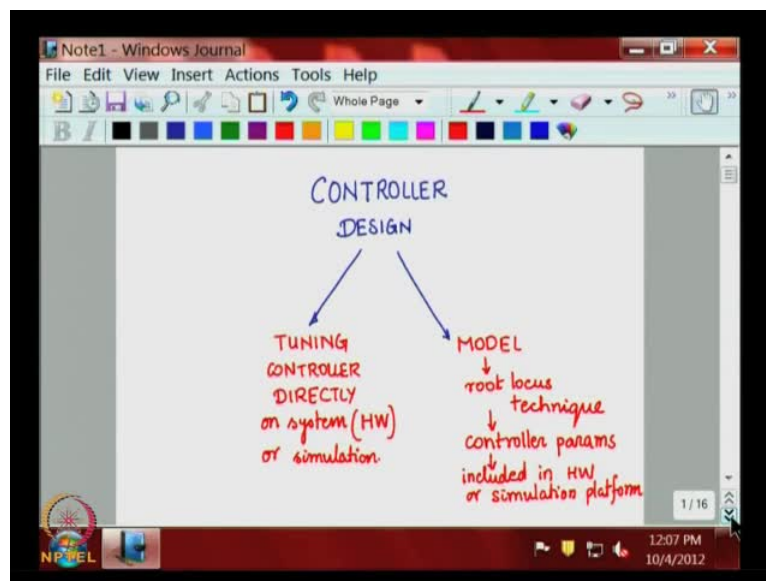


Switched Mode Power Conversion
Prof. L. Umanand
Department of Electronics System Engineering
Indian institute of science, Bangalore

Lecture - 32
Controller design – I

Good day to all of you. So, we continue from where we left off in the last class. Today we shall try to integrate the control system the PWM and the DC DC converter all together and see how the whole closed loop system is put in place. And then we shall try to have a look at how we go about choosing the controller parameters for any given system.

(Refer Slide Time: 00:58)



So, the objective of today's class is to actually have a controller which is designed to meet some performance specifications. Now, these we shall do by two methods; one is by, what should we say tuning, tuning controller directly on the system, directly on the system, so directly on a system which could be a hardware platform or it could be a simulation platform. The other method is from the model we have develop this small signal model of the DC DC converter using that model we use the root locus technique, use the root locus technique, and then from the root locus technique we arrive at controller parameters. These controller parameters are arrived at by iteratively looking at the root locus of the plant closed loop systems and these step response, and then after

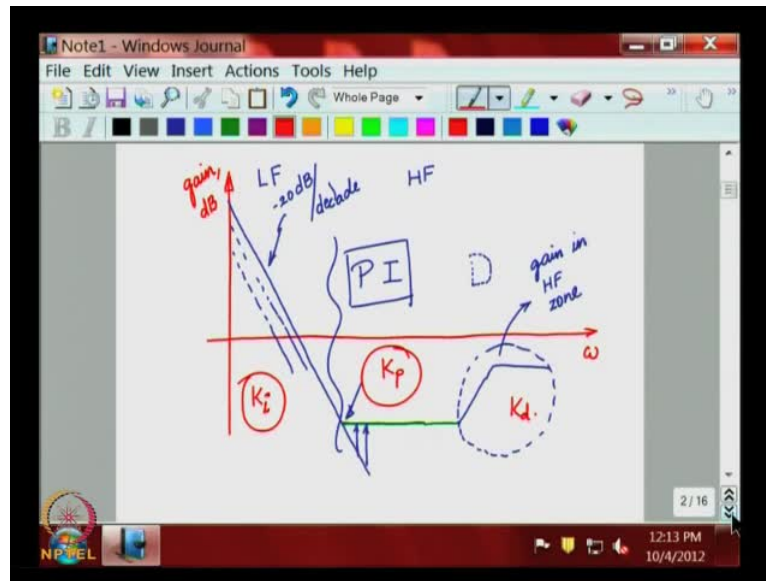
many iterations you can boil down to the appropriate value of the controller parameters. This will be populated or included in the hardware or simulation platform.

So, this is how we will try to address the issue of controller design, the tuning the controller directly by trial and error is a kind of popular method which is used by many people as does not involve a lot of maths and modelling. And it also handles many uncertainties and un-modelled dynamics, lack of over knowledge of the system all those issues are directly handled. However, this has to be done directly on the final target system or one level above that is an, the simulation platform.

Whereas, in this method mathematical modelling of the system obtaining the transfer function, from the transfer function, from the root locus in the root locus identifying, what is supposed to be the best in a closed loop pole position, and from they are identify the control parameter and then plug it into the hardware of the simulation platform and see the response.

This are the two methods this is much more formal method can be applied for are many complex systems, but we will look at both this methods. So, first tuning directly one of the more popular method cyclonic nickels method where the parameters at the K I parameters the PID parameters the propositional integral and the derivative parameters are sequentially tuned till, the response is satisfactory and then finalizing on the values appropriately depending upon what is the performance specifications. Of course, we are not going to try exactly the cycle in the cyclone nickels method, we are going to try something similar use something similar like the cycler nickels method tuning first the integrator parameter the propositional and then the derivative.

(Refer Slide Time: 06:48)



Derivative is actually if you see in over earlier class is a gain that we are trying to introduce in the high frequency zone, this is the omega axis and this scale is logarithmically distributed and this is the gain in D B remember form the old classes. We have divided this zone into the low frequency part and the high frequency part the integrator is something like this its keeps going down at minus 20 D B per decade, this is the slope of the integrator refer back to the earlier classes.

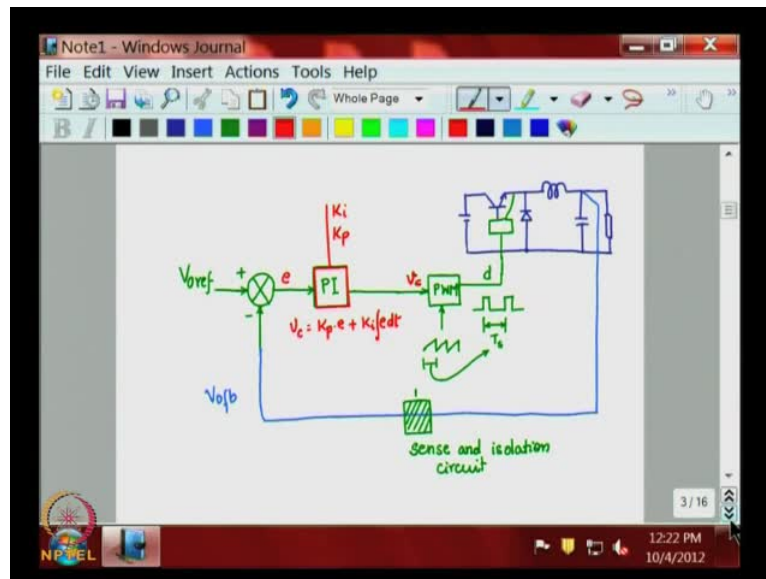
Now, the propositional part the moment you introduce the propositional part we see that its adds 0 and therefore, plus 20 D B per decade so the plus 20 D B per decade and the minus 20 D B per decade of the integrator get compensated and there is a flattening of the controller curve gain curve, and somewhere act much higher frequency. We are trying to introduce another 0 the derivative part which tense to increase the gain contribute the plus 20 D B per decade and much later it platens out because of the presence of the pole, because no derivative is ideal in the ideal sensitization is exist there has to be a pole atom much later state.

Now, this introduction of gain increase here in the high frequency zone, this is gain increase in HF zone. This is a dangerous zone in some sense because this is where noise is dominate in the high frequency zone an introducing the gain here is abytaicy and you will have to be very careful system to system you may have to tune this gain, preferably do not use the d position if a performance specification are met with the integrator and

the proportional stop that go for the d only if absolutely necessary and if you are not able to improve the performance with propositional and integral controller.

So, far most of the most of the cases just the P and I. P and I positions are sufficient the deposition you can use it only if necessary and that come in at a later stage of a design. So, we shall try to first tune the I position choosing which parallel the would you like to use and then use the propositional cut off point for, for improving the dynamics because we are trying to improve to the gain here. So, this is the approach start we are going to follow choose K i parameter first and then K p parameter next and only you required go for the K d parameter.

(Refer Slide Time: 12:11)



So, if we look at our system let us say that we have a buck converter which is something like this, so we have a switch and representing the semiconductor switch by a transistor and a diode, but remember that this transistor could be Mosfet or IGBT or depending on the power level. So, we have a buck converter here. Now, this buck converter is appropriately driven by a gate drive which drives this transistor with respect to its emitter or source the signal for this gate drive will be coming from a pulse with modulator.

So, here you will have your PWM IC which is going to give out the signals in this fashion. Where the on time can be varied switching time, switching period on time can be varied, so which basically means that we are giving a duty cycle input to the buck convertor. Now, to the PWM are two inputs one is a carrier triangle or saw tooth carrier

which basically keeps the reference for the period and other input is the control voltage which basically, modulates this carrier to give the duty cycle as we discussed in the earlier class.

