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

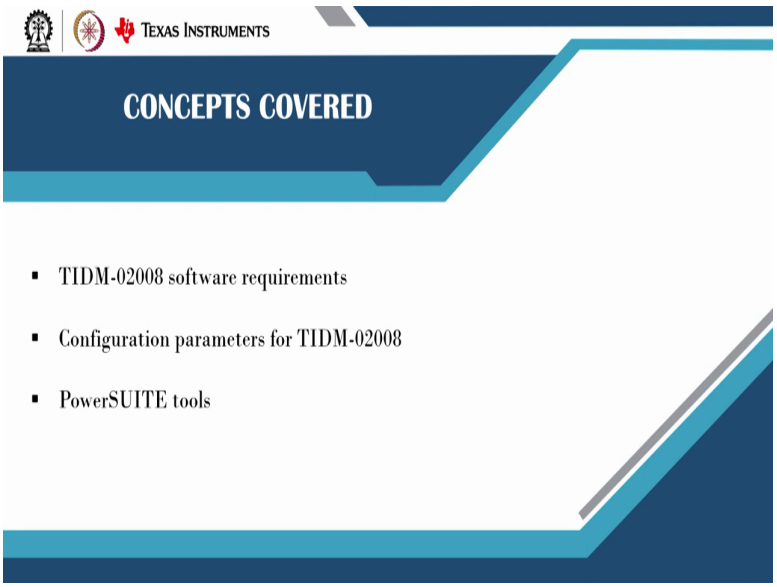
Dr. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Module - 09
Digital Control Implementation using Microcontrollers
Lecture - 90
Texas Instruments TIDM-02008 Reference Design Software Overview

Hello and welcome to the NPTEL online certification courses a course on Digital Control in SMPCs and FPGA Based Prototyping. I am Aditya Dholakia from Texas instruments and today we are going to talk about module 9 Digital Control Implementation using Microcontrollers. And specifically lecture 90 Texas Instruments TIDM-02008 Reference Design that is a Bidirectional Totem Pole PFC.

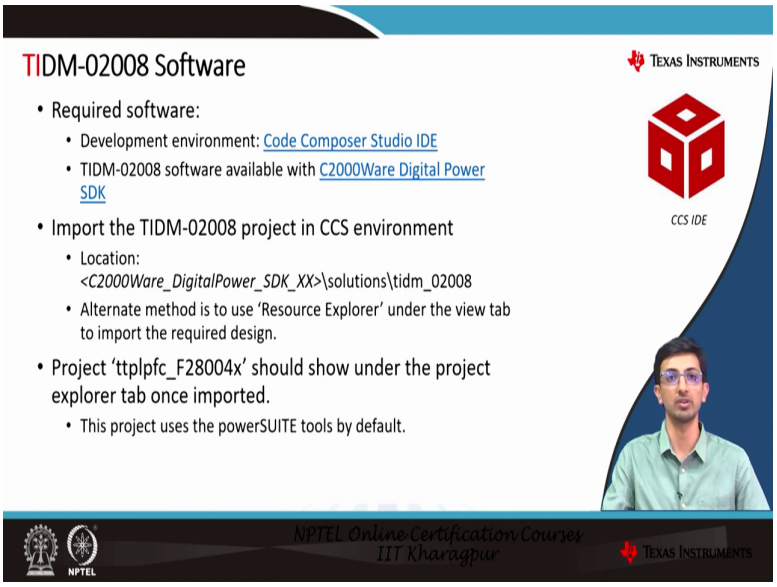
In the last lecture, we went through the bidirectional totem pole PFC in a brief overview, and in lecture 10 Professor discussed the totem pole PFC in detail. In this lecture, we are only going to go through the software overview and the software pieces needed to go through the design.

(Refer to Slide Time: 01:05)



The concept covered in this lecture will be the TIDM software requirements that is what equipment you need, and what infrastructure you need to set up the software. The second part is the configuration parameters that are needed for TIDM-02008. The third part is the power suite tool which will enhance your development experience.

(Refer Slide Time: 01:24)

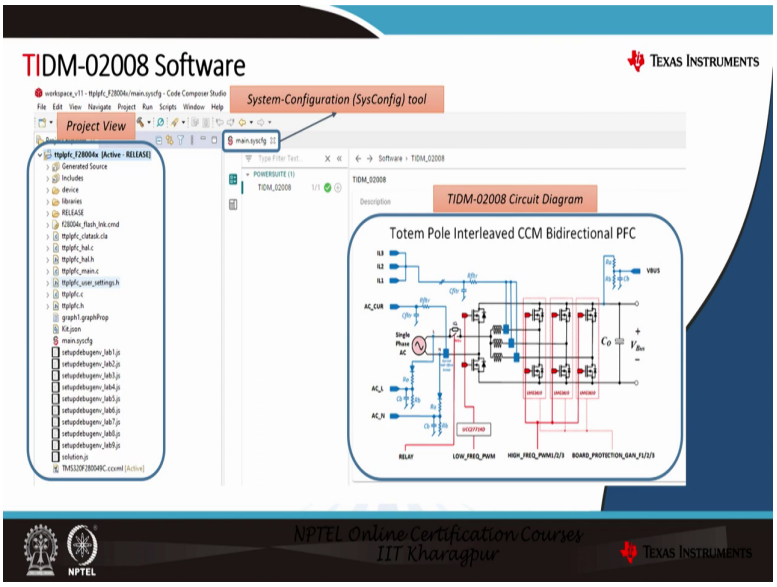


So, starting with the required software the first and foremost thing that is needed is the code composer studio integrated development environment provided by TI which is the SIDE. The second part is the C2000 Digital Power SDK which contains the software available for the TIDM- 02008.

