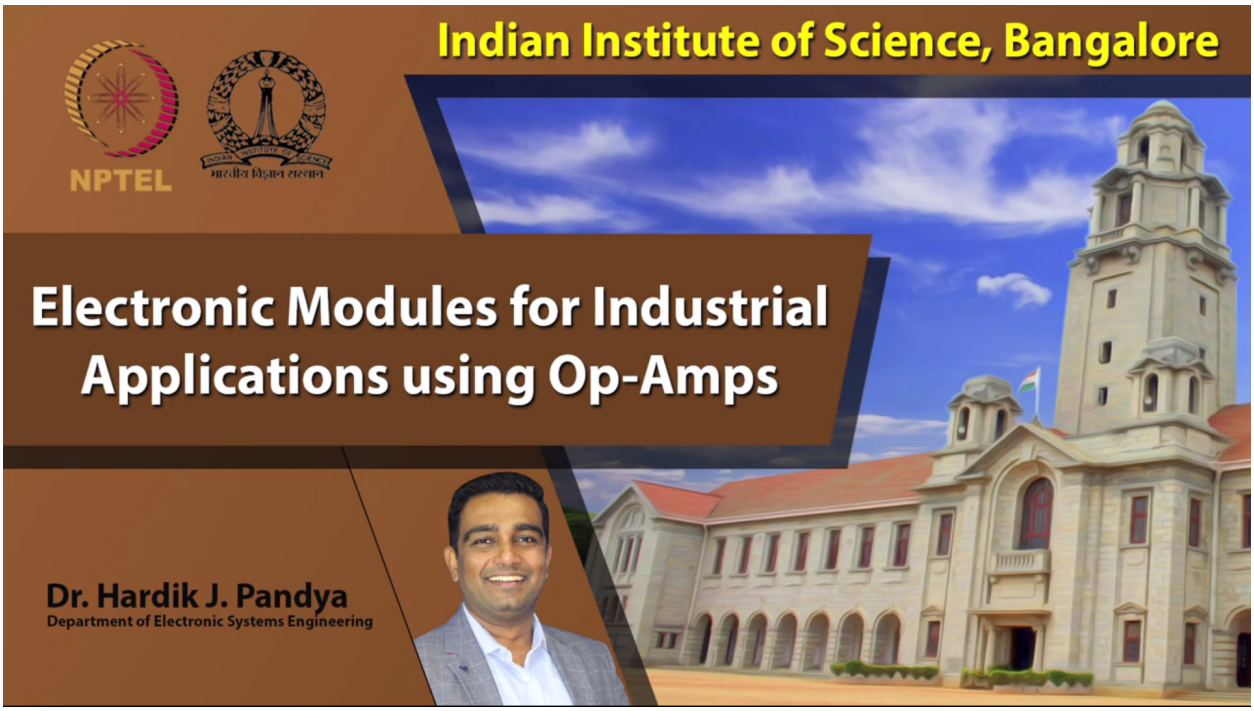
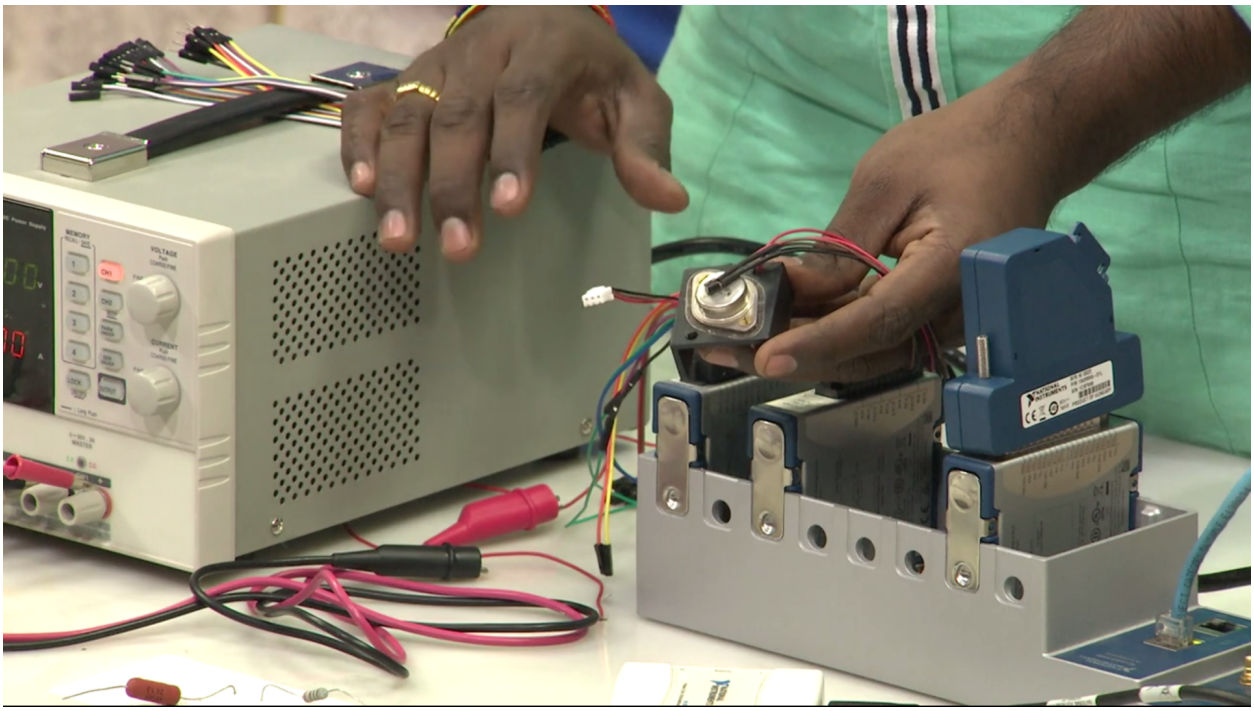