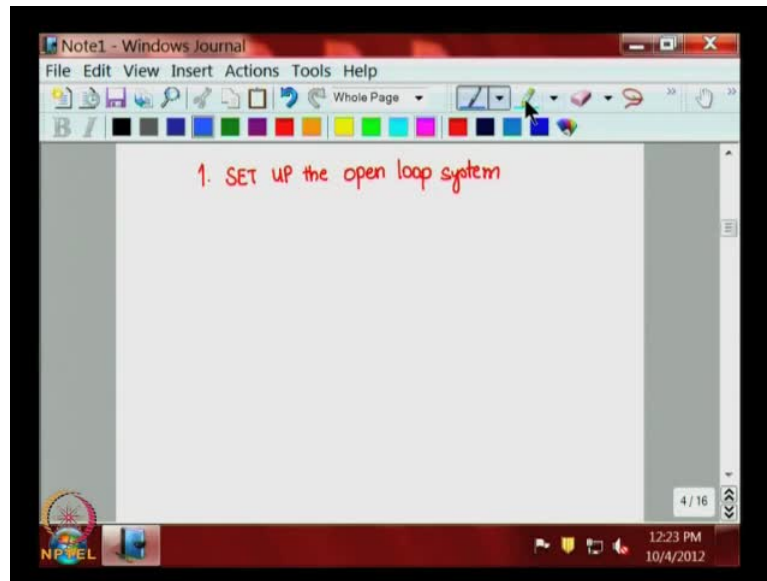
Now, the output of the controller the controller the central portion on this, this is the controller the output of which would be the control signal or the control voltage to the PWM or the pulse width modulator. Now, let us say this controller is a PI controller to start with and the input of the controller is actually coming out of a compare block which compares a reference with the feedback signal. So, we will call as the compare block. Now, let us say that we would like to control the output voltage if you are controlling the output voltage, then you provide a output voltage reference and you feed back as signal which is propositional to the output voltage. If you are controlling the inductor current then you will be setting the inductor current reference feedback into current or if you are controlling anything else that particular parameter is reference has to be set here.

So, let us say we set the reference for the output voltage, which means that you will need to feed back the output voltage sends that and bring it into the controller and this is how you feed back the output voltage v_{feedback} note that here for now. We are directly feeding back the output voltage to the compare block in practice you do not do that direct feedback it passes through another sensing circuitry, so actually you would see a block here and that block is called sense and isolation circuit.

So, for now we will assume that sense and isolation circuit there and it has input output gain of one unity. We will later an look at what is sense and isolation circuitry is, and you need to isolate this ground with the control ground to protect power and the signal ground, signal ground from the power ground and you need to sense and provide a voltage here which is propositional to the instantaneous value of the output voltage the control voltage. So, that is the job of the sense and isolate circuit this position which will involve probably up to isolator to bring in the isolation.

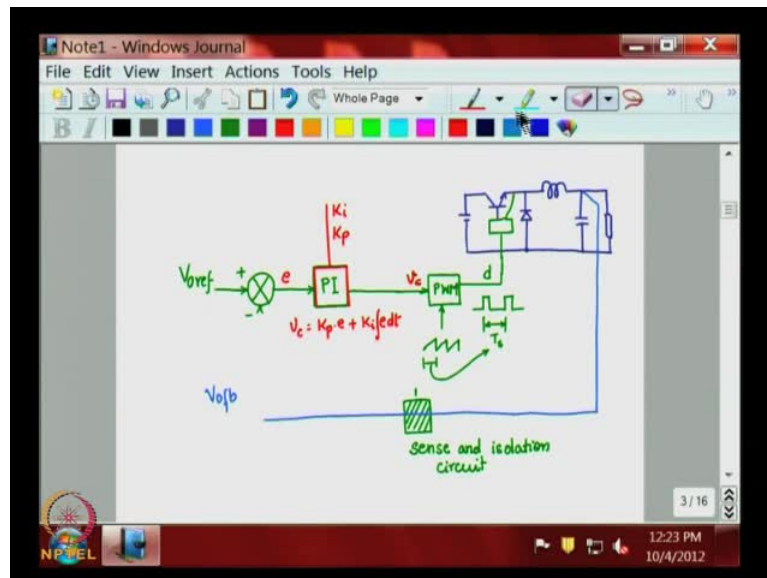
Now, our main aim is to concentrate focus on this block how are we going to tune this, now this block in its cell is having two parameters one is the K_i the integrator scaling factor and other is K_p the propositional scaling factor and we know the equation of PI which is if this is the error and this is V_c this is v_c and we know that V_c equal to K_p into e plus K_i into integral of EDT this is we want to achieve.

(Refer Slide Time: 20:54)



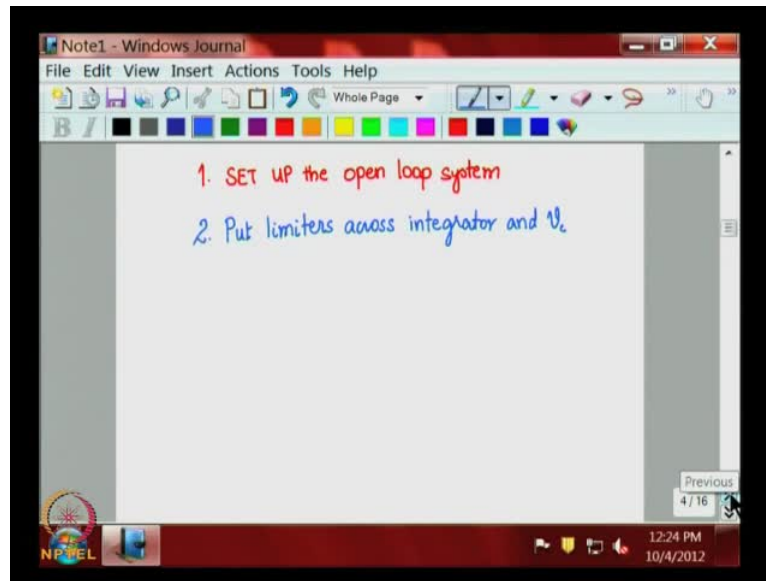
So, what we do step by step is first we set up the open loop system which is like this, the open loop system would be everything from everything from here the comparator block controller PWM and this DC DC convertor and the feedback sense and isolation up to this point.

(Refer Slide Time: 21:56)



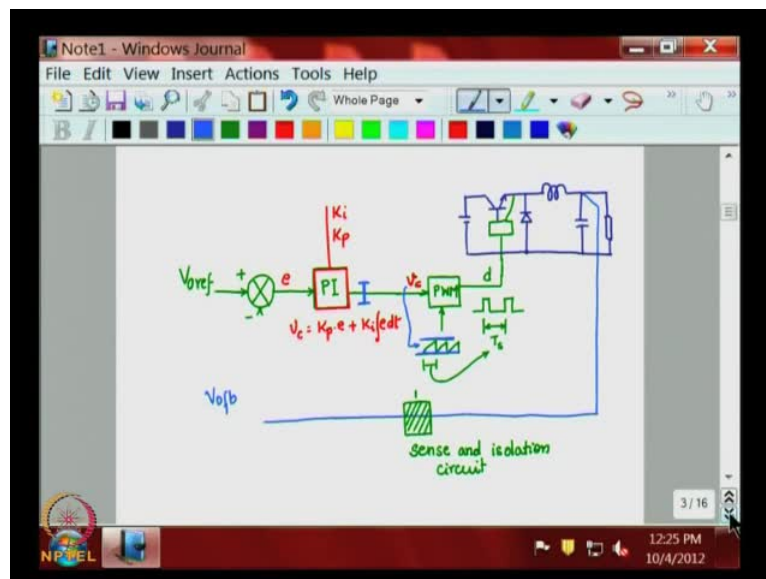
We break there up to that point. Now, first setup this open loop system.

(Refer Slide Time: 22:15)



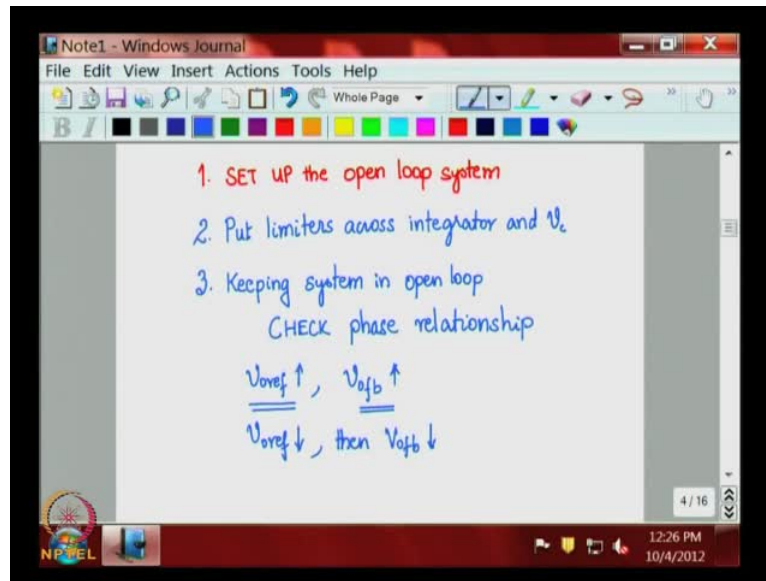
And then next put limiters across integrator and V_c .