Once you have installed all the CCS and the C2000 digital power SDK. The next step is to open the CCS environment and import the TIDM-02008 project. So, for that, you need to go to CCS to import the TIDM project from the location mentioned here. The alternate method is to use the resource explorer tab available under the view tab to import the required design.

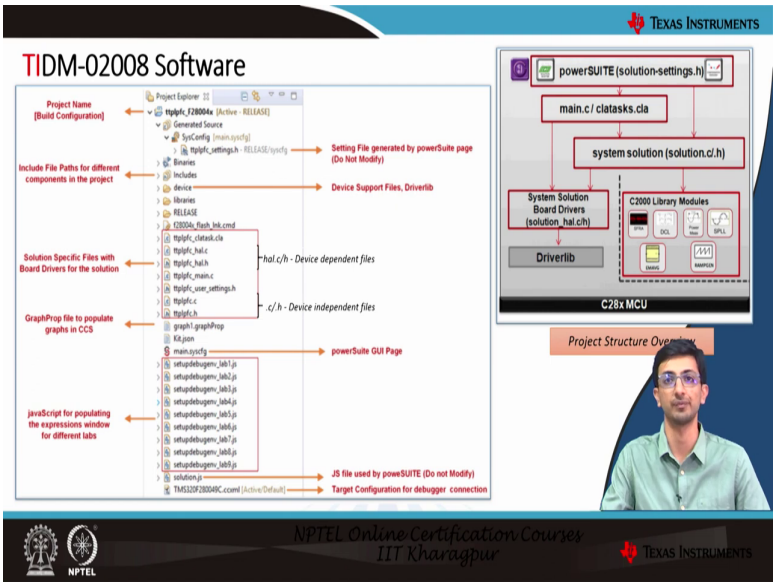
The C2008 ware digital power SDK also contains multiple more designs which you can explore from. Once you have imported the design the project type which is the totem pole PFC underscores the device part number F28004x and should show under the projects explorer tab once imported. This project by default uses the power suite tools.

(Refer Slide Time: 02:29)



Once you have imported the design into CCS the CCS will look something like this. On the left-hand side, you can see the project view by default it will open the main dot sys config which is the system configuration tool from which you will be able to see the TIDM-02008 circuit diagram. Once you achieve this we will deep dive into what are the factors that contributed to the project view.

(Refer Slide Time: 02:56)



So, going a little bit in-depth into the project view section the first thing that you see here is the type underscore F28004x project part number. The second thing that you see here is the ttplpfc underscore settings h dot file.

This settings file is generated by the power suite page and the user is not expected to modify it. In the previous slide, we saw the brief snapshot of a sys config tool that sys config tool is capable enough to generate the header files which is not configurable by the user directly. Whatever configurations are to be done in the totem pole PFC settings file are to be done through the sys config page.

Then the folder comes is the includes folder which has all the necessary file paths for different components in the project example the compiler path. The next thing is the device folder which contains the device support file or the driver lib file based on your style of coding. Then we come down to the solution-specific file with board drivers for the solution.

So, if you are using cla for any of the tasks you will have a file called cla task underscore cla along with the main dot c which contains the C28x specific code. Then there are specific files such as the totem pole PFC underscores hardware abstraction layer that is the hal dot c as well as hal dot h which are device-dependent files.

So, in this case, we are having all the device-specific information of F2804004 x in half underscore hal dot c as well as hal dot h files. The main file contains all the necessary initialization parts then there is something called the solution dot c that is totem pole PFC dot c, as well as totem pole PFC dot h, files these are device-independent files.

So, whatever device you want to choose you need not update these two files for whatever device you need to use you can only you only would need to update the hall dot c as well as hal dot h files. Coming down there is a file called graph properties that is graph 1 dot graph prop.

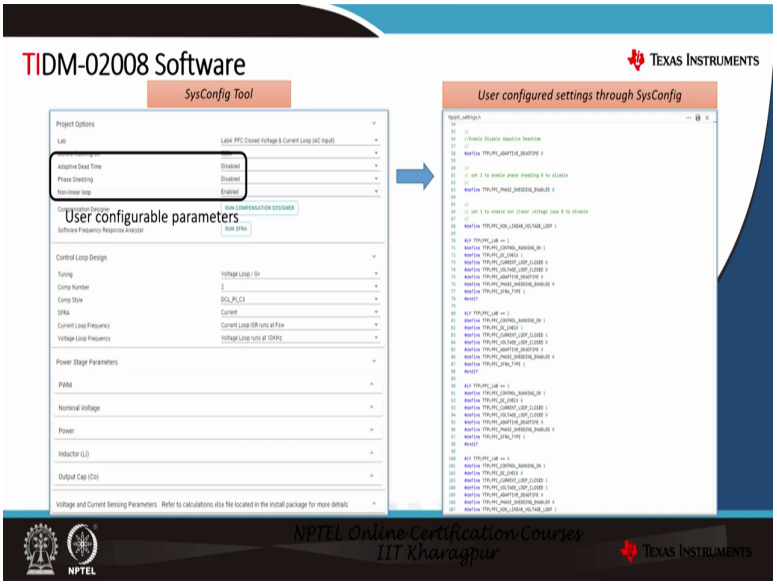
This file is used to populate graphs in the CCS window. Then there is again the main dot sys config file which is a power suite GUI page where you will be able to update all the other options we will go to that in a little bit more detail in the upcoming slides. There are many many available lab experiments for this particular design that you can operate an open loop closed loop, PFC then you can operate an inverter mode as well.

