Digital Control in Switched Mode Power Converters and FPGA-based Prototyping

Dr. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

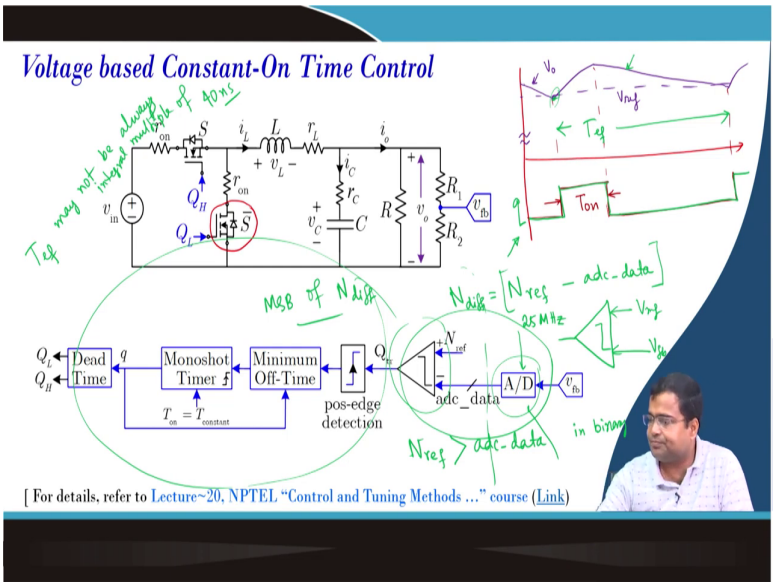
Module - 10
Steps for FPGA Prototyping of Digital Voltage Mode and Current Mode Control
Lecture - 95
Top-Down Design Methodology of Constant On/Off-Time Control

Welcome. In this lecture, we are going to discuss Top Down Design Methodology of Constant On-Off Time Control.

(Refer Slide Time: 00:31)

So, here we will first talk about we will try to understand the voltage-based constant on-time digital control and the top-down design methodology of constant on-time digital voltage mode control.

(Refer Slide Time: 00:43)



So, here if we talk about voltage mode control constant on time. So, this is an actual converter and you know if we go for DCM then we can replace it with turn off the switch then the body (Refer time: 00:56) will act like you know conventional. It will come into the picture then it will become a conventional buck converter.

So, here is what we are doing in this voltage-based architecture. If we go to the original concept; that means, if we go for you know typical diagram. Let us say we are talking about the output voltage. This is our reference voltage. So, whenever the output voltage falls below, then it triggers this constant on time when it falls and it triggers; that means if the output voltage is V0, and this is our Vref. Whenever this intersects this point then what does it do this modulator is turned on and it will be on for a fixed duration which is our constant on time and we call it a Ton.

And then again it will get triggered here Ton and this pulse if we take you to know different color this is the pulse and this is our nothing, but the gate signal; that means, you can say this is the Q, which is the signal and then you can put a dead time, but since we are talking about we are turning off the body diode. I mean low-side fate. So, you can just pass this signal to the high-side switch. So; that means, our controllable gate signal will be high when the output voltage will try to go below the reference voltage, and then your monoshot timer will be triggered, and so on.

So, this is what you see here we are using a digital controller. Imagine for the time being this ADC speed is very very high and we are talking about this comparison between these 2; that means, this is the digital platform. So, this ADC data; means, Nerf if Nerf is greater than adc data which is the digital version of the feedback voltage. Then this will go high. So; that means, we have discussed then you have to trigger or you have to turn on the monoshot timer.

Now, when you try to compare numbers digitally. So, this is digital in binary I would say in a binary number. So, we can simply make Nerf minus ADC data ok and if we take this difference then this difference will come it MSB of this difference; that means, the MSB of N difference. This will give you the sign if the MSB is 0; that means, your Nerf is greater than adc data then it will trigger the monoshot timer and as long as the MSB of the N difference is 1; that means, your ADC data is higher; that means, your output voltage is above the reference voltage.