Male speaker: Welcome to the module. So now we will see the case study on using a temperature sensor as well as heater circuit in order to control and monitor the temperature on your plant. So now we will see how exactly you use a temperature sensor in order to measure temperature of an object, and as a result, we also require to have a plant in order to understand the exact behavior of a temperature sensor.

So for that, to explain you it with the case study of making use temperature sensor, as well as a plant and as we've already seen about a data acquisition, by combining all these systems, I would like to show you today about controlling the temperature of a plant using transistor as well as data acquisition system, as well as a controller which is building in a LabVIEW.

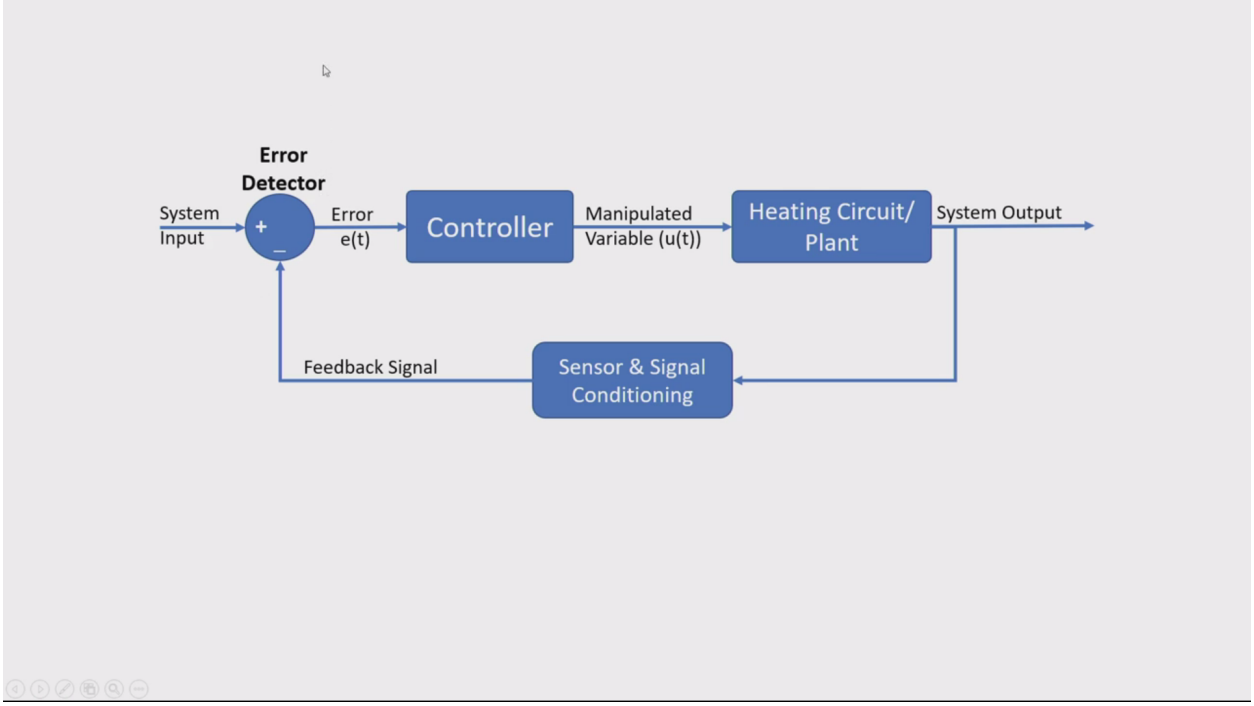
So in the last module, we have also seen about the data acquisition device, different type of data acquisition devices. One is our normal USB based data acquisition device, and see that compact deck, which has multiple channels and multiple slots, which is attached to the chassis.