So, all these contribute to different labs you can select any one of the labs which you want to use. Lastly, you see the cc XML file which is the target configuration file for the debugger connection. This is the file that you will need to connect your computer with the actual launch pad or the control card that you are using.

On the right, you see that is a generic view of the structure. So, you start with the power suite page which is the solution underscore settings h dot file that we saw on the left. From that you have the main dot c or the cla task dot cla based on where you are running your code from that is either C28 x or your cla.

Then there are some solution-specific files that we saw that are t ttplpfc solution dot c as well as solution dot h files. And then coming down we have the hardware abstraction layer files which are device-dependent files and the bottom-most layer is a driver lib which is there basically inside the device folder. There are additionally some C 2000 library modules also available with the design that is like SFRA compensator designer etcetera.

(Refer Slide Time: 06:28)

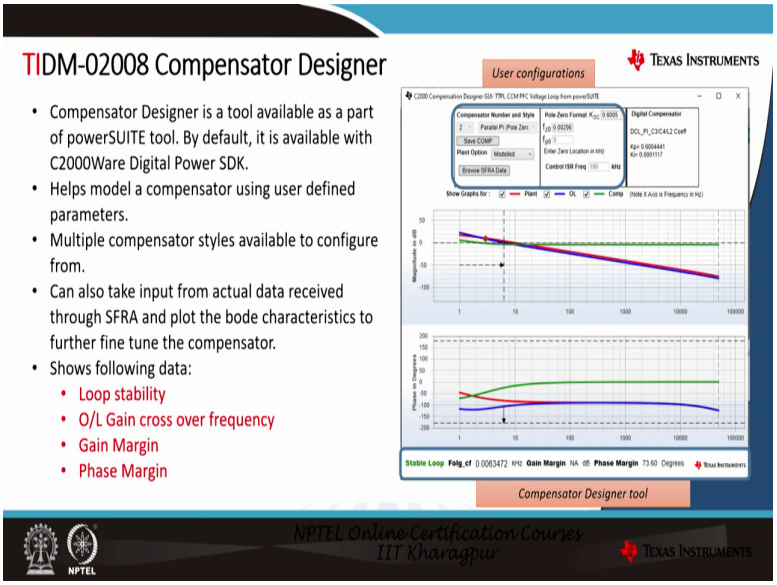


So, going a little bit in detail about the sys config tool what the sys config tool enables you to do is that the left side view shows when you open the sys config tool. Now I have highlighted certain aspects of it that as the adaptive dead time, the phase shedding, and the non-linear loop mode where the options are to enable or disable.

So, let us suppose you start with adaptive dead time, and currently, the option is disabled. Correspondingly on the right-hand side, you can see a macro defined as ttplpfc underscore adaptive underscore dead time. So, as and when you change the adaptive dead time in the GUI from disabled to enabled correspondingly the macro will automatically change from 0 to 1.

You the user need not go and update within the header file what they need to do. same way for other parameters as well. There are many more configurations available within the sys config tool that they can update. The ttplpfc underscore settings dot h file cannot be edited by the user this is wholly and solely generated by the sys config tool.

(Refer Slide Time: 07:31)

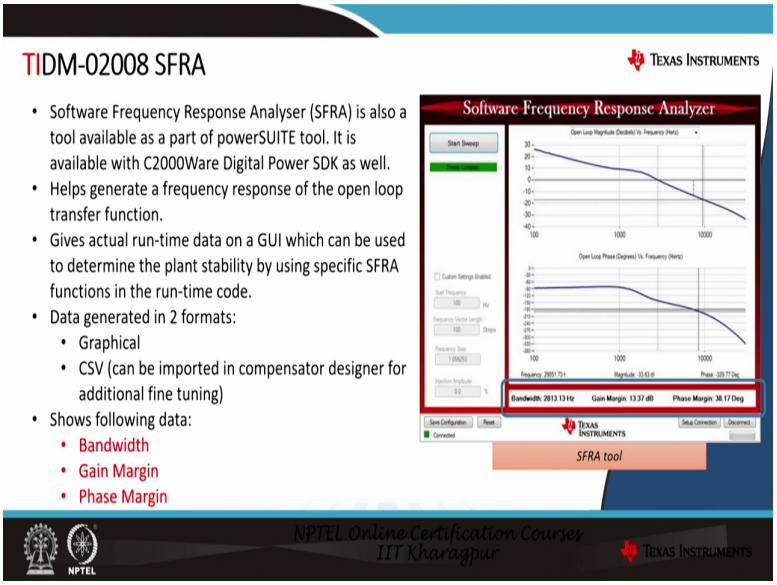


Coming to the compensator designer part the compensator designer is a tool available as a part of the power suite. By default, it is available with the C2000 ware digital power SDK you can also use it as a standalone tool. This helps model a compensator using user-defined parameters.