(Refer Slide Time: 22:48)



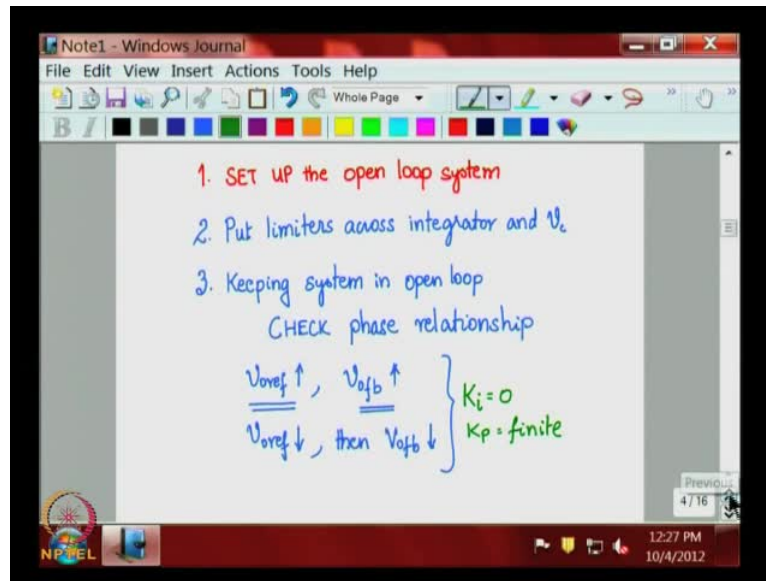
This is very very important very essential, what is basically means is we need to limit the voltage swing at this point, remember that this is a pulse width modulation system where the triangle as the limited voltage, limited lower point and limited upper point and your V_c swing must fall within that limit. So, therefore it is very essential that limit the output of the PI controller within the limits to stay within the limits of the triangular carrier.

(Refer Slide Time: 23:37)



Then the third point in the open loop keeping system in open loop, in open loop check phase relationship, this is very important when $V_{naught\ ref}$ is increased $V_{naught\ feedback}$ should increased, then you are having in proper phase relationship, when $V_{naught\ ref}$ decreases then $V_{naught\ feedback}$ should decrease only then you have proper phase relationship. So, what its basically means is that you have a potential meter here increase $V_{naught\ ref}$ let us say if it is a 5 volt increase from 5 to 6 volts, this would increase e would will increase the output of the PI this will increase V_c which will increase the duty cycle should increase the voltage and then should increase the $V_{naught\ feedback}$.

(Refer Slide Time: 25:45)



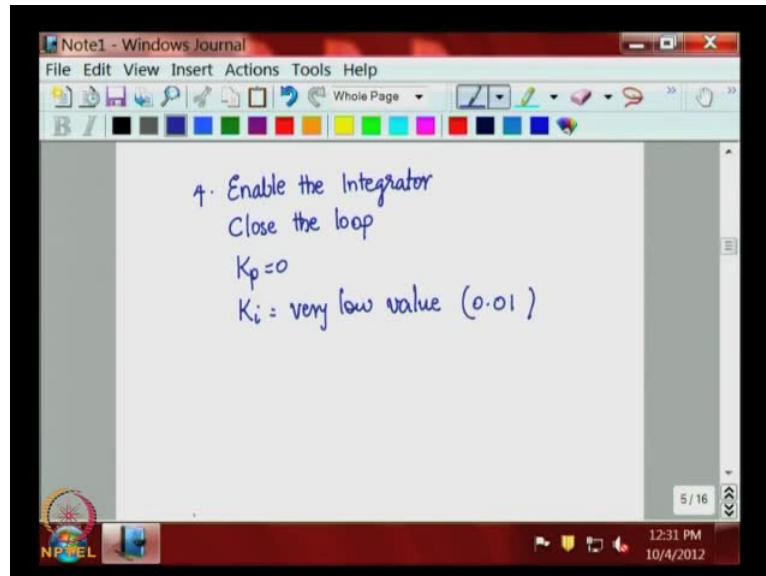
When you are doing the open loop test as like this it would be a good idea to make when you are doing this, make K_i equal to 0 K_p equal to finite value. Because if you have K_i has a non-zero value the implementation of the i portion is the capacitor, the capacitor will invariably charge up and it will charge up to the maximum because it is in open loop. Therefore, bypass K_i or make K_i is equal to 0 by pass the integrator or make K_i equal to 0. So, that when you increase this, this will increase V_c will increase or decrease d will change and this will result in a change in V_{naught} .

So, an increase here increase in $V_{naught\ ref}$ should be felt as in $V_{naught\ feedback}$ decrease in $V_{naught\ reference}$ should be felt as a decrease in $v_{naught\ feedback}$, then your overall loop phase relationship is okay. If it is not okay, if is increase here an resulting an decrease here you will have to change the phase relationship in the any of this component block in the PWM the output, the PWM can give you plus or minus 180 degrees phase change by choosing the output across the collector or the emitter point of the PWM ic that is one possible point. There would be op-amp in-between op-amps in the PI where you could possibly have 180 degree phase shift achieved and also in the op-amp at this compare point.

So, if this is 180 degree out of phase change here results in 180 degree out of phase change here adjust the phase relationship any one of the blocks here, so that you make this two phase change relationship same an increase in $V_{naught\ ref}$ should results in

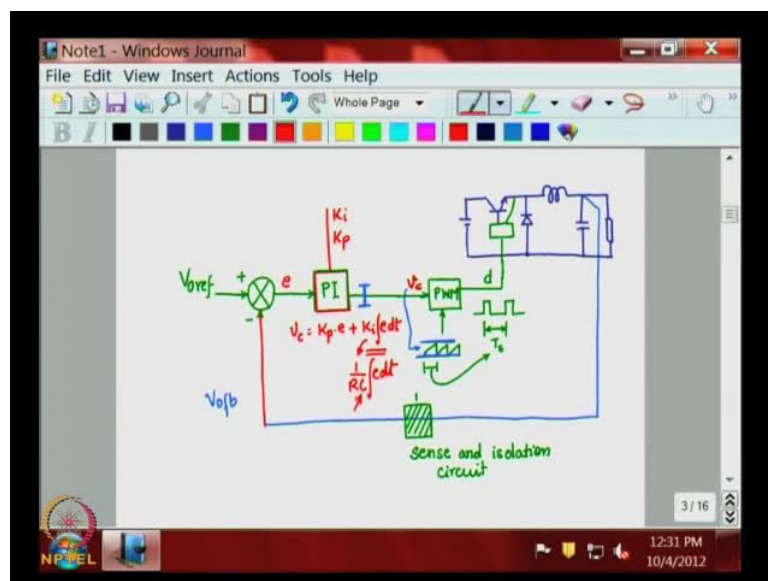
increase in v naught feedback an decrease in V naught ref should results in V naught feedback only then can you go forward. Next...

(Refer Slide Time: 28:23)



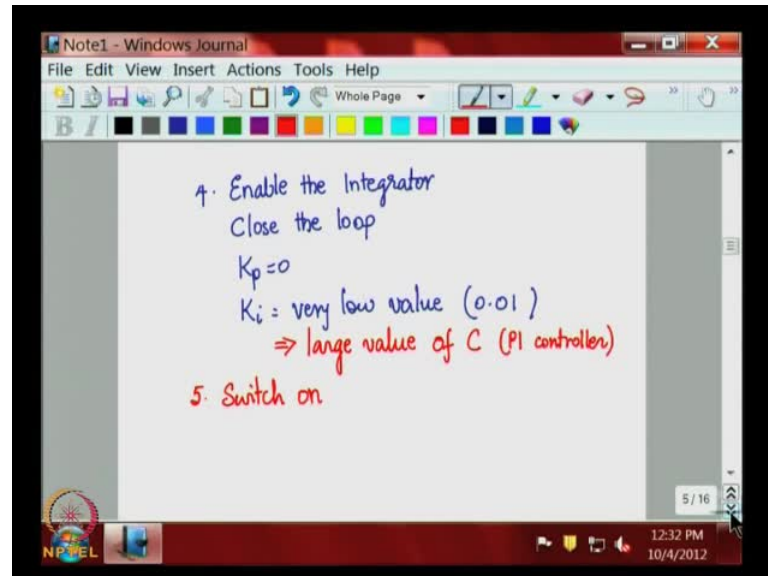
So, after that enable again bring back enable the integrator close the loop, close the loop make K_p equal to 0 K_i equal to very low value, start from very low value like 0.01 are so.

(Refer Slide Time: 29:35)



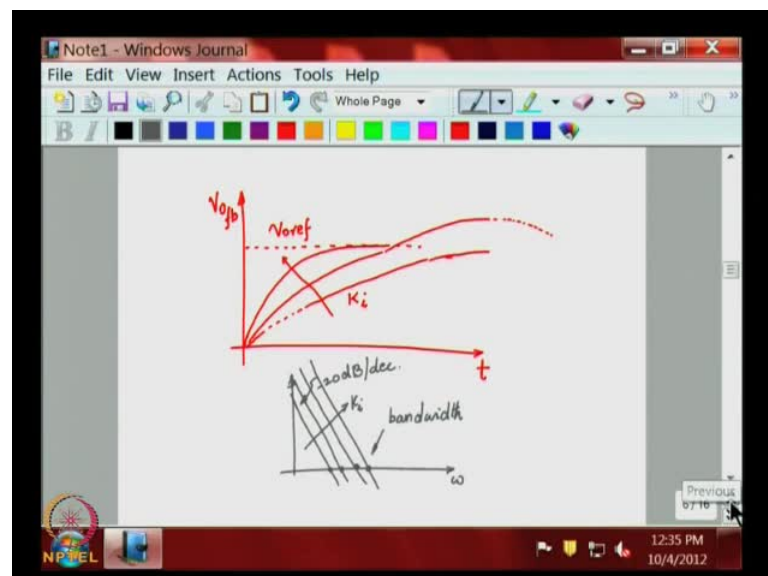
So, what does it mean here you are making this connection close this enable the i position, because you want to use K_i , now make K_p equal to 0 the K_p portion right now cannot focus only i proportional will come later and start with a very low value of K_i or in other word, this is generally implemented as one by RC integral of edt the implementation in op-amp things like that it would mean have very large value of c.

