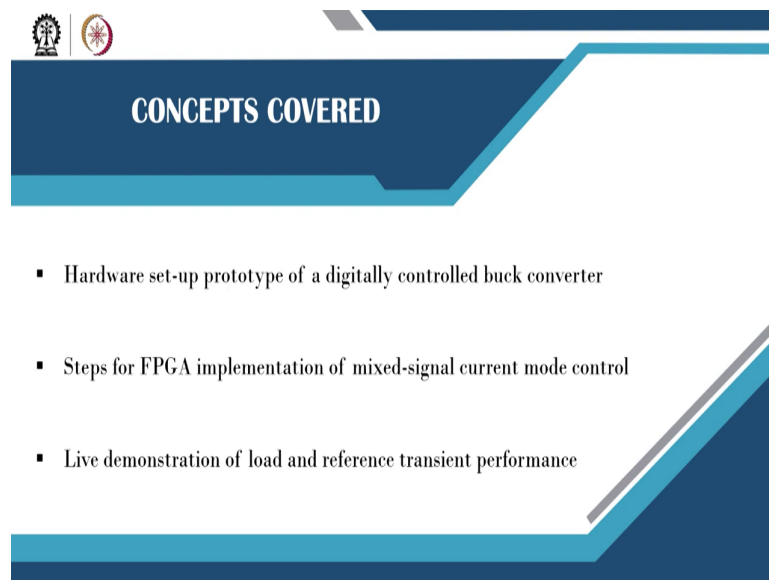


Digital Control in Switched Mode Power Converters and FPGA-based Prototyping
Dr. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Module - 10
Steps for FPGA Prototyping of Digital Voltage Mode and Current Mode Control
Lecture - 92
Steps for FPGA Implementation of Mixed-Signal Current Mode Control

Welcome back. So, previous lecture we have shown a live demo of digital voltage mode control. In this lecture, we are going to show the Step for FPGA Implementation of Mixed Signal Current Mode Control with the live demo.

(Refer Slide Time: 00:37)



The slide features a dark blue header with the text 'CONCEPTS COVERED' in white. To the left of the header are two small circular logos. The main content area is white with three bullet points. The slide is decorated with blue and grey geometric shapes on the right side.

- Hardware set-up prototype of a digitally controlled buck converter
- Steps for FPGA implementation of mixed-signal current mode control
- Live demonstration of load and reference transient performance

So, here we will show a harder setup prototype of a digital control buck converter. Step for FPGA implementation of mixed-signal current mode control and followed by a live demo of load and reference transient performance.

(Refer Slide Time: 00:49)

FPGA-based Mixed-Signal Current Mode Control Implementation

Mixed-signal current mode control

Test set-up of the prototype

NPTEL Online Certification Course
IIT Kharagpur

So, we have discussed mixed signal current mode control in lecture numbers 75, 76, 77, and 78 all 4 lectures we have discussed in sufficient detail. These are test prototypes of the buck converter and there is a prototype of the power converter, and buck converter signal conditioning board, and below we will be using FPGA.

(Refer Slide Time: 01:12)