So, multiple compensator styles are available that you can choose from. So, on the right that you see here in the image, the highlighted part indicates several compensators as well as the style that you can choose from. There are multiple available compensator styles that as PI controller PID controller etcetera also you can configure the pole location zero location within that GUI itself.

You can also take input from actual data received in the real-time system through SFRA and plot the bode characteristics to further fine-tune the compensator. I will briefly touch upon this part in the upcoming slide as well. What the compensated designer provides is the loop stability, open loop gains crossover frequency gain margin as well as the phase margin based on which even before loading the actual compensator on the system we will be able to figure out what the system stability is going to be.

(Refer Slide Time: 08:41)



The next part is the software frequency response analyzer. So, the software frequency response analyzer is also a tool available as a part of the power suite. It is again available with the C2000 ware digital power SDK and in the same way as compensator designer, you can use this tool as a standalone tool.

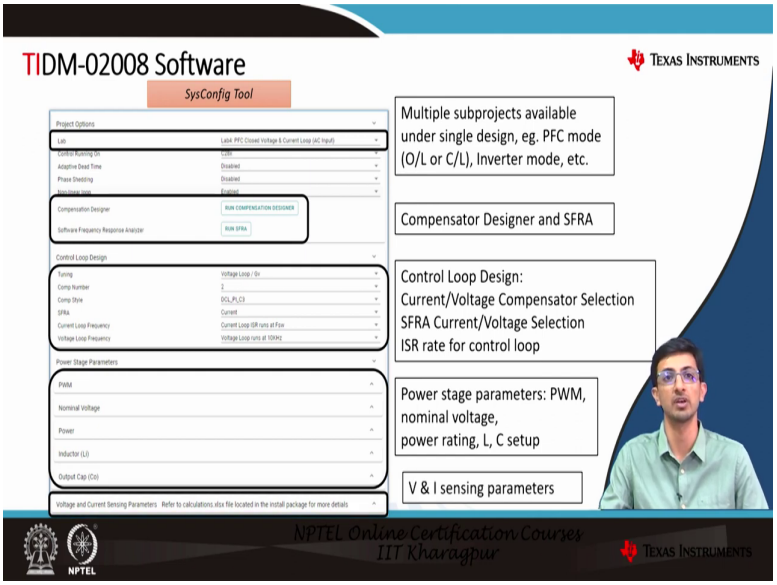
The special part about this tool is that it gives actual runtime data on the GUI which can be used to determine the plant stability. So, what the user needs to do is that in their runtime ISR or their runtime code they just need to inject a function specific to SFRA which pulls the data specific to current or whatever current or voltage they desire in the runtime code.

So, the SFRA will read that data using the function and will continuously plot the curve as and when what is required. The data is generated in two formats one is graphical as you see here in the picture and the other is CSV format which can be imported into the compensator designer for additional fine-tuning.

Now, the first question that may arise is if we already designed our compensator in the compensator designer then why would you need something like an SFR? But whatever you design is theoretical and it may not be exactly replicated in the actual system because of limitations like compensator capacitor value or the inductor value etcetera.

There may be certain other design parameters that may affect the actual compensator from the design compensator. So, this closed-loop performance can help you help fine-tune the performance of your closed-loop control. The following data is showcased using the SFRA tool that is bandwidth, gain margin as well as phase margin. Based on this you can take a call runtime that the system is stable or whether it is having marginal stability.

(Refer Slide Time: 10:23)



So, in the sys config tool for TIDM-02008, you will be able to see the following options. The first option as I discussed earlier is the lab option where you have multiple lab options available starting with lab 9, lab 1 to lab 9. An example can be the PFC mode you can run either in an open loop or closed loop same way for inverter mode etcetera. You are free to explore all the options available as and when you select your lab options automatically the other options will be pre populatedpre-populatedault values.

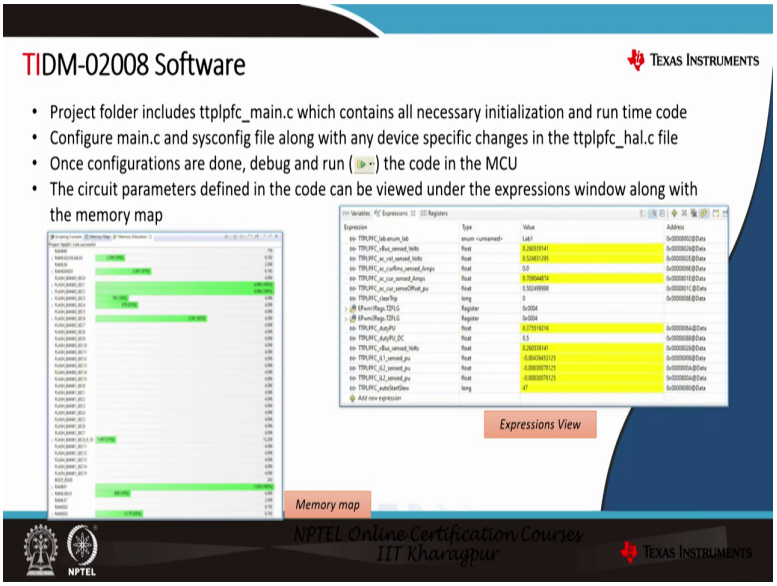
Then there is the second option the selection of a compensator designer as well as a selection of SFRA. You can directly load the SFRA as well as the compensator designer using this sysConfig tool. You can also configure whether the control is running on C28x or cla there is an option to select adaptive time phase shedding and non-linear loop as well.