(Refer Slide Time: 30:25)



So, in other words very low value of K_i means this implies start with a very large value of capacitance C of the PI controller PI controller C.

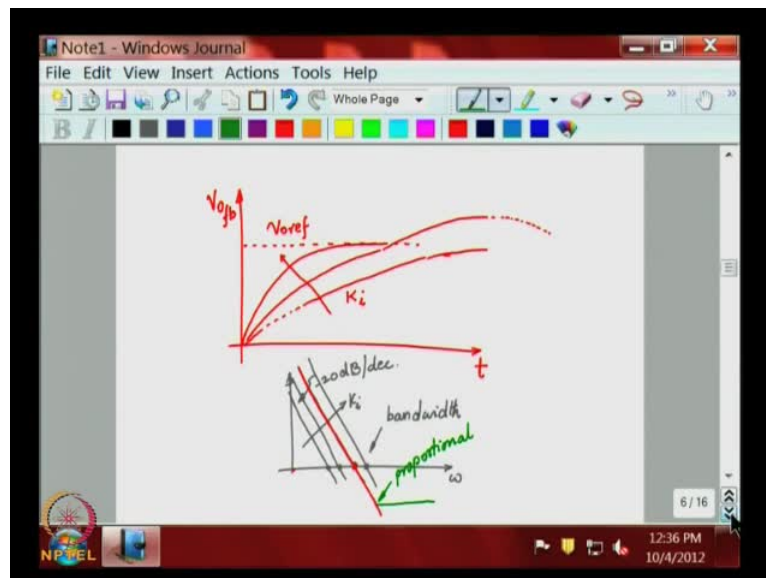
(Refer Slide Time: 31:18)



Now, now when you switch on, switch on the system and observe the switch on transition, the switch on transition dynamics t versus let say V_{naught} or V_{naught} feedback after you switch on v_{naught} feedback is supposed to come to the V_{naught} ref value V_{naught} ref value. So, by adjusting the C value K_i value very low K_i value will imply that it will take very long time for the system to reach study state and sometime it there are overshoots it can take a long time before they start coming down, because you are using only K_i so you will see this, this is K_i is increasing like this.

So, basically if you look at the omega curve we had that i minus 20 db per decade, you are trying to use different parallels by choosing different i is and therefore, different bandwidth unity gain bandwidth. So, this bandwidth will increase as K_i increases as K_i increases so that is what you will be seeing will try to become faster and faster but of course the overshoot may also increase. So, you stop at some performance where it reaches study state at some particular time as set it your performance criteria and then freeze on the value of K_i okay? Stop the value of K_i which basically means that you would have chosen one particular value of K_i parallel the integrator parallel with the particular bandwidth.

(Refer Slide Time: 34:20)



So, you fix that so your low frequency and study state position is fixed and you know that I am in the study state the error is going to be 0, because the dc gain will be infinite.

Now, we need to fix the proportional position which will take care of the high frequency dynamics.

(Refer Slide Time: 35:27)

The screenshot shows a Windows Journal window titled "Note1 - Windows Journal". The notes are as follows:

- 4. Enable the Integrator
Close the loop
 $K_p = 0$
 $K_i = \text{very low value (0.01)}$
 $\Rightarrow \text{large value of C (PI controller)}$
- 5. Switch on
- 6. $K_i = \text{fixed}$ based on response as viewed on the scope of simulation platform

The window also shows a taskbar at the bottom with the time 12:37 PM and date 10/4/2012.

So, the proportional position is fix next. K_i is fixed based on response as viewed on the scope or simulation platform.

(Refer Slide Time: 36:10)

The screenshot shows a Windows Journal window titled "Note1 - Windows Journal". The notes are as follows:

- 7. start increasing K_p from 0

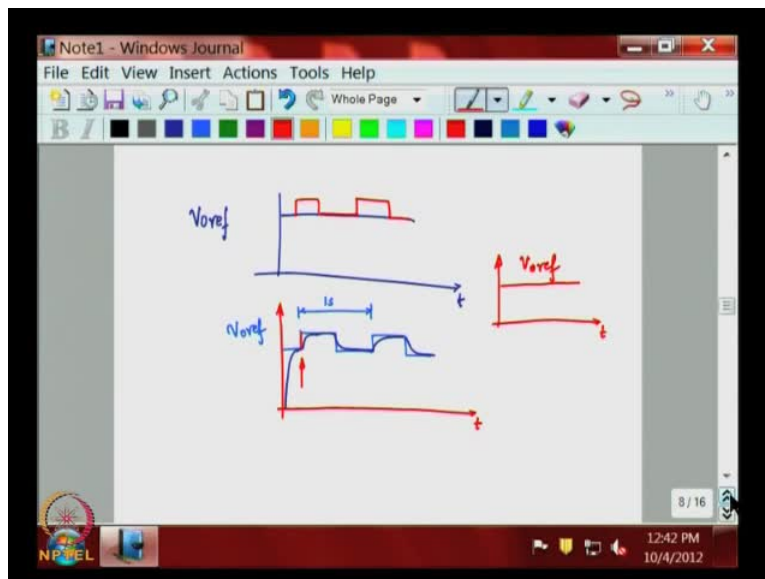
Below the text is a graph showing a transient response. The vertical axis is labeled K_p and the horizontal axis is labeled t . A dashed line represents the target value. A solid line shows the response, which starts at the origin and rises towards the target value. A blue circle highlights the initial part of the response, and a red circle highlights the steady-state part. Below the graph, the text K_i, K_p is written.

The window also shows a taskbar at the bottom with the time 12:39 PM and date 10/4/2012.

The next start increasing K_p value from 0. So, as you start increasing K_p value from 0 and then look the transient response with the particular value that the integrator you would see that let us say, it had the profile like that you should probably now we able to

improve its dynamics, something like that that is your improving the high frequency dynamics study state till governed by the integrator portion and this is due to the P. So, this is what you would see happening when you try to keep increasing P keep on increasing K_p , till you reach a satisfactory by a visual in visual manner satisfactory response and freeze K_i and K_p value and this values of the K_i and K_p the one which using in the controller. Most of the time when you working with the hardware every time you will have switch on and off and try to capture on a storage scope to study that.

(Refer Slide Time: 38:31)

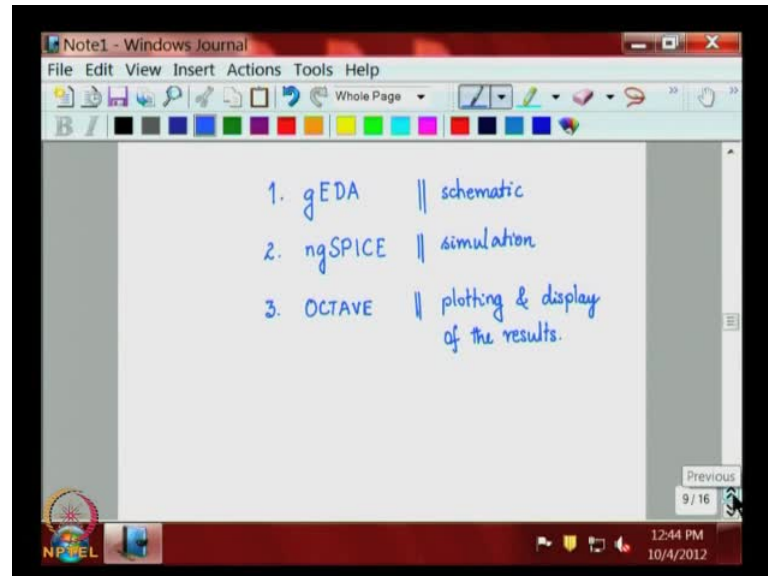


A better way would be to $v_{naught\ ref}$ which is a fixed DC superpose a pulse waveform like this. Then you would get $V_{naught\ ref}$ which is of this nature such this period to have a very low time period, let say one second and there is sufficient time for you to see the response of the system. So, initially you would see that V_{naught} or V_{naught} feedback go to like that, then like this so on and if you trigger your scope for this edge, you trigger the scope for this edge then you will look you will have a stands still waveform on this scope and you code use potentiometer to tune the K_i K_p values to see the improvements in the response transient response.

This way you could tune it and then after the tuning is over the pulse position which you used superposing can be removed and then you can give the proper $v_{naught\ ref}$ something like this DC later on. So, this is the process of tuning the controller directly on

to the system. Let us see view this on simulation platform and see how we go about tuning it this exact circuit can be simulated on many platforms here.