Demonstration of Mixed-Signal CMC Buck Converter using FPGA kit

Mixed-signal current mode control

Buck Converter

Signal conditioning board

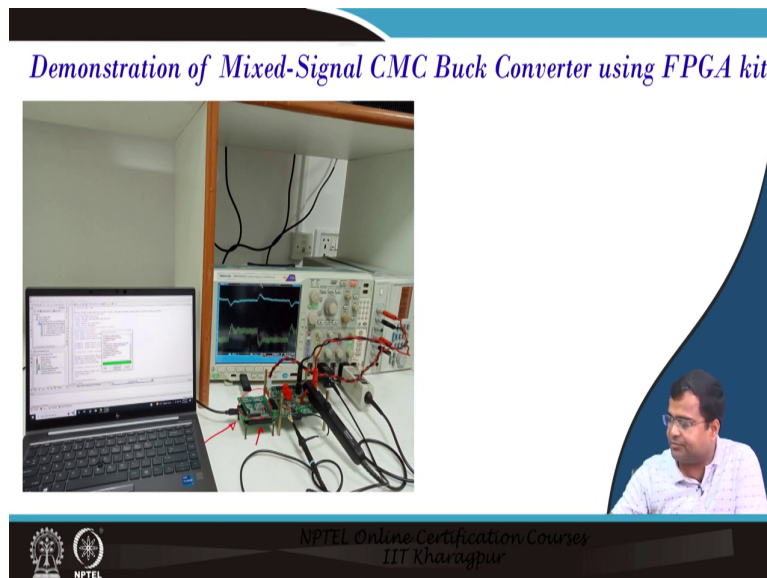
Xilinx FPGA kit

NPTEL Online Certification Course
IIT Kharagpur

And these are the buck converter then we will be using the signal conditioning board and the Xilinx FPGA. And you can see this interface and this interface we are plugged in because this

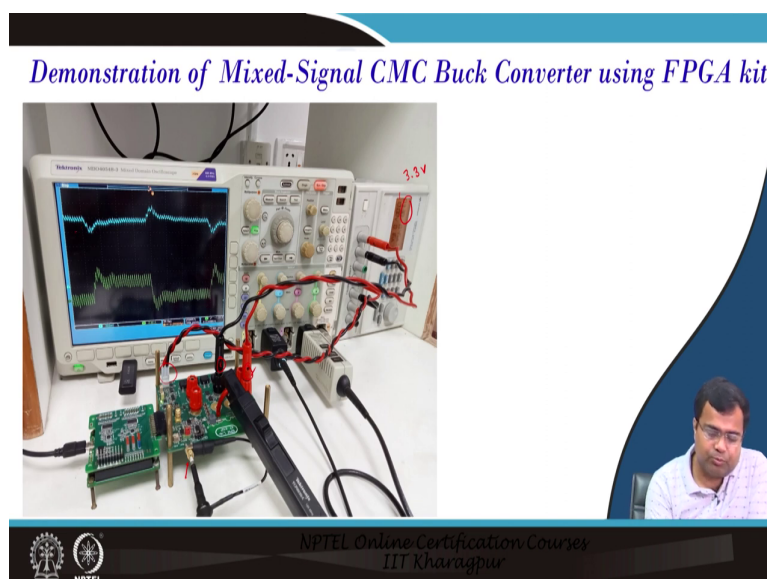
interface and this interface we are just plugging in onto that. So, that is why that board is not visible.

(Refer Slide Time: 01:31)

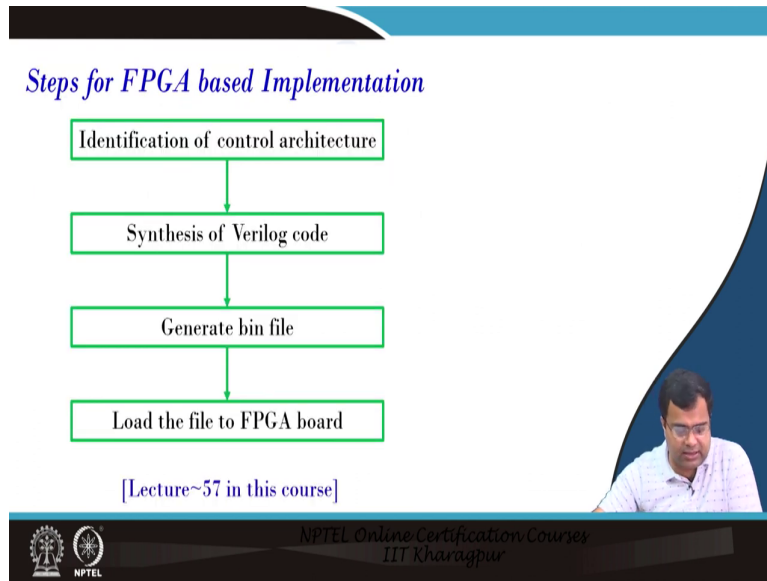


So, this is the FPGA kit, that we have demonstrated and this is connected to the signal conditioning board on the top. Just a board-to-board connector and this is a picture of the live demo picture of the hardware setup we will be going to a live demo shortly, but I just want to give you a glimpse. So, through a computer, we are programming this FPGA, and then running the whole converter.