So the whole idea of this experiment is that, we have a temperature sensor. So here if you observe, wee have a temperature sensor which is of LM35, which is interface or mounted on a transistor. So this is a power transistor, so 2N3055. So we will be building an electronic circuit in order to heat this transistor to a particular temperature. So what temperature, it entirely depends upon the user requirement.

So we are developing a controlling unit, so user has the ability or user flexibility in order to maintain the temperature on top of this transistor, but for that we are required to have driver circuit or excitation circuit in order to make the transistor to heat. So in order to monitor and in order to control the temperature on the transistor, we have mounted this temperature sensor, which is LM35.

So this temperature sensor measures what is amount of heat available and what is the required, the set point as decided by the user, and based upon that the control will understand how much amount of input to be given to the plant, such that it can maintain or it can able to reach to that particular temperature.



So if you recall, our complete closed loop control system. As the user intended to understand about system output. So whereas the output in this case is nothing but your temperature, and the temperature is not some kind of electronic unit. We require to have a sensor which converts from a temperature to an electronic output. But moreover, sometimes it is necessary to have signal conditioning circuit. The reason is that, in order to match the input units to the get off the -- system input unit to get off the output of a sensor.

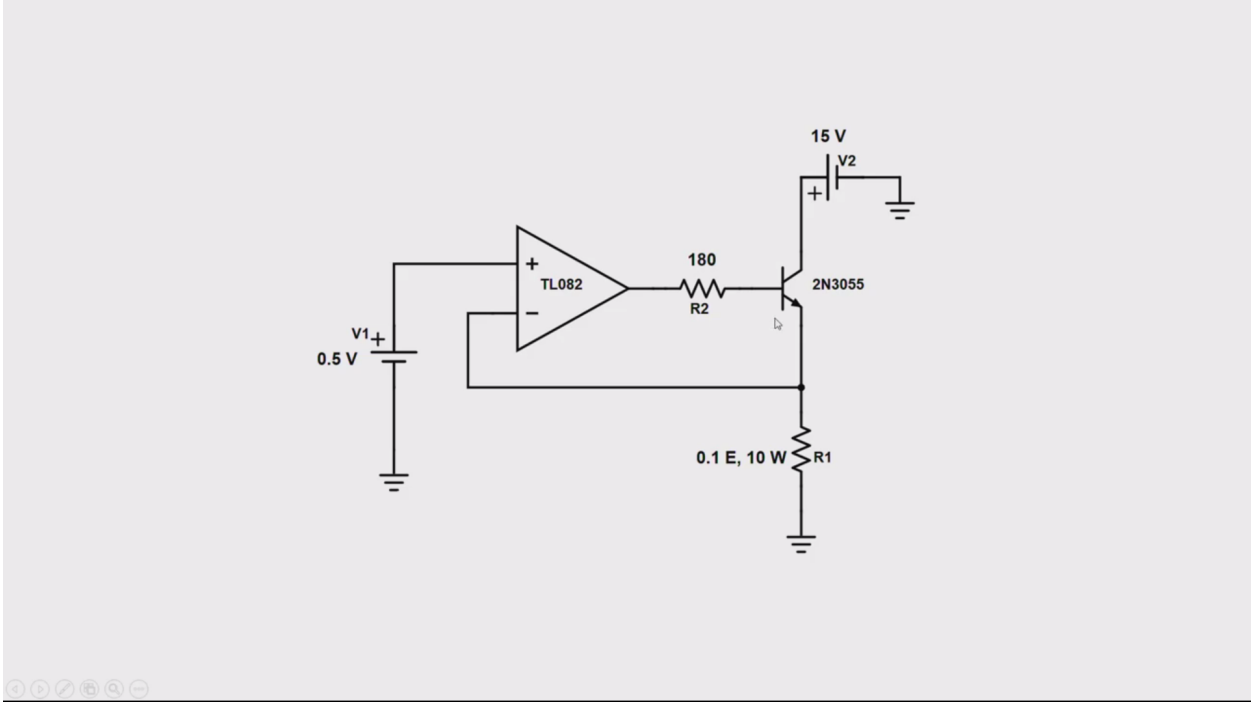
So in this case, if you clearly observe, we have an error detector, which the input is connected to this positive terminal of error detector, and the negative is connected from the output of sensor, as well as signal conditioning circuit if required, such that this particular block will measure the difference between the input as well as the output signal, but both should be in the same units. In this case, both are off electronic units. As we are developing the complete system using electronic unit, both should be understandable by the electronic system. So that’s the reason we have realized the temperature as well as input in terms of a voltages.

So the system voltage is nothing but what is amount of voltage required, what is amount of temperature it is necessary in order to maintain the temperature for the heating plant. For example, like say, if I consider the mapping factor or the scaling factor as 1 volt per 10 degree centigrade, which means that if I provide system input as 1 volt, the system will understand that the plant should heat it to 10 degree centigrade.