So; that means, though it is a comparator, in digital 2 numbers if you want to compare you can simply subtract that, and looking at the MSB you can do that. So, virtually this will look the whole system will look like an analog comparator. You can imagine that analog comparator, where you have a feedback voltage, and as if you have a reference voltage because we are only dealing with the MSB of this error.

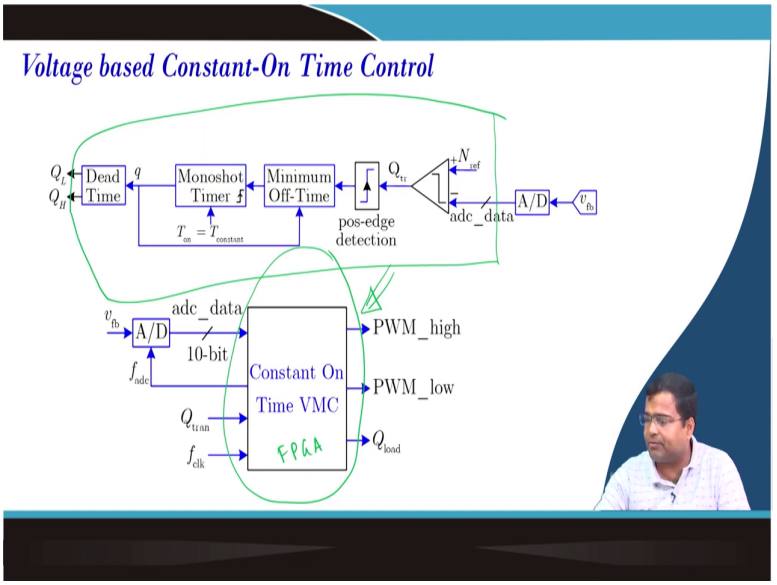
So, we are all concerned about the MSB of the ADC data. Whether because it is the difference. So, you can pass the error voltage also and take the MSB. So; that means, if this is high or low then it will decide the monoshot timer. So, just for analogous analog implementation if you take a very high amplitude ADC. It will behave like a comparator and we are only talking about the data that are different from their MSB.

But, I will show you since our sampling rate is finite we are using 25 megahertz. So, this will give a time resolution of 40 nanosecond, and since we are not and this time period which is the effective switching period under pfm tef it may not be always an integral multiple of 40 nanosecond; which means, the ADC sampling the quanta is in the time or the sampling point if they are not matching with the switching frequency then naturally there will be some you know the mismatch between the sampling time and the switching time and this will lead to some jitter in the clock.

You may not get the perfect switching frequency throughout. So, that may lie vary sometime the T effective will be slightly higher or lower. So, on average it will be T effective, but it

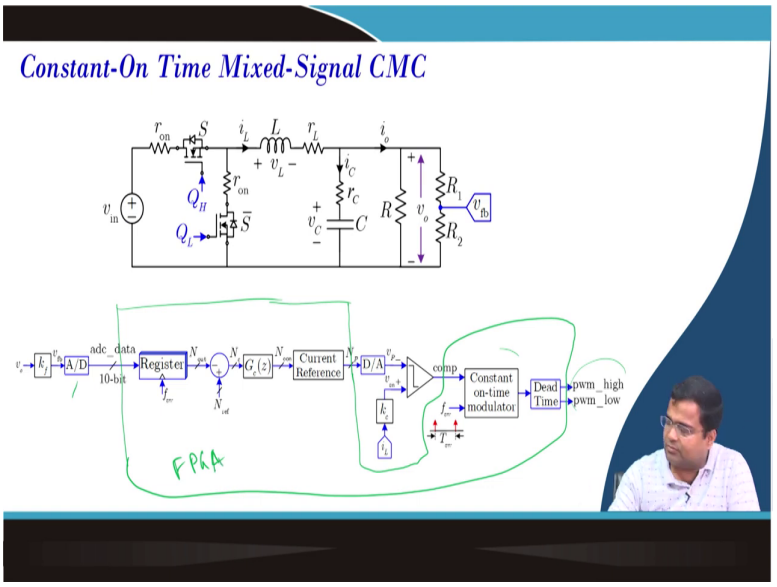
may not be exactly 2 this is because of the quantization or the sampling finite sampling rate of the ADC, but you can implement this logic digitally or you can do it in analog also, but this whole circuit in the digital domain it is digital domain and we have discussed this methodology in lecture 20 in our earlier NPTEL course.