(Refer Slide Time: 01:57)

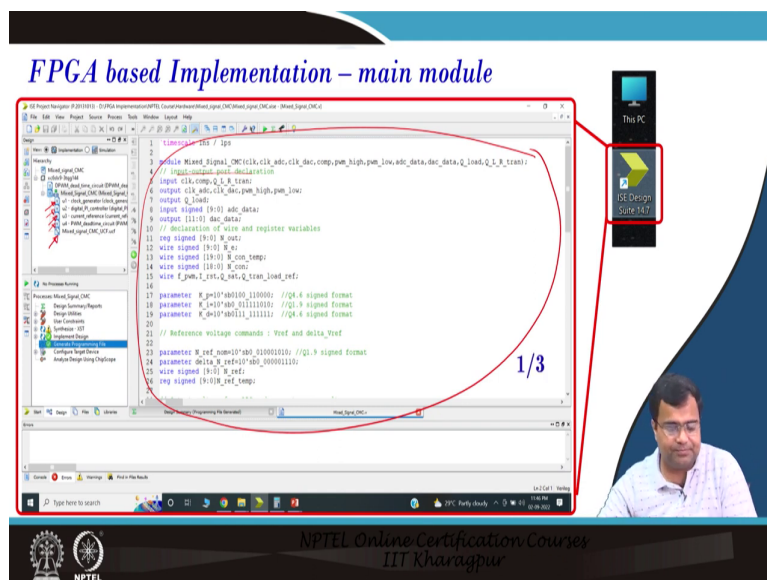


(Refer Slide Time: 03:00)



And the step for implementation is presented in lecture number 57. So, if one can refer for more detail and in today also we are going to demonstrate this thing you know this step we will demonstrate in an actual hardware prototype.

(Refer Slide Time: 03:15)



And the code is the main code mixed signal current mode control. This Verilog code has been explained I mean we have explained this Verilog code I think lecture number 76 in detail.

And we have considered different submodules for this 1 clock, 1 digital PI controller, 1 current reference, 1 PWM with dead time, and 1 with the UCF file which is to define the i o pins.

(Refer Slide Time: 03:44)

FPGA based Implementation – main module

```
50
51 assign Q_train_type=Q_s_train; // 0 for load train, 1 for ref train
52
53 always @posedge f_pwm begin
54   if (counter==M_train/2) begin
55     Q_train=1;
56     M_ref_temp=ref_num;
57     counter=counter+1;
58   end
59   else if (counter==M_train) begin
60     Q_train=0;
61     M_ref_temp=ref_num;
62     counter=0;
63   end
64   else begin
65     Q_train=0;
66     M_ref_temp=ref_num-delta_M_ref;
67     counter=counter+1;
68   end
69 end
70
71 assign M_loadM_train_type ? 0:Q_train;
72 assign M_ref=Q_train_type ? M_ref_temp:ref_num;
73
74 endmodule
```

NPTEL Online Certification Courses
IIT Kharagpur

(Refer Slide Time: 03:48)

