it and let us run it. We want to compare both analog continuous time and discrete time.

(Refer Slide Time: 26:26)



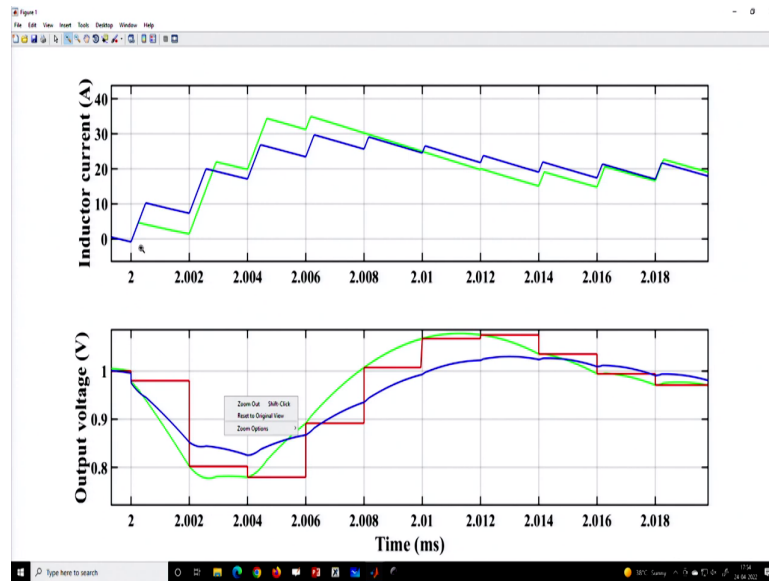
So, this is the comparison the blue one is the analog voltage mode control for the same PID controller gain.

(Refer Slide Time: 26:30)



And the green one is the digital voltage mode control because of the sampling effect you can see the performance is degraded somewhat. But it is still reasonably good and you say there is a delay and the performance degradation is due to this.

(Refer Slide Time: 26:52)



Because your transient happen ah; that means, here, but it is not fully detected. Because in analog control this undershoot is already all the time is getting upgraded into the controller which is why it is taking faster action. But, in digital control, because you are sampling at this point. So, you are unable to update the effect due to the transient and that is why it is delaying as a result you have an additional undershoot ok.

So, this is one of the problems, because we are using only one sample per cycle and that makes a little bit slowing down the transient response. So, there are many possibilities either you can increase the sampling at the transient point or you can use a hysteresis band to increase the sampling rate analog hysteresis band. So, we will discuss this in the subsequent lecture, but here we are just showing the one-to-one mapping between analog and digital control.

But in general, without any higher sampling rate, etc. You will have some delayed effect due to the zero-order hold and that is reflected because the transient is not captured at the right time. So, we have compared the performance using analog and digital voltage mode control. So, with this, I want to finish it here thank you very much.