(Refer Slide Time: 06:41)



Now, we want to design this particular thing constantly on time. So; that means, we are taking from the ADC data to this dead time this complete block is realized by this block and it will be sitting inside the FPGA. So, how do you design this block? And how do you design this monoshot timer and the off time?

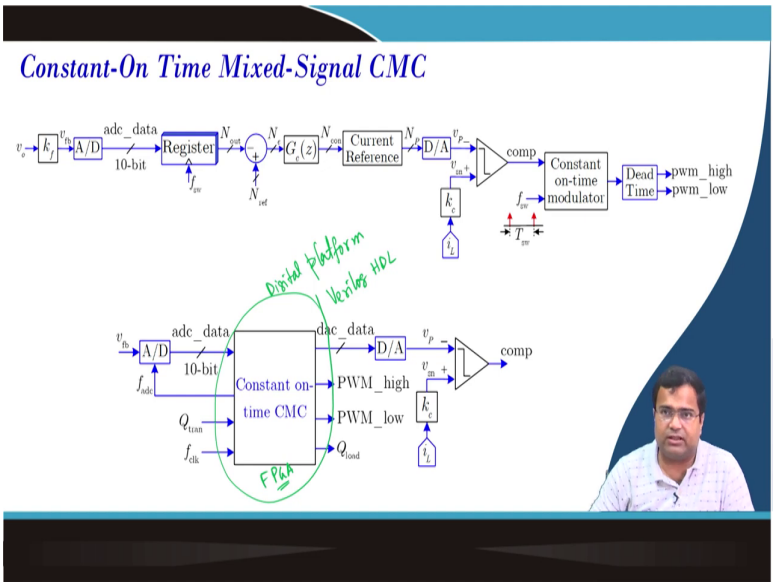
(Refer Slide Time: 07:08)



So, similarly, this is voltage based. Now, if you do the current base we know that the current base is the voltage loop is digital; that means, this starting point will be digital it comes to a register then-current reference then here again it goes to analog because it has d to a converter. It is going outside, but comparator data again come inside and this part is your FPGA because this constant modulator dead time these all can be a digital circuit.

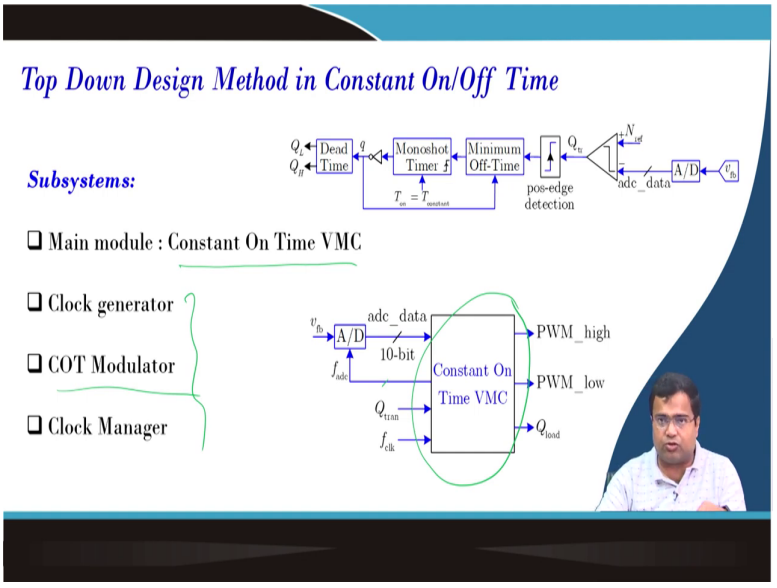
But outside it is an analog mixed signal circuit like an ADC, DSC analog signals, and the gate signals which are going out it is going to a gate drive. So, if you want we can realize this in FPGA and we will be discussing the subsequent lecture detail.