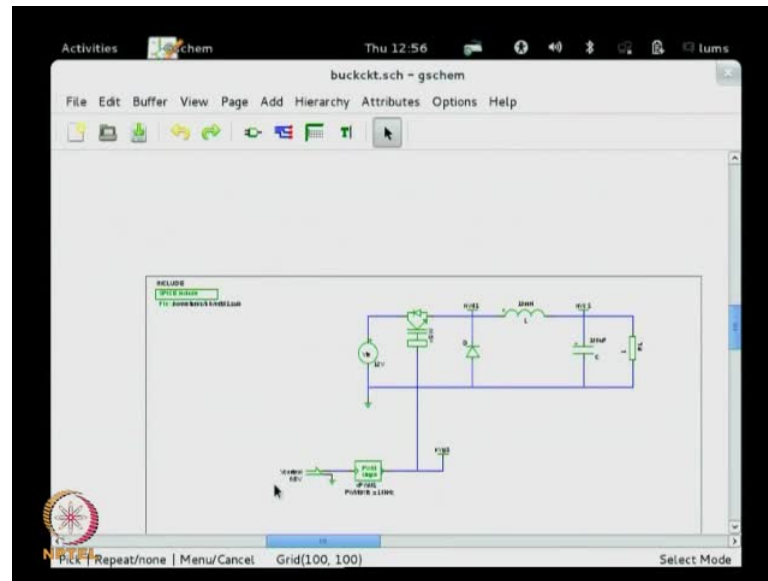
(Refer Slide Time: 41:25)



Let me demonstrate you on few open source software's there is a gEDA the gnu gEDA tool electronics development automation tool there is ngSPICE which is an open source one open source software and of course, octave. So, this three have been combined have been taken to gather and some tools have been develop which can be used for simulating the circuit. So, gEDA is primarily right now being used for schematic in this particular exercise ngSPICE is actually used for performing the simulation run and Octave gnu plot is used for actually plotting in the displaying the results of the results.

So, let we demonstrate to you this first let me enter the schematic in gEDA the schematic will be something like this. So, let us take the bug convertor itself for working this example now let me switch over to computer platform computer I have used gEDA open gEDA this is the schematic detail of gEDA and let us draw the circuit I will use the circuit component and draw the I will probably use capacitance.

(Refer Slide Time: 43:42)



I will need, I will need to use diode, I will need to have a source the dc source will convert that was source, I will need an inductor and I will need power switch and of course, resistor ok. So, let me rearrange all these so this is D C of 5 say 12 volt this is VIN and let us have a switch diode for freewheeling inductor capacitance and the load resistance. So, we position them and let me zoom like the use the wire joined the components like this just like any schematic editor.

So, we have over basic circuit done we can then enter some typical values we will do that later and then we need to we need to put in the controller. So, let us take from the block first PWM we need that we need PID with wind up we need that we need the compare block we need that. So, let us quickly put them together like this and we use the wire connect them so we have connected so we will capable or editing the PID block parameters here later will do that.

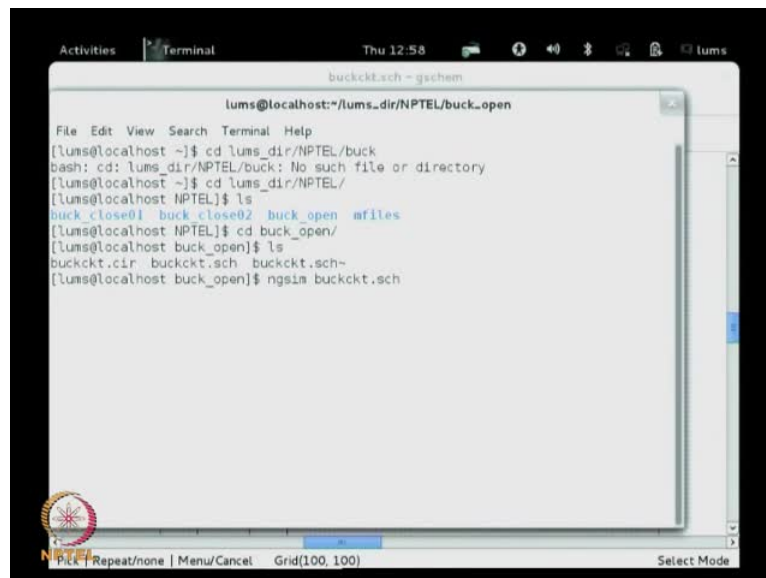
So, we have this setup now we need to add one more important thing and include file now this include file I will place it here this basically points file basically points to the you see that we have added this component this are generic switches, generic diodes and here you have the PWM block which has logic inside the PID blocks with the logic inside. All this logic in the equations are in a file called sub-circuit file sub file and you have to point to that so that model can be taken from that file, so just include that file there. So, after you have included all that file so we will have this completed circuit do

not forget to put the ground with the circuit should have the ground in spice must be knowing that all the calculation done with respect to a zero node with the ground node.

Now, I given a VIN 12 volt and this is the switch which named as Xsw now remember here I use the label generic label and named it named this junction as nVd point and there is another point node here this is nVc voltage across the capacitance L is set at 10 millihenry rl load is set as one, one Ohms see this is a 12 volt this is a buck convertor. Let us, say it is a 5 volt it will be 1 by 5, 5 amps 5 into 5, but 25 volts of power will we sent to the load.

So, if we look closer right now I keeping it into open loop this is a reference value of 0.5 control voltage which I am setting for the PWM stage which is set at 10 kilo hertz output of the PWM shown as nVg just this one I just remove PI now and look at the include file where right now I am using in edt 01 dot sub which contains all the equations for the underline components component blocks. Now, this can be simulated open loop can simulated we can see the various waveform so that is why this can be enabled ok. So, let have look at these various simulated it and see how the various wave form comes.

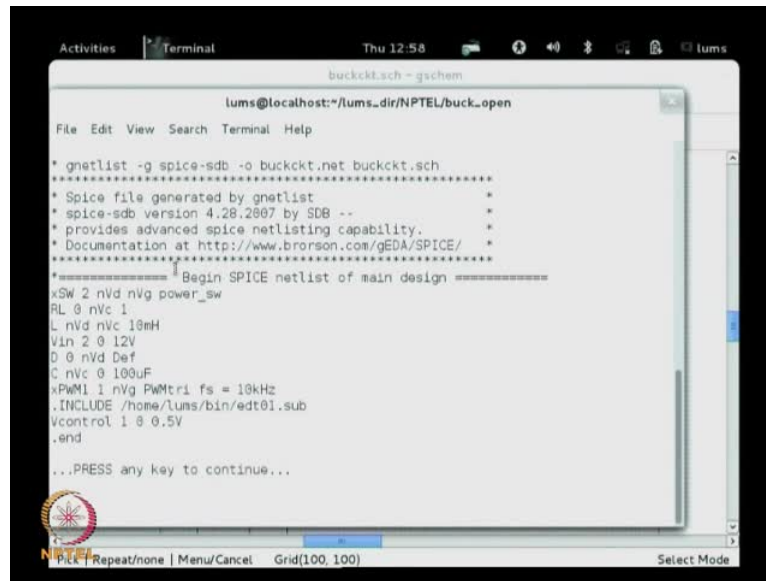
(Refer Slide Time: 51:21)



```
Activities Terminal Thu 12:58
buckckt.sch - gschem
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
[lums@localhost ~]$ cd lums_dir/NPTEL/buck
bash: cd: lums_dir/NPTEL/buck: No such file or directory
[lums@localhost ~]$ cd lums_dir/NPTEL/
[lums@localhost NPTEL]$ ls
buck_close01 buck_close02 buck_open mfiles
[lums@localhost NPTEL]$ cd buck_open/
[lums@localhost buck_open]$ ls
buckckt.cir buckckt.sch buckckt.sch~
[lums@localhost buck_open]$ ngsim buckckt.sch
```

Let me go to the command line and run as write the ngsim the schematic file is executed.

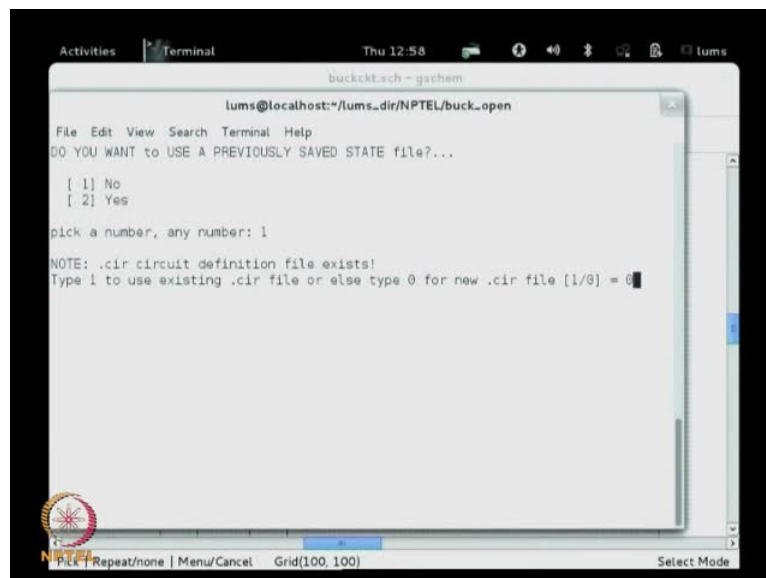
(Refer Slide Time: 51:35)



```
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
* gnetlist -g spice-sdb -o buckckt.net buckckt.sch
*****
* Spice file generated by gnetlist
* spice-sdb version 4.28.2807 by SDB --
* provides advanced spice netlisting capability.
* Documentation at http://www.brorson.com/gEDA/SPICE/
*****
***** Begin SPICE netlist of main design *****
xSW 2 nVd nVg power_sw
RL 0 nVc 1
L nVd nVc 10mH
VIn 2 0 12V
D 0 nVd Def
C nVc 0 100uF
xPWM1 1 nVg PWMtri fs = 10kHz
.INCLUDE /home/lums/bin/edt01.sub
Vcontrol 1 0 0.5V
.end
...PRESS any key to continue...
```

So, on executing first the net list is generated so this is the net list of the schematic that just saw r l l v i n d c and the PWM let us finding to the including file all those things.