Then specifically in control loop design, you are allowed to choose whether you are operating at a voltage loop or a current mode control which was taught in the earlier classes as well. Same way compensator style you can select whether you are operating for a PI, or PID controller and all number of poles and all zeros that you are using.

The SFRA that you are using whether it is injecting current whether it is pulling current or whether it is pulling voltage also you can set the current loop frequency voltage loop frequency at whatever rate that you are setting.

Then there are power stage parameters where you can configure your PWM the nominal voltage that you want to achieve, the power ratings, the filter inductor, the output capacitor that we saw, and lastly the voltage and current sensing parameters which will be used for protection purposes. So, you give a high-end and low-end value based on which there will be protection defined and the PWMs can be tripped using these values that you have provided

(Refer Slide Time: 12:04)

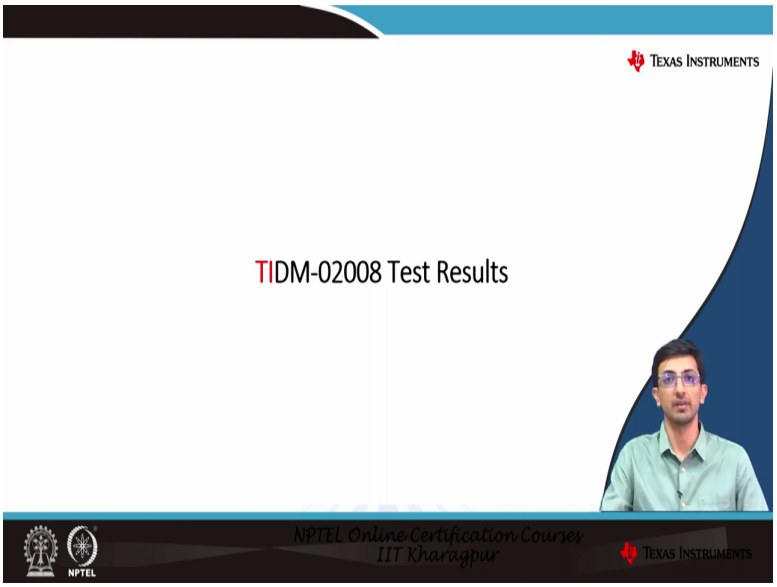


So, once you are through to this sysconfig the sys config pulls in the example code into the main dot c file. So, the project folder which included the ttplpfc underscores main dot c that contains all the necessary initialization and the runtime code. You can configure the main dot c and sys config file along with any device-specific changes.

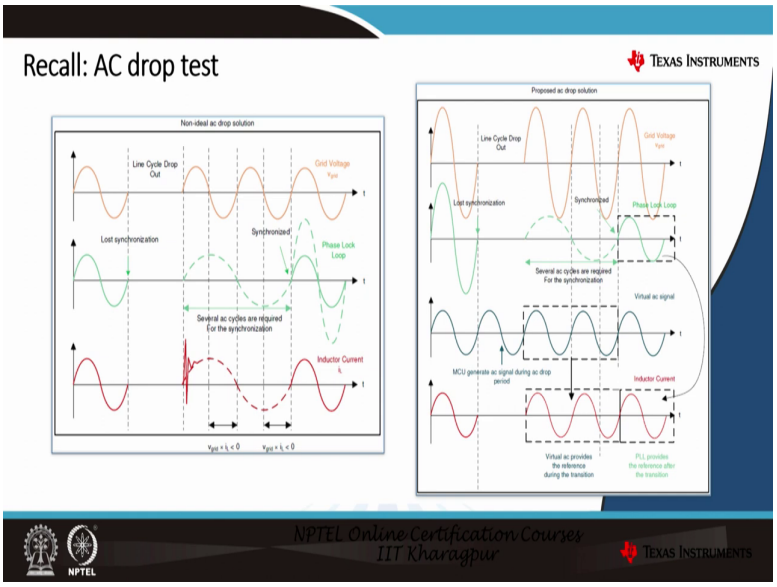
So, whatever project that it has provided is based on F28004 x. You can have any other device based on which you can have your device-specific changes incorporated in the hall dot c file. Once your configurations are done you can debug and run the code in the MCU.

The circuit parameters defined in the code can be viewed under the expressions window as shown here in the image also you can see the memory map. So, the memory map will indicate all what amount of memory is being occupied in the processor right now. You can enable the continuous refresh option in the expressions window which will give you a continuous update of all parameters that you are sensing.

(Refer Slide Time: 13:03)



(Refer Slide Time: 13:09)



So, now we will move to see the test result in part of it. Now before moving to the test result I just want to briefly touch upon the ac drop test that we discussed in the previous lecture. So, the AC drop test was that we had the grid voltage, and let us suppose the grid voltage drops out for a certain number of cycles.

Now in the usual case scenario what happens is that when the grid voltage recovers back the pll takes a certain number of cycles to synchronize it back. Because of that, the inductor current will be delayed than the grid voltage for a certain number of cycles which the pll uses for synchronization.