(Refer Slide Time: 07:58)



Now, here is what we can do again we have to do an FPGA-based implementation we can do. Where this guy will be sitting on the digital platform? So, it can be a microcontroller or it can be FPGA in our case we want to implement using Verilog HDL and prototype using FPGA. And the detail of this will be discussed with Verilog code in the subsequent lecture.

(Refer Slide Time: 08:28)

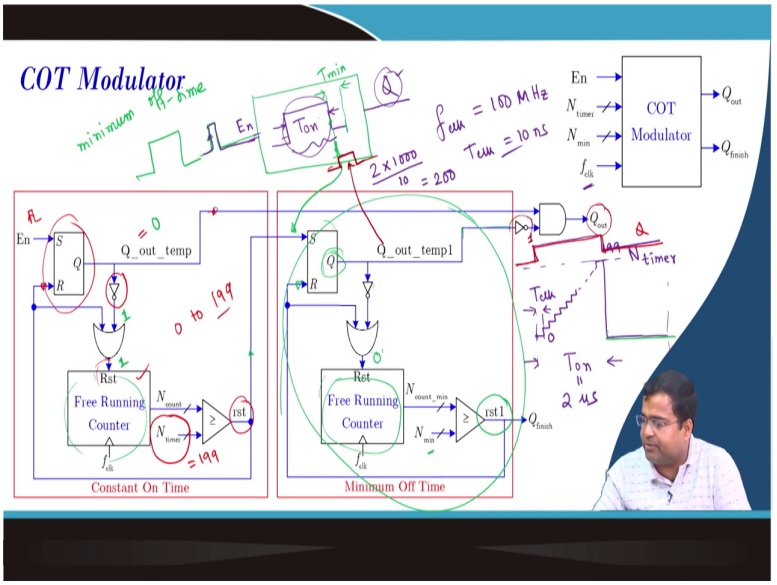


Now, we want to do a top-down design methodology how do you design this let us say we will start with the constant on-time voltage mode control. So, how to design this block in Verilog HDL? So, the main module is your constant on-time voltage mode control. It requires

a clock generator because we need to generate the clock for the ADC and we have discussed in lecture number 8 that we need to generate a clock for the ADC, DAC switching frequency clock.

Then we need a constant on-time modulator. What is that? We will discuss and we will need a clock manager what is the clock manager we will also discuss this, but these are the submodule that will come under the main module and now this goes into top down. So; that means, the main module will instantiate these are the submodule and we are going to discuss the Verilog implementation of this digital control on time voltage mode control in the subsequent lecture.

(Refer Slide Time: 09:25)



But we want to show the basic concept of a constant on-time modulator.

How does it work? So, we know that suppose we have a trigger pulse. And it is coming from a comparator this trigger pulse goes to a monoshot timer and this monoshot timer has an on-time, this is our Ton. Let us we are talking about the constant on time. So, the counter; means when you trigger this, and we are talking about this enable signal. Whenever the enable high edge comes to this clock. So, this will generate a Q signal which will be high for this fixed duration.

After this fixed duration is over because it has a timer once the Ton counting is over then Q will go low, but we also know ok. So, first, let us say constantly on-time modulator. How do

you do? So, this enables signal, and once this Ton is over. So; that means, the Ton should be implemented using a counter. And this is exactly what is a free-running counter.

Whenever your trigger pulse comes; that means, the counter will start counting; that means, we are generating this on time let us say we have this kind of sawtooth. Whenever it reaches this height it turns off and we are we want to generate this time as a Ton. So, we want to set a Nontimer it is a digital command; that means when the counter hit the upper limit or you can simply say it is a timer. Whether it is on time or off time I mean it will generate a time. So, it is a timer.