FPGA based Implementation – main module

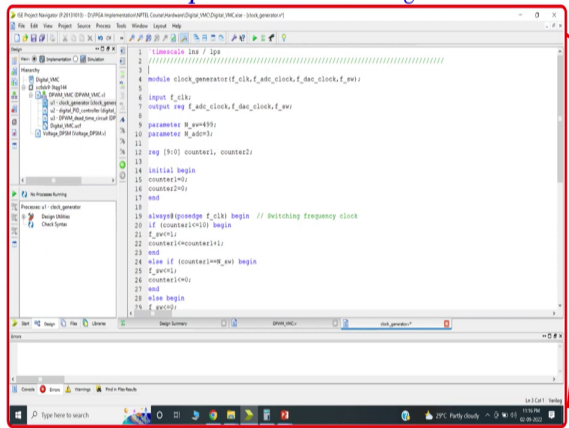
```
26 reg signed [1:0] ref_temp;
27
28 // Output voltage from ADC and generate error voltage
29 always @posedge f_pwm begin
30   M_out=abs_data;
31   M_out=abs_data;
32 end
33
34 assign M_ref=M_out;
35
36 clock_generator u0(f_clkclk, f_adc_clockclk_adc, f_dac_clockclk_dac, f_pwm);
37
38 digital_pi_controller u1(f_pwm, M_ref, M_ref, M_ref, M_ref, M_ref);
39
40 current_reference u2(M_ref, M_ref, M_ref, M_ref);
41
42 PWM_deadtime_circuit u3(f_pwm, M_ref, M_ref, M_ref, M_ref);
43
44 // Clearing transient events
45 parameter M_train=0;
46 reg [0:0] counter;
47 reg Q_train;
48 reg Q_train;
49 reg Q_train;
50
51 assign Q_train_type=Q_s_train; // 0 for load train, 1 for ref train
52
```

NPTEL Online Certification Courses
IIT Kharagpur

So, the main code is divided into 3 pages, because it's a long code, and then we have. So, this code of this main code of mixed-signal current mode control is explained already in lecture number 77, I believe 76.

(Refer Slide Time: 04:02)

FPGA based Implementation – clock generation

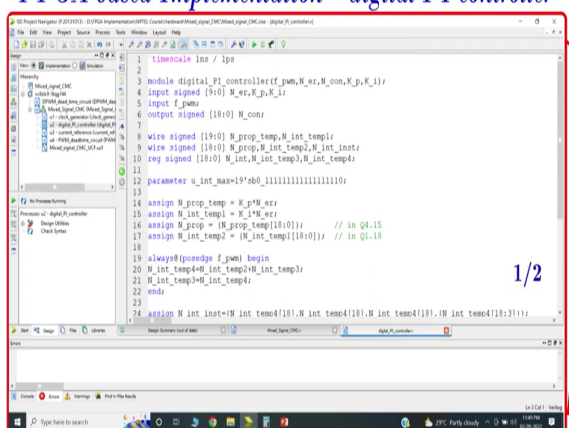


```
1 timescale 1ns / 1ps
2
3 module clock_generator(f_clk,f_adc_clock,f_dac_clock,f_wi)
4
5     input f_clk;
6     output reg f_adc_clock,f_dac_clock,f_wi;
7
8     parameter M_w=490;
9     parameter M_adcclk;
10
11     reg [9:0] counter1, counter2;
12
13     initial begin
14         counter1=0;
15         counter2=0;
16     end
17
18     always@(posedge f_clk) begin // Switching frequency clock
19         if (counter1<10) begin
20             counter1<=counter1+1;
21         end
22         else if (counter1==M_w) begin
23             counter1=0;
24             counter2<=counter2+1;
25         end
26         else begin
27             counter2<=counter2+1;
28         end
29     end
30 end
```

NPTEL Online Certification Courses
IIT Kharagpur

(Refer Slide Time: 04:16)