The proposed solution was that for the grid voltage, you need not use the pill for synchronization. But instead of that what we do is that during the initial phase, we generate a virtual AC signal similar to the PLL synchronization and during the line cycle dropout. We consider that the virtual AC signal is a reference for the inductor current and based on that reference we generate the inductor current.

So, now the total number of cycles which were lost in the synchronization phase can be recovered using the virtual AC signal. Once that is virtual once the pll is again synchronized we will again move back to the PLL synchronization level.

(Refer Slide Time: 14:14)

So, now we will go to the demo part of it. So, starting with the demo this time we will see the startup as well as the steady-state performance of our bidirectional totem pole PFC design. So, during starting up you can see the yellow-colored waveform is the voltage bus waveform. The green-colored waveform that you see is the line voltage waveform which is 230 volts and the pink-colored waveform that you see is the input current.

So, the initial phase that you see here is nothing, but the pre-charging phase. So, in the pre-charging phase of the capacitor, you can see the current abruptly changes along with that the capacitor voltage rises to a certain value.

(Refer Slide Time: 14:51)

Moving forward now, we have added a certain load which is a 1000 ohm resistive load on top of the design. So, there is a certain change with a current that you see that is moving from the no-load current to some certain load current and the voltage bus is also achieved.

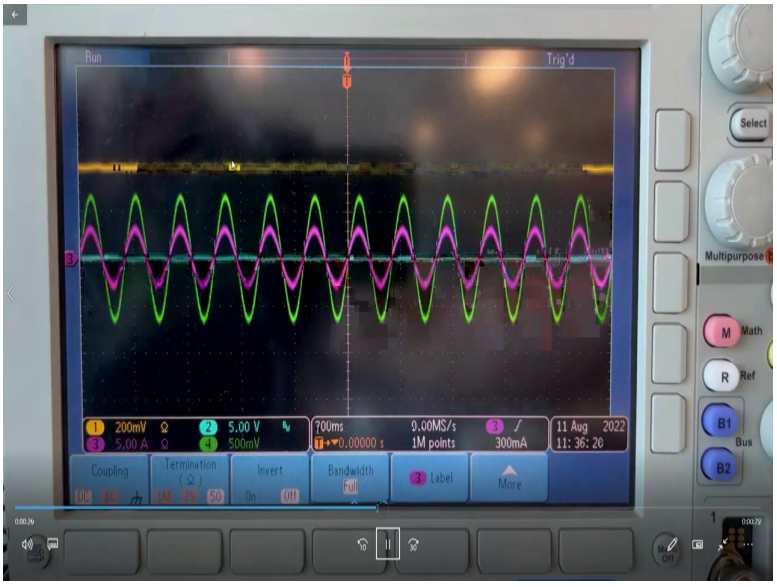
(Refer Slide Time: 15:11)

To see the transient response we have now changed the load from 1000 ohms to 300 ohms which are we have increased the load. And you can see the abrupt change in the inductor current, but with a certain transient response capability. So, there is no transient shoot-through that is observed in the current.

(Refer Slide Time: 15:34)

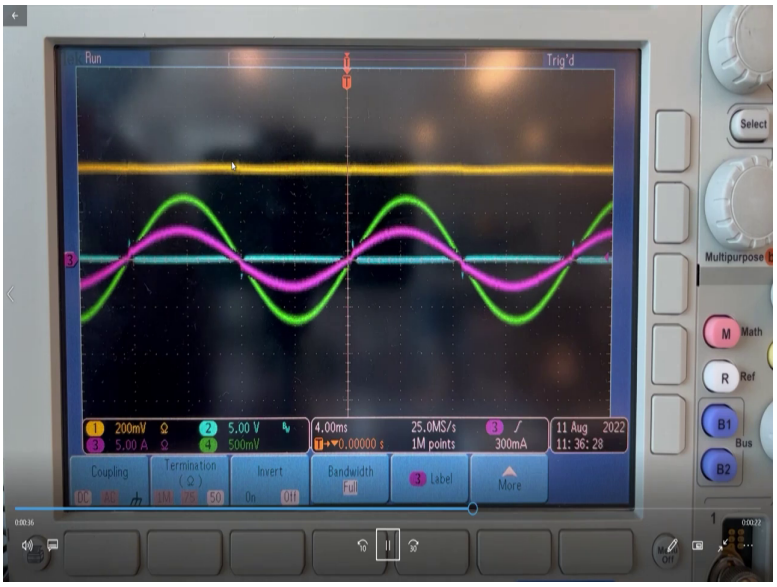
We can also see vice versa that the 1000 ohm resistive load is again recovered back from the 3; 300 ohms and in the same way, we do not see any shoot-through in the current this shows the transient capability of the response.