(Refer Slide Time: 51:56)



```
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
DO YOU WANT TO USE A PREVIOUSLY SAVED STATE file?...

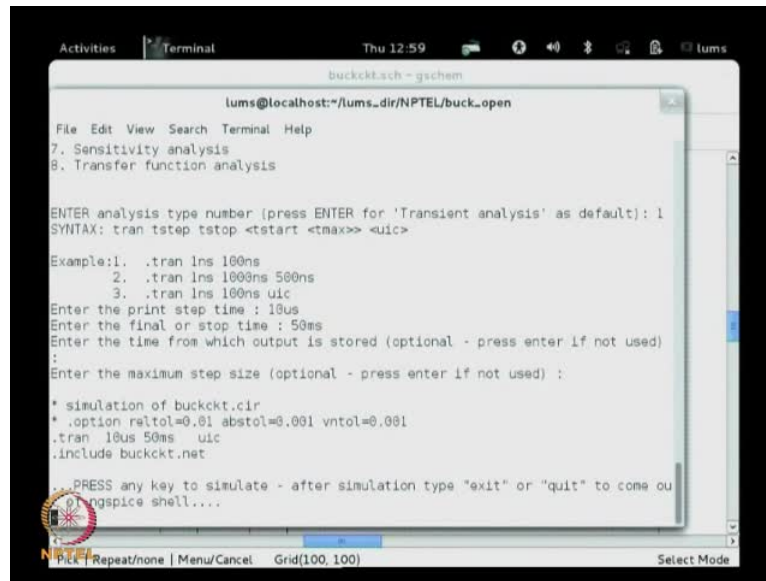
[ 1 ] No
[ 2 ] Yes

pick a number, any number: 1

NOTE: .cir circuit definition file exists!
Type 1 to use existing .cir file or else type 0 for new .cir file [1/0] = 0
```

And then we need to such a parameters.

(Refer Slide Time: 52:04)



```
Activities Terminal Thu 12:59 lums
buckckt.sch - gschem
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
7. Sensitivity analysis
8. Transfer function analysis

ENTER analysis type number (press ENTER for 'Transient analysis' as default): 1
SYNTAX: tran tstep tstop <tstart <tmax>> <uic>

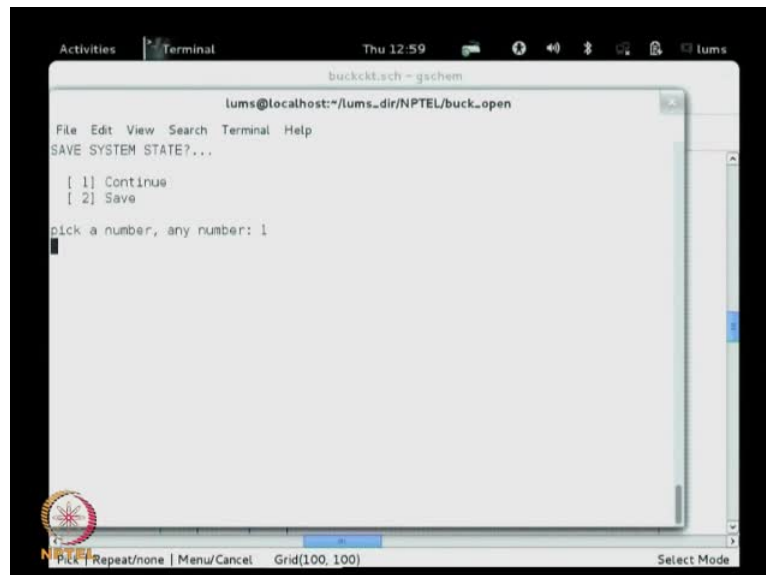
Example:1. .tran lns 100ns
        2. .tran lns 1000ns 500ns
        3. .tran lns 100ns uic
Enter the print step time : 10us
Enter the final or stop time : 50ms
Enter the time from which output is stored (optional - press enter if not used) :
Enter the maximum step size (optional - press enter if not used) :

* simulation of buckckt.cir
* .option ra1tol=0.01 abstol=0.001 vntol=0.001
.tran 10us 50ms uic
.include buckckt.net

PRESS any key to simulate - after simulation type "exit" or "quit" to come ou
ngspice shell....
```

So, let us the choose analysis type. So, let us choose transition analysis and we will give print step at the time 10 micro seconds and the final stop time of the 50 milliseconds that is same. And we request ng SPICE to perform the simulation see the simulation is running and then we are now we will able to see the waveforms.

(Refer Slide Time: 52:31)

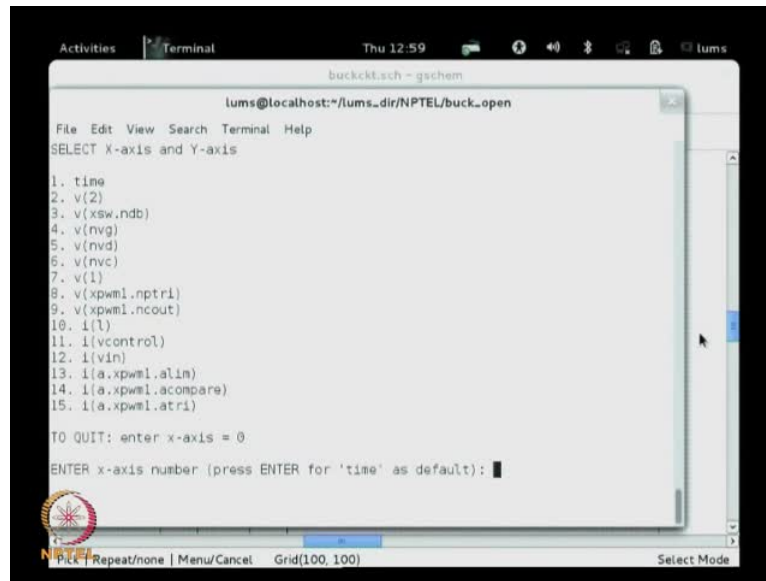


```
Activities Terminal Thu 12:59 lums
buckckt.sch - gschem
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
SAVE SYSTEM STATE?...

[ 1] Continue
[ 2] Save

pick a number, any number: 1
```

(Refer Slide Time: 52:34)

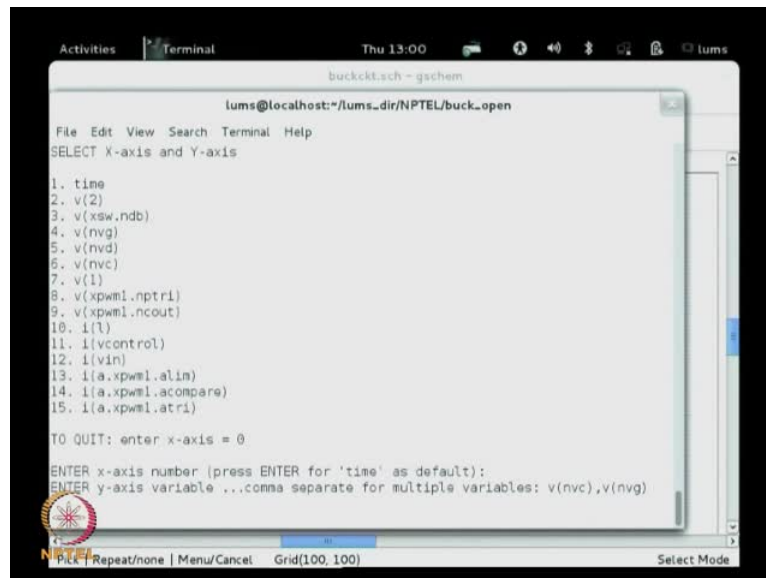


```
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
SELECT X-axis and Y-axis
1. time
2. v(2)
3. v(xsw.ndb)
4. v(nvg)
5. v(nvd)
6. v(nvc)
7. v(l)
8. v(xpwm1.nptr1)
9. v(xpwm1.ncout)
10. i(l)
11. i(vcontrol)
12. i(vin)
13. i(a.xpwm1.alin)
14. i(a.xpwm1.acompara)
15. i(a.xpwm1.atr1)

TO QUIT: enter x-axis = 0
ENTER x-axis number (press ENTER for 'time' as default):
```

Now, here the waveforms you have time and we have host of more voltages and branch currents, x axis of course, we want it to be time and the y axis we would like to see the waveform across the output may be and may be the waveform which is given as pulse width to the switch and probably even the current here.