FPGA based Implementation – digital PI controller



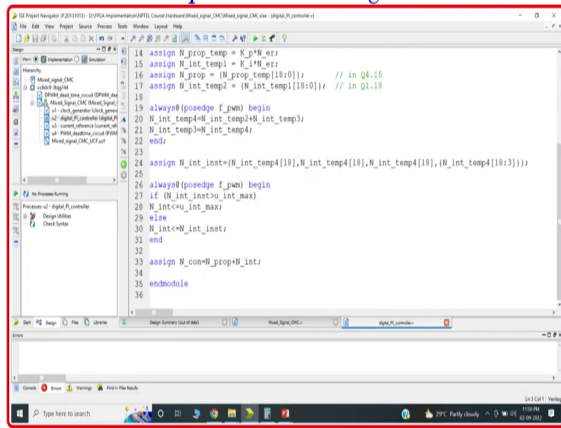
```
1 timescale 1ns / 1ps
2
3 module digital_PI_controller(f_pwm,M_er,N_con,K_p,K_i)
4
5     input signed [19:0] M_er,M_p,K_i;
6     output signed [18:0] M_con;
7
8     wire signed [19:0] M_prop,temp,M_int,temp1;
9     wire signed [18:0] M_prop,M_int,M_int1;
10    reg signed [18:0] M_int,M_int1,M_int2;
11
12    parameter u_int_max=19'b0_11111111111111111110;
13
14    assign M_prop=temp = K_p*M_er;
15    assign M_int=temp1 = K_i*M_er;
16    assign M_prop = (M_prop<0)?0:M_prop; // in Q4.15
17    assign M_int=temp1 = (M_int<0)?0:M_int; // in Q1.10
18
19    always@(posedge f_pwm) begin
20        M_int1=temp1+M_int;
21        M_int2=M_int1;
22    end
23
24    assign M_int1=(M_int1>u_int_max)?u_int_max:M_int1;
25    assign M_int2=(M_int2>u_int_max)?u_int_max:M_int2;
26 end
```

1/2

NPTEL Online Certification Courses
IIT Kharagpur

(Refer Slide Time: 04:25)

FPGA based Implementation – digital PI controller

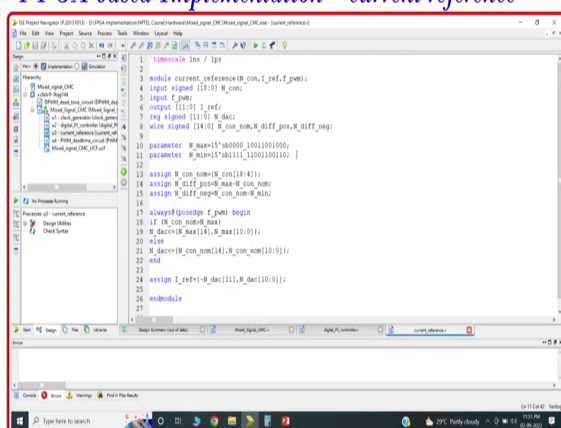


```
14 assign M_prop_temp = K_p*E_err;
15 assign M_int_temp1 = K_i*E_err;
16 assign M_prop = (M_prop_temp[18:0]); // in Q4.15
17 assign M_int_temp = (M_int_temp1[18:0]); // in Q1.18
18
19 always@(posedge f_pwm) begin
20 M_int_temp4=M_int_temp2+M_int_temp3;
21 M_int_temp3=M_int_temp4;
22 end;
23
24 assign M_int_inst=(M_int_temp4[18],M_int_temp4[18],M_int_temp4[18],M_int_temp4[18:3]);
25
26 always@(posedge f_pwm) begin
27 if (M_int_inst>=M_int_max)
28 M_int<=M_int_max;
29 else
30 M_int<=M_int_inst;
31 end
32
33 assign M_con=M_prop+M_int;
34
35 endmodule
36
```

NPTEL Online Certification Courses
IIT Kharagpur

(Refer Slide Time: 04:27)