So it depends upon what is the requirement and what is amount of heat to be generated on the heater circuit, the user has to be specific at the input site. So based upon that, based upon the current temperature available on the heater, the sensor will understand and gives an electronic output voltage, which is fed back to the input of the system. As a result, it depends upon how much amount of input signal is available and what is the output temperature, it will measure the error.

This error will be, given as an input controller, so generally, you can use on/off controller, P controller, PI controller, or neural network base of fuzzy logic base, any kind of a controller which will understand the error component and will generate a manipulated variable such that, it maintains or it controls the temperature on the plant, the temperature on the plant is equivalent to the set temperature by the user. But unfortunately, sometimes whatever the manipulated variable given by the controller may drive the plant or sometimes it may no. So in order to make compatibility with respect to the output of the controller to he heating plant, we require to develop a driver circuit.

So in this case, as I have already told, we are using, we are generating a temperature on a transistor. So we require -- and whereas the controller which is nothing but a LabVIEW plus data acquisition device in this case, the data acquisition device cannot provide you enough current in order to heat a transistor to the required temperature. As a result, we required to build a triggering circuit or drive circuit which can act as an input from the controller and takes the power directly from the power supply and allows the plant to heat to certain temperature.



So for that, we will be using this particular circuit where it uses an operational amplifier. It need not to be TL082, it can go with 741 general purpose operational amplifier too, and this is a 2N3055 which we have seen just previously, which contains a temperature sensor which is mounted on it, and it is powered with the +15 volts. The base of the transistor is connected to the output of operational amplifier using a current limiting resistor so that it will not damage the transistor, and to the negative feedback, we have connected the emitter of the transistor. As a result, if you recall the complete working of the circuit, it maintain, the output voltage across this R1 resistor to be same as the voltage connected at positive terminal. This is because of virtual concept.