(Refer Slide Time: 15:55)

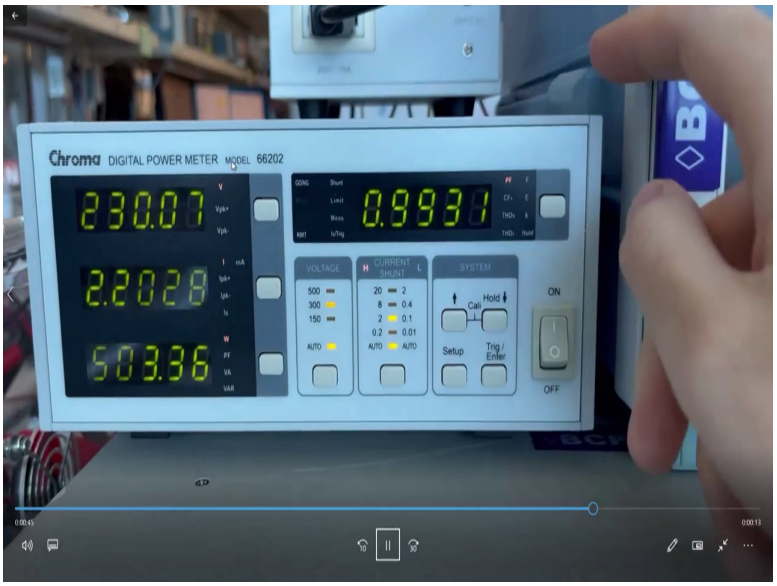


Lastly, we will zoom into the design and see what the actual power quality is. So, you can see the voltage and current waveforms based on which you can figure out the power quality which is quite nice.

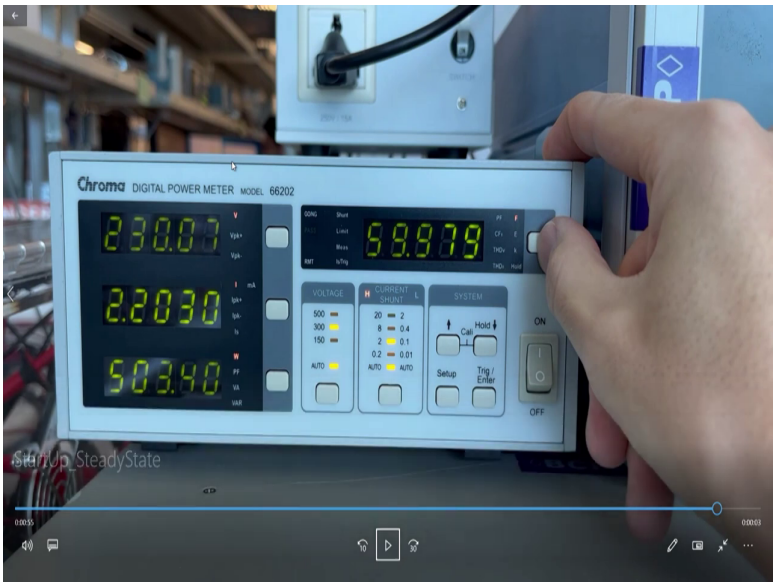
(Refer Slide Time: 16:03)



(Refer Slide Time: 16:13)



(Refer Slide Time: 16:26)

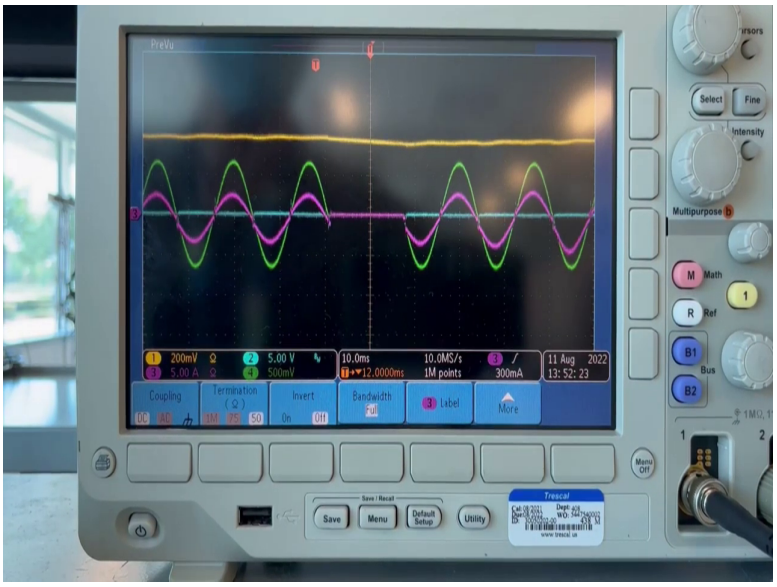


Now, moving on to calculations of power factor as well as current THD you can see the power factor is currently 0.99 and now we will see the current THD which is nearly 3.6. Also what you can see is the steady state load that we have provided which is just 500 watts for a rated load of 3.3 kilowatts. So, this design is operated at nearly 15 percent of the load which is a light load condition, and still, we are operating at less than 4 percent of THD at a very high power factor.

Now, we will see the AC drop test results in the same way as shown earlier in the steady state as well as the startup curve. Here also we can see the pre-charging state and the runtime state where we are adding a certain load to the system. The color coordination is still the same the green color is the line voltage and the pink color is the current waveform, the yellow is the voltage bus.

Now, the disruption that you see here is the AC dropout that we are doing. So, you can see at certain regular intervals the AC is being dropped back and again recovered almost instantaneously. This can be verified by seeing the inductor current waveform in line with the voltage waveform. There is no difference between the inductor current waveform timing as well as the voltage-current waveform timing.