FPGA based Implementation – current reference



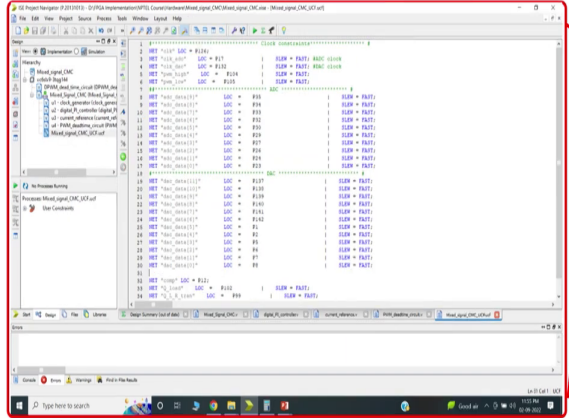
```
1 timescale 1ns / 1ps
2
3 module current_reference(M_con,I_ref,I_pwm);
4 input signed [10:0] M_con;
5 input f_pwm;
6 output [10:0] I_ref;
7 reg signed [11:0] M_dac;
8 wire signed [14:0] M_con_nom,M_diff_pos,M_diff_neg;
9
10 parameter M_max=15'd8000;1001001000;
11 parameter M_min=15'd11;100101010;
12
13 assign M_con_nom=(M_con[14]);
14 assign M_diff_pos=M_con-M_min;
15 assign M_diff_neg=M_con-M_max;
16
17 always@(posedge f_pwm) begin
18 if (M_con_nom<M_max)
19 M_dac=(M_max[4],M_max[10:0]);
20 else
21 M_dac=(M_con_nom[14],M_con_nom[10:0]);
22 end
23
24 assign I_ref=(M_dac[11],M_dac[10:0]);
25
26 endmodule
27
```

NPTEL Online Certification Courses
IIT Kharagpur

And then the clock generator circuit we have presented both for digital and current mode and voltage mode control for both in lecture numbers 72 and 76. So, I am not going to discuss this again. Digital pi controller I think we have demonstrated in lecture number 77 we discussed. So, we are not going to discuss this, but this is just the screenshot and then the current reference generator generation we have discussed in lecture number I think it is lecture number 76 that we have discussed.

(Refer Slide Time: 05:19)

FPGA based Implementation – Programming file



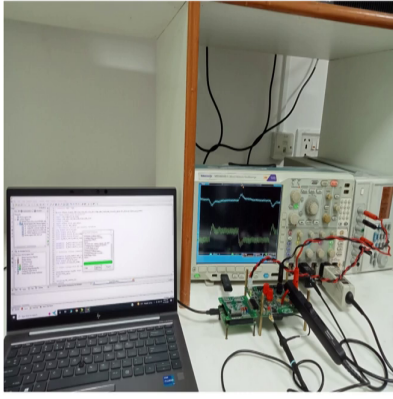
The screenshot shows a software development environment with a code editor containing Verilog HDL code for an FPGA. The code includes module declarations and logic for a mixed-signal CMC buck converter. A red box highlights a 'Program' button in the top right corner of the IDE window. A small inset image shows a person's face in the bottom right corner of the slide.

NPTEL Online Certification Courses
IIT Kharagpur

And then yeah. So, this is enabled sorry this is a write file. I think the earlier one was not the written file. So, this is the right file programmable file, because only what was disabled is now enabled for this con.

(Refer Slide Time: 05:36)

Live Demo of FPGA based Mixed-Signal CMC Buck Converter



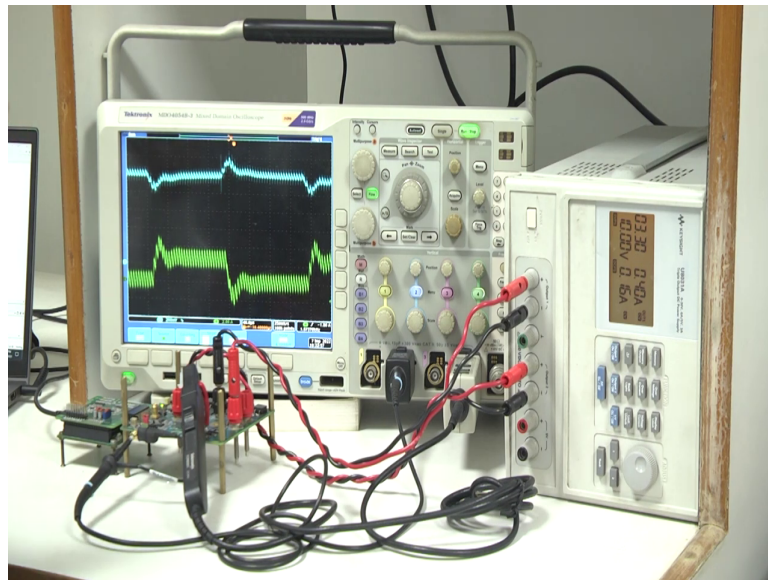
The photograph shows a laboratory setup for a live demonstration. A laptop on the left displays a software interface. In the center, a green printed circuit board (PCB) is populated with various electronic components, including an FPGA chip. To the right, a digital oscilloscope displays a waveform on its screen. A person's face is visible in the bottom right corner of the slide.