So as the operational amplifier virtual concept says that, the difference between both the terminals should be always equal to 0 since the input is -- positive terminal is connected to 0.5, it tries to maintain the native terminal also be at 0.5. As a result, the drop across R1 will be 0.5. So since the R1 is known to us, which is 0.1 ohm 10 watt resistor, and the voltage connected is 0.5 volts, so the current flowing through the emitter of this transistor would be somewhere around 0.5/0.15 amps. So as a result, out of 15 volts -- so since the emitter is connected with R1 resistor, the emitter current will be 5 amps, and based upon, the principles of transistor, we know that the collector and the render current is approximately the same, which is with a proportionality constant as alpha. As a result, the emitter -- the collector current will also be approximately of 5 amps.

But when we see the input is connected with the 15 volts, but the drop across resisted is 0.5, but what about the rest of 14.5 volts. So that 14.5 volts as well as the current flowing through this transistor will generate heat on top of the transistor, and this will be the heat which is generated on top of a transistor will be measured using a sensor and fed back as an input to driver detector. So as a result, it generates an error and it continues the understanding based upon the intelligence that we created on the controller, it will try to understand how much amount of input voltage to be provided at the input side of the driver circuit.

Now if you carefully look into the working of the circuit, the current flowing through this R1 resistor, entirely depends upon the voltage drop which is being generated across this R1 resistor, and moreover, the voltage drop across this R1 resistor is due to the voltage connected at the positive terminal of an operational amplifier. Higher the voltage, higher the drop across the R1 resistor, as a result higher the current flow through this, but lower the voltages, lower current so lower heat distribution across the transistor.

So the complete working, as we've already seen, now we will see how to build the controller logic using LabVIEW and how could interface, the driver circuit, the plant, as well as sensor to the data acquisition device and how can we understand the working of different controllers is the today's task.

So in LabVIEW I will be developing error detector and the system input is also be controlled by using LabVIEW user inputs, whereas the controller can also be realized inside the LabVIEW environment, whereas the plant as well as the sensor will be externally connected to the data acquisition system, where it acquires the data and displays on the LabVIEW device.

Now when we look into the LabVIEW, so it contains front panel as well as a block diagram. So it is similar to that of our day to day life systems, what we see in our laboratories too. So the front panel really is similar to that of our functional generator power supplies where user will have -- user can have access only to either the knobs can visualize what exactly is happening, the buttons, charts, if it is oscilloscope, it's like having a graph or chart on the front panel, but the complete logic will be inside the block diagram. So during the operation the user will not have any access on the block diagram, but user can manipulate things in front panel.

So it is similar, the structure of LabVIEW is also similar to that of our real-time environment. So even in case of our normal systems if you observe, the electronic modules will sit inside the system. When the system is under running condition or operation, we don't have any access to manipulate or modify any items, but user can change the knobs to understand in order to

create a curses inside a graph. So everything can be controlled by the user end. So the same even the LabVIEW platform looks in the similar way.

So now what we require, we have to connect our controller unit, one is error detector as well as other one is controller in a LabVIEW environment. Then we have to interface to the real world. Now if you see -- so as our application is similar is similar to temperature monitoring application, I will take one input, the set point, so this I will emit as set point, so like we are doing -- so I am changing it to control, so I will also provide unit display as well as digital display, so that whatever the required temperature that user require can be set at this point.

Now you also require to have the output system, so now I am generating the temperature output, actual temperature. So this is set point, what is the exact temperature that the system has to be. There is this actual temperature, what is the temperature right now the system has, and so this should be an indicator, so I'll change it to indicator. Now if you observe. If I directly connect from here to here and when I run the process, we can see that, say, for example 100, automatically the temperature on the indicator will also change. So this is the difference between in control and an indicator, so the user can only have an access in order to change any parameter, but whereas the indicator can only display, but the intention was not to directly connect it here, our intention was to develop a logic algorithm, which can monitor and control the temperature on the plant.