(Refer Slide Time: 53:31)

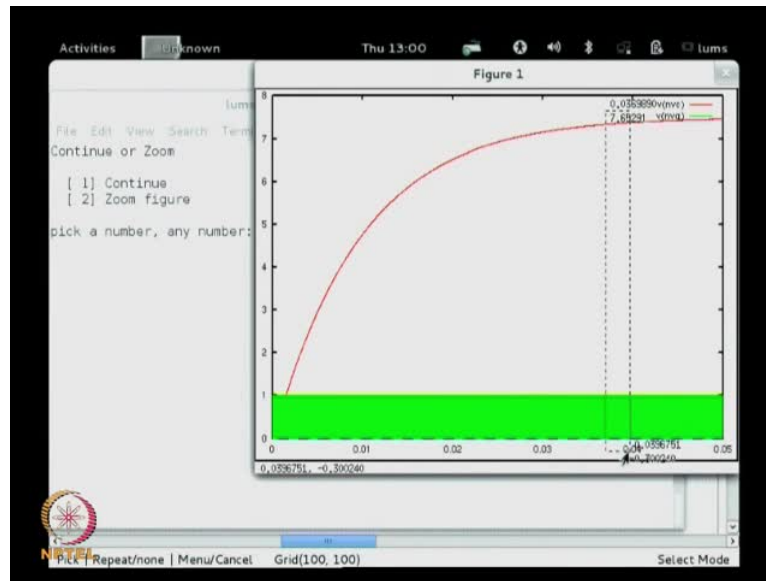


```
lums@localhost:~/lums_dir/NPTEL/buck_open
File Edit View Search Terminal Help
SELECT X-axis and Y-axis
1. time
2. v(2)
3. v(xsw.ndb)
4. v(nvg)
5. v(nvd)
6. v(nvc)
7. v(l)
8. v(xpwm1.nptr1)
9. v(xpwm1.ncout)
10. i(l)
11. i(vcontrol)
12. i(vin)
13. i(a.xpwm1.alin)
14. i(a.xpwm1.acompara)
15. i(a.xpwm1.atr1)

TO QUIT: enter x-axis = 0
ENTER x-axis number (press ENTER for 'time' as default):
ENTER y-axis variable ...comma separate for multiple variables: v(nvc),v(nvg)
```

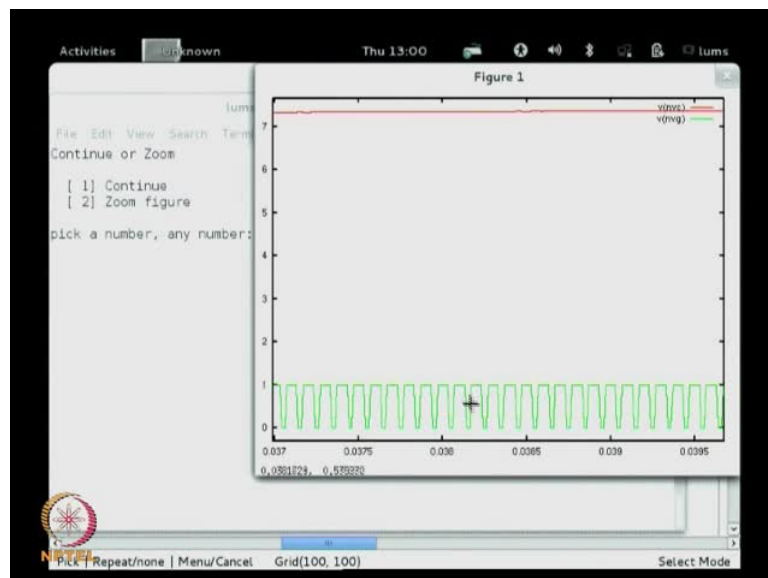
So, let us see these waveforms so one is nVc v nVc and also probably nVg the get waveform.

(Refer Slide Time: 53:47)



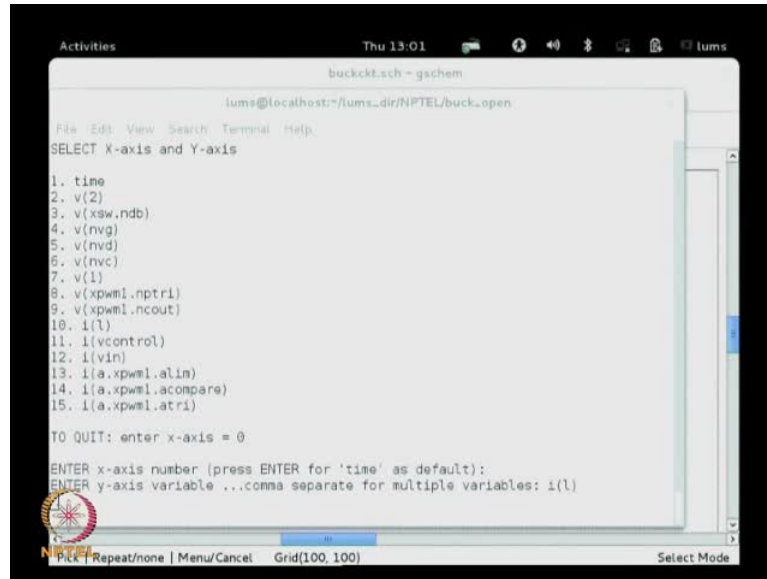
So, this the waveform that you seeing and you see that naturally going a settling at 7 or 7.5, because we had set some arbitrary value of.

(Refer Slide Time: 54:13)



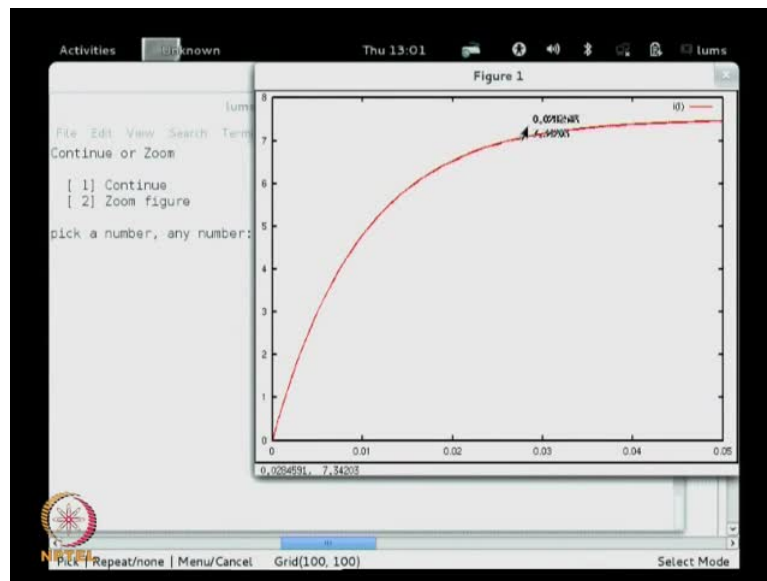
So, let us say you can zoom in this is pulse width something like 70 percent also waveform which seen in gate of the power switch and if you would like to see the waveform of the inductor current i_L . We will like to see the waveform of the inductor current i_L though it shows as smooth.

(Refer Slide Time: 54:36)

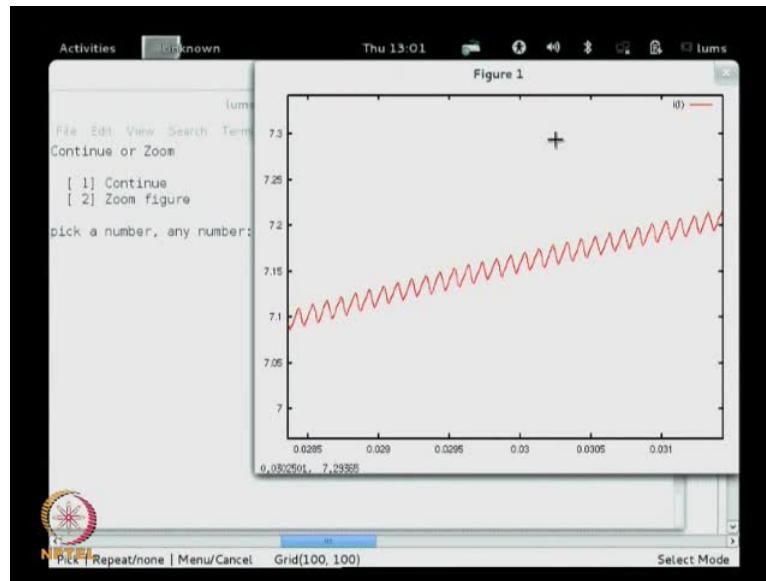


```
Activities Thu 13:01 buckckt.sch - gschem
lums@localhost:~/lums_dir/NPTEL/buck.open
File Edit View Search Terminal Help
SELECT X-axis and Y-axis
1. time
2. v(2)
3. v(xsw.ndb)
4. v(nvg)
5. v(nvd)
6. v(nvc)
7. v(l)
8. v(xpwm1.nptr1)
9. v(xpwm1.ncout)
10. i(l)
11. i(vcontrol)
12. i(vin)
13. i(a.xpwm1.alin)
14. i(a.xpwm1.acompare)
15. i(a.xpwm1.atr1)
TO QUIT: enter x-axis = 0
ENTER x-axis number (press ENTER for 'time' as default):
ENTER y-axis variable ...comma separate for multiple variables: i(l)
```

(Refer Slide Time: 54:46)



(Refer Slide Time: 54:50)



When you zoom in an open out you will see the triangular wave shift to the inductor current of the bug just like as we see in regular bug converter up and down in the triangular fashion. This is the open loop system and we would like to tune the controller.