NPTEL Online Certification Courses
IIT Kharagpur

And next this is what we are going to demonstrate now; that means, this is a picture of the prototype that we are going to consider. So, now, we are going to the live demonstration. So, now, we are going to demonstrate mixed signal current mode control. This is a part of this lecture that we have already shown using a screenshot of what are the steps you know, first of

all, we discuss in detail the Verilog code for implementing mixed signal current mode control in 8th the ek, and in this lecture, we have demonstrated what is the step that user constant file? The main module sub module how to instantiate. Then we have shown how to dump this code into our FPGA.

(Refer Slide Time: 06:21)



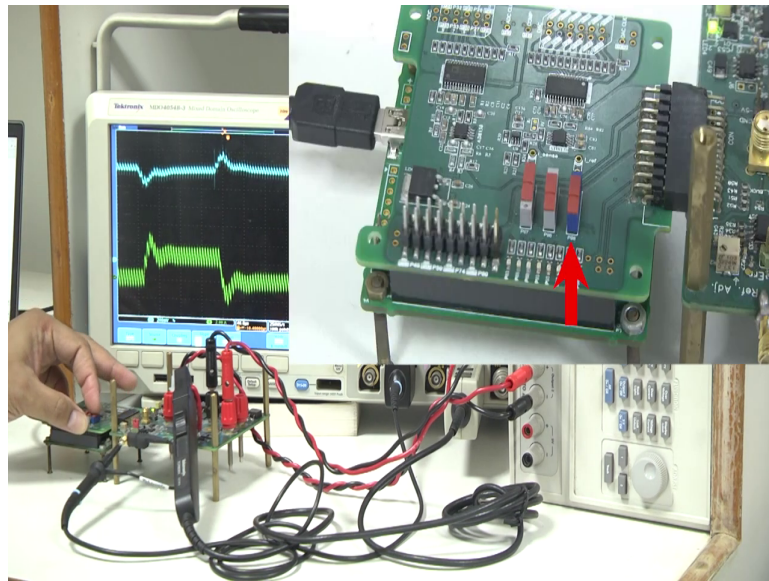
So, I am not going to repeat that it has already been in the dump. Now, we are going directly to the hardware. So, this code is running. Now, we are turning on the power supply 3.3 volt is the input and we are making first let us say we are making load transient. So, this is a load transient where again we are changing the load current from, yeah.

So, the load current was initially the resistance was 13.5 ohm now we have to make a load step transient. How? We have added the parallel resistance is 0.33 ohm with the 13.5 ohm. So, effectively it was becoming like 0.32 ohm. So, this has created a load step of roughly 3 ampere. So, it is going you can see the current load step happening and because of the load step of the transient there is a voltage undershoot, and if you just stop it you can see the voltage undershoot at 3.3 volt input.