So for that, what I'll do is I'll go with the control system toolbox, which is nothing but control and simulation toolbox, where I'll create a control loop. So this a control and simulation loop, and in this, I have to build an error. So first step in our circuit is we have to build an error detector. So I'll build an error detector now. So for that, I'll take submission block, which is already available in signal arithmetic of control and simulation toolbox. So I just rename this to error amplifier, because the operation of the summer in this case is similar to that of -- actually it's a different amplifier, it's similar to that of our error amplifier. So not to confuse, I'll change it to error amplifier, our error detector.

So one input to this error detector is connected with a set point and another input, the native terminal should be from the sensor, but in this case, the sensor is externally connected. So we require to create a data acquisition module in order to acquire the data from the real world. So for that, what I'll do is that I'll create a measurement I/O. I'll create the data acquisition modules in order to acquire the data, so for that I'll create new VI, so in this case, I'll go to -- I'll create a channel and then I have to read the channel, then I should display it, but which channel. So for that, I'll create a physical channel, change to control. So I'll rename this as sensor channel.

So if you observe, the system has been connected with two different modules, one is our C DAQ, other one is USB DAQ. So in order to understand about the configurations of USB DAQ as well as C DAQ, what we can do is that, as we've already seen in the last time, we can go to NI MAX, which is a measurement and automation explorer, where we can have complete access of the particular data acquisition modules connected to the PC. So in this case if I go to devices and interfaces, here we can see one is USB DAQ which is named as Day1, and other one, when I look into the network devices, we have C DAQ 9189, which is also named ass CDAQ9189-1D0A882. So we'll be using these two.

So in this case, so I'll be using -- since it for the acquisition, I'll be going with C DAQ module 2. So now I can see what are all the modules has been connected into this module. So we'll be accessing NI 9207 in order to acquire the data from sensor. So in NI, which is a modulate 9207, if you look into the P notes of this particular devices, so this is the particular system what we have. The first set of -- if you see here, we have AI0 to AI7, which can make differential output and it can also work for both acquisition of analog signal as well as generating acquisition of current, as well as it can also be used as a power supply, but it cannot power enough current or enough voltage in order to drive any circuit.

But since our intention was to acquire the data from the sensor, we will be connecting the sensor to AI0+ and AI0- which will be in voltage format. So what I'll do is that I'll create a sensor channel with C DAQ module 8, so C DAQ 9189, yeah mod 8, but AI0, so this one, data operations, make current value default, and our intention is to acquire analog data, analog voltage. So as this particular block can be also used for acquiring analog input as well as analog current as well as voltage. So I'll be selecting it as a voltage.

Then so the next step is creating an error, create, control, so that it gives an indication in case if any particular block throws an error without any problem during the execution. Custom scale, task in, and task out will be connected here. So error out I require, change to indicator, and replacing with this, and this particular block is my data, so create indicator. So I'll be creating a pin out for this, so this is sensor channel in, this is error in, whereas this one is error out and this particular pin out will be connected to the data. So which data is this, sensor data, temperature sensor output.

Now let me save this file, save as. So I am going with, I'll be replacing everything in the desktop, I'll create a folder, temperature acquire. So I save this file, then I'll be calling this file. Since I'll be using the same file as sub VI, so just for our sake of interest, we can also change the -- edit the icons as well. So I'll replace this with this one, let me delete this part. I replace, so this particular block I'll replace with temperature sensing, whereas this one is AI. So since it is temperature sensing value, using some graphics, we can also

indicate the functionality of this particular block. So, say, temperature, so here we can see different types of blocks available, which will not fit in this space. I'll take this. So I'll write a text showing as C, okay.

Now let me save. So here we can see, but whatever the output that we get -- so we have to provide this channel. So create control, I am creating a control. So since the default is this particular channel, it will automatically save. Now what is the mapping factor that we consider for the set point. If you recall the mapping factor that we use of 1 volt per 10 degree centigrade, meaning if I provide input as 1 volt, which means that system will understand, the system has to operate at temperature of 10 degree centigrade or control, the system should have a temperature of 10 degree. But when you look into the sensor, LM35, the sensory of sensor is 10 millivolt per degree centigrade.