So, the timer is a set value, and what is the resolution of this clock? This clock is our Tclock; that means, then it is the time period of the high-frequency clock, and what is that our high-frequency clock is 100 megahertz. So, our Tclock will be 10 nanoseconds. So; that means, you cannot generate any on-time or off-time resolution that cannot be smaller than 10 nanoseconds, because that is the highest clock available, but yeah if you go for ic implementation on or even FPGA also has hard IP core which can generate the high-frequency clock, which can even boost the clock

It can be you know 500 megahertz or so, but we are only dealing with a 100 megahertz clock. Now, if I want to generate Ton to be let us say two microsecond. So, we need to set this counter how much? We want to generate Ton to be 2 microsecond. So, it is in the order of 10 nanoseconds. So, 2 microsecond 2 into 10 1000 divided by 10 nanosecond. It will be how much? It will be 200.

So; that means, the N timer of the counter should start from 0 and should reach 199. So; that means, 0 to 199 will give 200 again it will reset. And this is exactly what is happening, but we should remember this on-timer should not run when this guy goes low; that means, whenever the Q timer; that means, whenever he hit the upper limit; that means, during this time the Q will be high; that means, during this time I would say this Q will remain high; that means, we will use a different color Q will remain high throughout this period.

But when Q goes low, then the timer should not be incremented otherwise it will generate a false trigger. That is why you can see there is you know inverted logic; that means when this is high. So, then your reset is disabled; that means, when it is high this will be low if it is high if this is 1, this will be 0. So, this will be 0 and this is an active-high reset signal. So, reset

will be disabled. If the reset is disabled this free-running counter will start counting. And will count from 0 to 199, which is set by this timer, because this is 199.

Once this is over counting is over this reset pulse is coming and this positive trigger turns off this Q because there is an r s flip flop whenever the enable trigger comes the set is enabled because that time reset disabled the timer this Q will become 1 if the Q is 1 this will be low. So, your reset is disabled. So, the counter will start increment. When the counter hit the upper limit the reset is high that positive s come and it will turn off this counter flip flop Q becomes low. So, their Q becomes 0.

So, then this guy will be 1 and if this is 1 then what will happen; that means, I am talking about the second case when this is 0. This will be 1, this will be 1. So, reset is enabled, then no counting will happen. So, it will simply disable. This is the bottom line. So; that means, this block will generate a timer.

Now the next task we know is that every constant timer modulator should have a minimum off time. We also know the minimum off time. And this is implemented by this logic. Again this is a similar logic here whenever this reset goes high; that means, you're on time is elapsed; that means, you will get a trigger pulse and this trigger pulse is nothing but this pulse. That is a reset pulse because that counting is over.

Then it will turn on this Q, this is the other Q and now this Q is high this will be disabled. So, this will be 0, then this free-running counter will start. And this free-running counter will count till the minimum value is reached because we want to make sure is up to this minimum value. So, I am just talking about this Tmin. During this time this Q must be low irrespective of any other trigger pulse come the Q will not respond, because we need to ensure that after the on-time there should be a minimum on time. After all, this is practically there in all commercial products.

So; that means, then when it hit the upper limit this reset goes high and this reset will turn off this Q. When the Q is turned off then this will be 0 this will be Q. So, this counter will be disabled. Now, once you generate you see we are taking this output and we are inverting this logic. This means during this time; that means, in this time particular instant this guy goes high. And that is why you are talking inverting beyond this this is low; that means, during this off time this will be low as a result Q out will be low.

And otherwise other duration it is high, this is low. So, this will be high otherwise then it will simply pass the Q output. So, the Q output is the output of the monoshot timer without I mean outside the minimum duration, but when the minimum duration is enabled then Q out is simply set to 0. And this is the logic that we are going to implement.

(Refer Slide Time: 17:32)

So, in summary, we have discussed the voltage-based constant on-time control and we have discussed the top-down design methodology of constant on-time digital voltage mode control. In the subsequent lecture, we are going to discuss the Verilog implementation of this particular logic that is it for today.

Thank you very much.