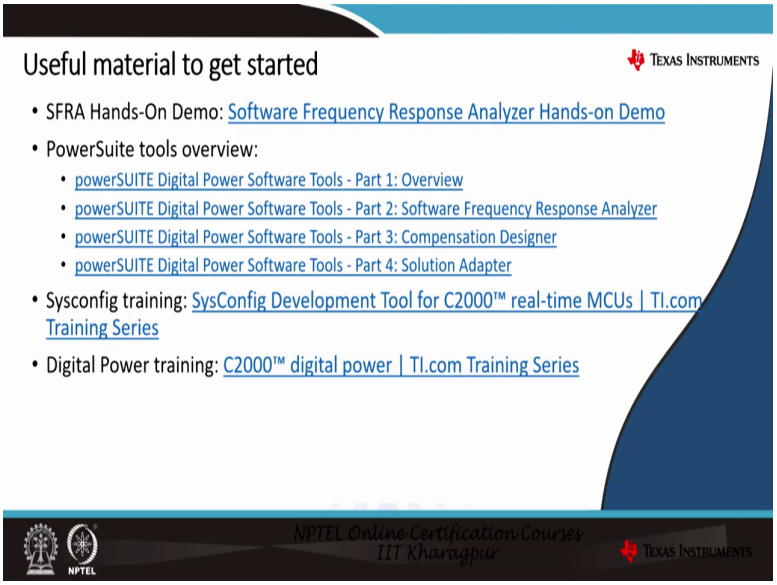
(Refer Slide Time: 17:32)



Here the instant where the voltage is going dropped and where it is recovering we are seeing very less amount of shoot-through in the current because of low load condition.

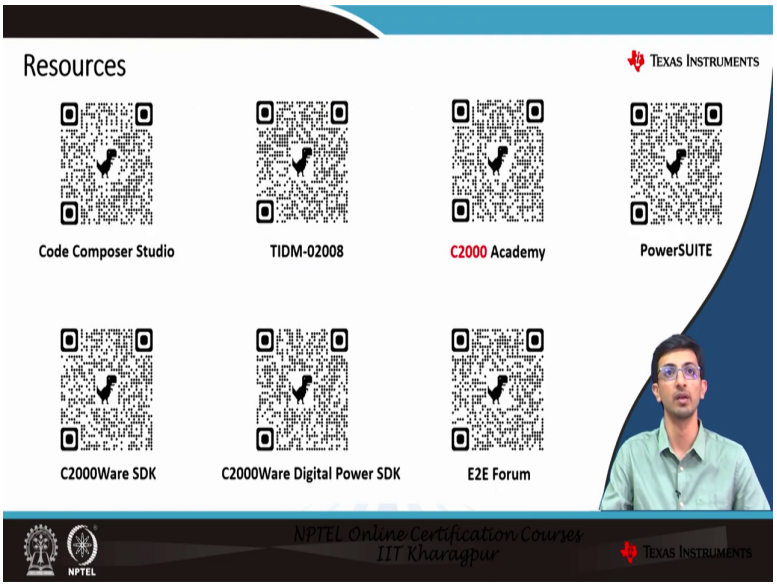
If the current would have been higher at the instant when the dropout would have taken place then we may have observed certain shoot through because of inductive capability. Also, you can see that the voltage bus has a little bit dropped which was expected, and again recovered back when the line voltage comes back.

(Refer Slide Time: 18:00)



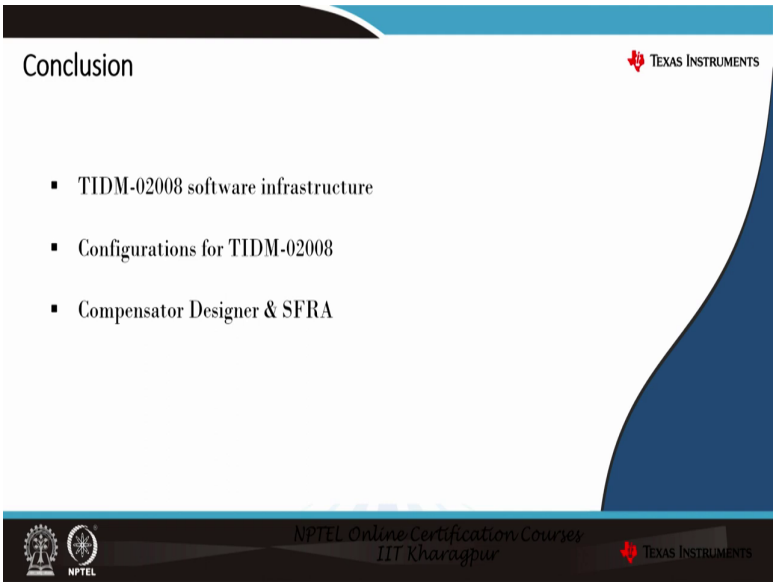
So, for this TIDM-02008 the useful materials to get started will be the SFRA hands-on demo. We have a good understanding and good video recording available for SFRA which you can go through. Additionally, we have a power suite tools overview which gives an overview of the power suit, then SFRA, the compensator designer as well as the solution adapter. We also have training on sysConfig as well as digital power which one can look through.

(Refer Slide Time: 18:24)



These are the resources that are presented that is the Code Composer Studio, TIDM-02008, C2000 Academy, PowerSUITE, C2000Ware SDK, DPSDK, and the E2E Forum. You can scan it and start taking a look at it.

(Refer Slide Time: 18:41)



In conclusion for this lecture, we went through the TIDM software infrastructure required, the configurations required for TIDM-02008, the compensator designer, and the SFRA. I hope you enjoyed the lecture.

Thank you.