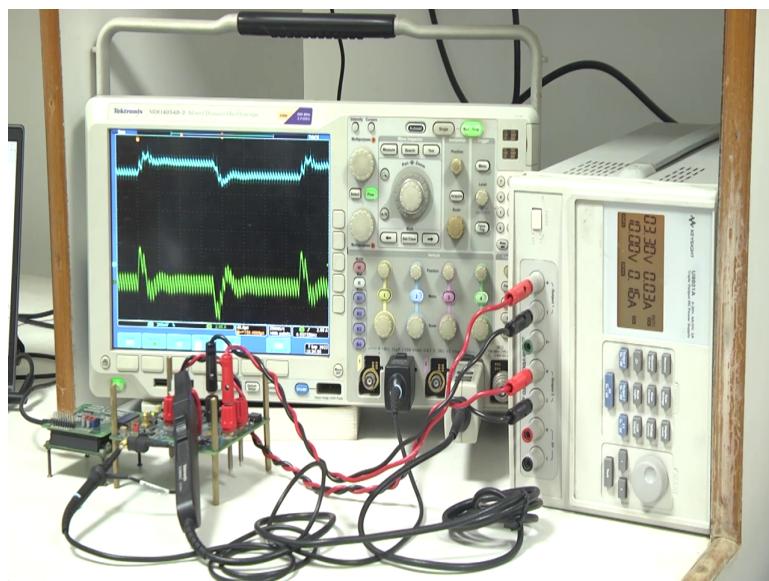
And then there is a followed load step-down transient. So, we got a reasonably good transient response, but we are going to design the current mode control in the subsequent lecture in the 11th week, but here we have considered some reasons I think good parameters of the K_P and k_i , which we have presented in the class, but we will show the step for design in the 11th week the lecture in the 11th week.

So, now, this is the load step-up transient load step transient.

(Refer Slide Time: 07:56)



(Refer Slide Time: 08:10)

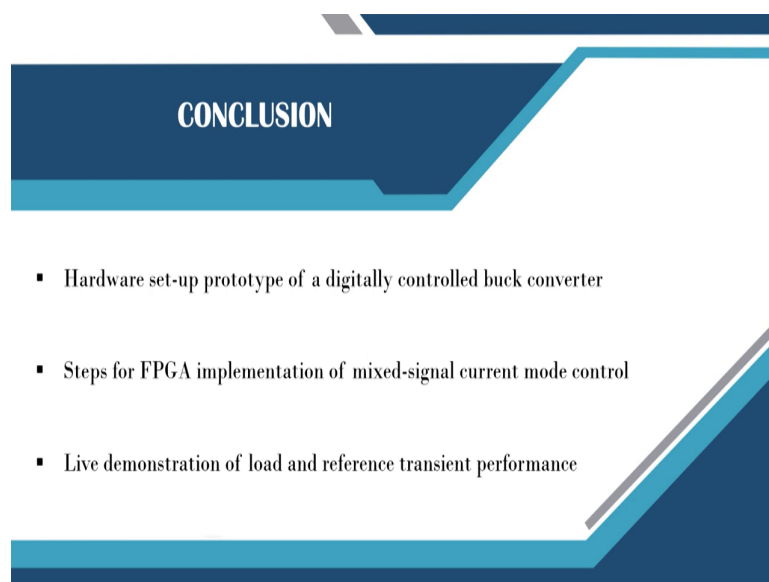


Now, we have a switch provision we have discussed. If you change this switch position it is making a reference transient. So, in the reference transient if you trigger it I want to show that this is for first it is going to step up reference transient where it is changing from 11 volt, because 1, 2, 3, 4, 5, 1 volt so; that means, the 5 division 1 division is for 0.2, you can see 0.2 volts.

So, 1 volt to 1.1 volts step-up transient followed by the step-down transient and this is the current overshoot during step-up and current undershoot during the step-down transient. So, this is the response of the current mode control since the reference is generated from the inside. So, this has I mean there is no excess delay. So, it responds almost immediately, and because of this, you know finite output impedance, because we cannot make the closed loop faster than one-fifth or one-sixth of the switching frequency closed-loop bandwidth because we also discussed the small signal validity.

So, this is what about the mixed signal current mode control, now we are going back to our class.

(Refer Slide Time: 09:07)



CONCLUSION

- Hardware set-up prototype of a digitally controlled buck converter
- Steps for FPGA implementation of mixed-signal current mode control
- Live demonstration of load and reference transient performance

So, just now we have completed we have shown the live demo of the mixed signal current mode control. Where we have discussed the harder setup prototype of a digitally controlled converter, we have demonstrated steps for FPGA implementation through a live demonstration and we have seen the load and reference transient performance. That is it for today.

Thank you very much.