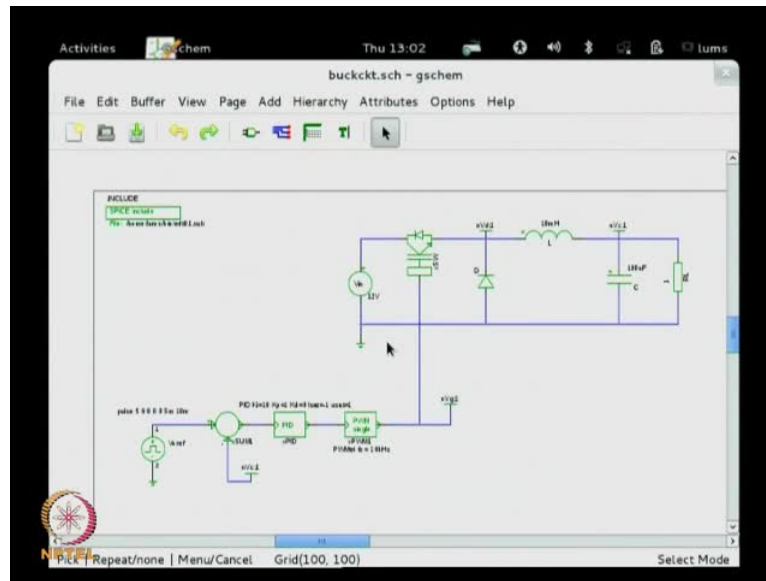
(Refer Slide Time: 55:15)

```
lums@localhost:~/lums_dir/NPTEL/buck_close01
File Edit View Search Terminal Help
[lums@localhost buck_open]$ cd ..
[lums@localhost NPTEL]$ cd buck_close01
[lums@localhost buck_close01]$
```

The terminal window shows the user navigating through directories: [lums@localhost buck_open]\$ cd .., [lums@localhost NPTEL]\$ cd buck_close01, and [lums@localhost buck_close01]\$. The terminal window title is "lums" and the system time is "Thu 13:02".

So, let me show the close loop system which I have already built, so the circuit is like this so you see the closed loop system here.

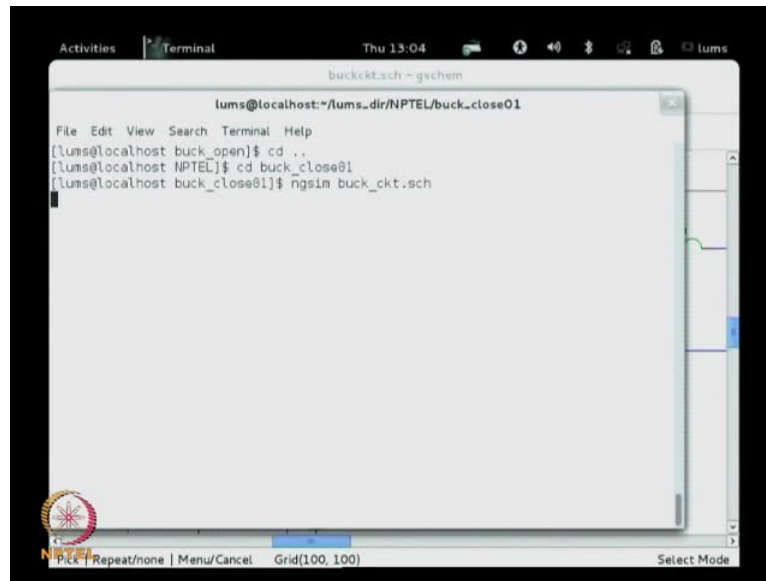
(Refer Slide Time: 55:48)



This is the buck converter I am giving a reference pulse reference like I just discussed this is the reference v_{naught} . Let me zoom into this point plus minus and feeding back the voltage over the capacitor at this point the labelled point output of the controller is goes to the PID as set d 0 not using right, Now it is limited to plus minus 1 lower saturation and upper saturation of the PID PWM signal is triangular carrier having minus one to plus one which has the frequency of the 10 kilo hertz K_i 10 and K_p one value set and the output of that goes to the power switch.

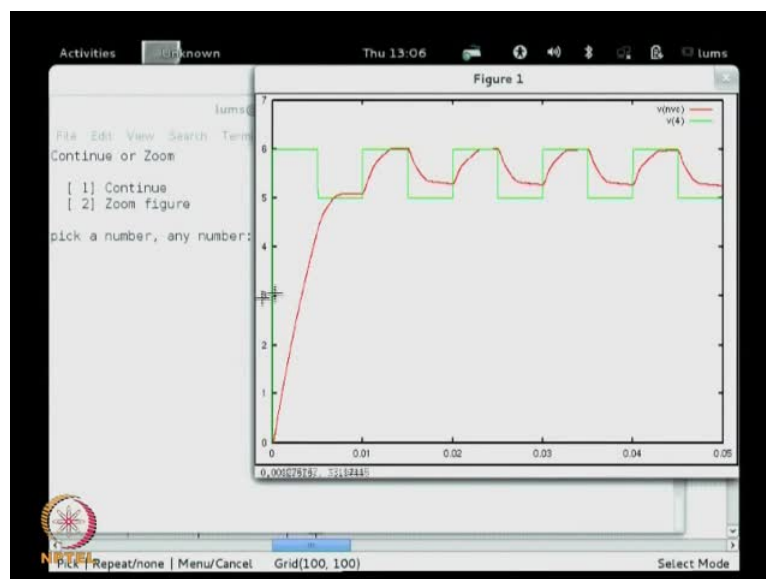
Note this point here, this reference is set a pulse which having 5 volts from 5 volt is going to 6 volt and then back again to 5 volts it is a pulse one volt superimposed on 5 volt and it has 5 milliseconds to 10 milliseconds the 5 milliseconds is on time 10 millisecond period. So, that we will able to see the dynamics of this so this let us simulate and see how the wave shape looks like.

(Refer Slide Time: 57:50)



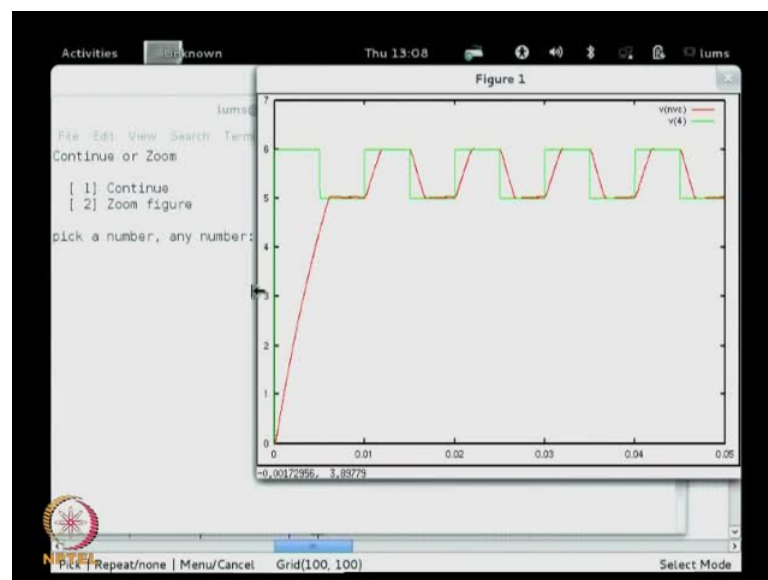
So, we go again back here ngspice buck circuit dot schematic, so you see the net list get generated all these the summer PID the switch RL L VIN voltage diode capacitance the PWM block the, include file all those thing capture from the schematic and the net list generated and I could use the volt values of transient analysis value ask ng SPICE to simulate and ng SPICE will simulate and put the data into raw file and we are ready to view the outputs. So, time we would like to time and the voltage across the output capacitance volt we would also like to view the voltage of the control voltage the pulse v naught ref.

(Refer Slide Time: 59:21)



So, you see that for this values of K_i and K_p the this is v naught ref five to 6 volts every 5 milliseconds up and down and this red waveform is output voltage which is tracking the reference waveform tracks the reference waveform and then the dynamics occurring and the tracks re-steady state accuracy, then starts coming down. So, this is what it is trying to do so you can try to tune your K_p and K_i to improve the responses. So, in this way you iteratively, iteratively manipulate let say we go here increase a K_p value K_i value set it little then increase the K_p value let us go make it 10 and save the file go back again here regenerate the net list and ask ng SPICE to simulate it, it will give you the waveform.

(Refer Slide Time: 61:30)



And let us view the same waveform you see because we increase K_p the dynamic response has improved and it is able to go within, within the pulse period up to its stable point comes back to this stable point so on. So, you can play around with K_i and K_p value and get better and better dynamic responses once you get satisfactory response you stop, because there are infinite solutions and you, you need not try more than to get a better way that is when there are infinite solutions you do not try to search for the optimum in this trial and error method. Trial and error method just give one solution so that it just work getting an optimal solution is the ball game where you have mathematically try to extract the optimal out of a cast function.

So, with this we stop here at this point and continue in the next class with the other method which is the root locus method we use the model small signal model of the system and then try to obtain the root locus and then the controller parameter. This is one method what we discussed today by the trial and error trial and error approach of the cyclonic nickels method we you try to first tune K_i then K_p and then view the responses.