So in order to map the scaling factor, in order to match the units of the output from the sensor to the setpoint, we have to provide a gain of 100, such that both will have same mapping factor, which is 1 volt per 10 degree centigrade. So for that, what I'll do is that. I'll take a multiplier, I'll multiply the output of the sensor data with 100. So let me create a constant, dbl constant, which is of 100, then this I'll be connecting it here, so that whatever the data output that we are getting, we'll have the same units as input.

Now what we have to do, the output of this error detector has to be interface at your controller. So as we have discussed, we can design on/off controller, we can design P controller, we can design PI controller, as well as PID controller, in this case I'll go with either P, PI controller, but I'll also how you how can I build PID controller using this. So just we'll go to control and simulation toolbox once again, and then we'll go to simulation. Here if you see, we have different types of signal arithmetics where we can create a gain, and since we require P + I + D in order to construct a PID controller, as well as we also require a differentiator as well as integrator, so we can go to control linear systems, and we can take integrator as well as a derivative component here and output of the error detector, which is nothing but output, which measure the difference between the setpoint and the feedback signal, output signal. So by connecting the output of error to the input of all the games and by adding an adder circuit, we can realize the either P or PI or PID controller. So provided depends upon what kind of a controller that we are realizing, other particular gain should make it as 0.

So we also required another arithmetic. I'll copy and past this particular error detector, but in this case we require 3, 4 inputs, and this should also be positive, or I'll make in this way, this is positive, this is okay. So such that this IC adder, so connected here and this will be connected here. But the problems comes here is how do we manipulate the gain from the front panel. So for that what we can do is we can double click on the gain, instead of

taking parameter source from dial up box, we can make it as a terminal, so that -- see P-Gain -- so that we can create a control at this point, silver. Okay I'll go with numeric, horizontal point slide. I will also create digital display, so that easy to understand what is the value of a P-Gain, but this should P-Gain. So this particular block, I can connect it to the input terminal. Similarly, even for I-Gain and even for D-Gain.

So if I make P-Gain as greater than 0 and I-Gain and D-Gain as 0, 0, that means the particular controller that we are realizing is of P controller, but if you make only P-Gain as high and I-Gain as some value but D-Gain is 0, that means that particular is PI controller, so with this we can understand how can easily we create P as well as, PI as well as, simple I or combination of P and I or combination of P and I and D, but we cannot use a simple D controller, because when the error is a constant, the derivative gain will become 0, so which the controller will confuses, what like the output given by the controller will be always 0.

So what we have done, we have crated a complete system, with error detector, setpoint connecting, as well as the sensor data with signal conditioning system. So this is setpoint, the user can provider -- this is like comments -- the required temperature for the plan to operate, and whereas sensor data with amplification, temperature sensor data with amplification, and whatever we get is error output P + I + D controller realization, and output will be manipulated, but this manipulated variable should be given as an input to driver circuit, but right now whatever the system it is being generated will be -- the data acquisition device will not understand, how much amount of voltage to be given. So this has to interfaced to the data acquisition by using analog output modules from measurement I/O, which is similar to that of what we have created for acquiring the signal from the sensor.

So in order to do that what I'll do is that I'll create another LabVIEW file, I'll open another LabVIEW file with containing the data about from which particular channel the manipulator variable has to be given to the real world. So since we require analog output voltage, which is nothing, but generation. So I just created a channel, I'll be creating a channel control. So this particular channel will provide the manipulated variable as output to the real world. So from which channel we have to provide, so we can use Day1.

So from the NI MAX if you observe, we have seen different types of devices connected which is NI USB 6008 as well as C DAQ 9189. So for acquiring the data from the sensor, we are using C DAQ, whereas for generating the signal to the real world, we'll be using 6008. It is not meaning that the same device cannot be used, but if we have connected with the same analog output module to the same data acquisition device, even the same chance can be able to generate as well as acquire the analog voltage signals, but since

right now I have not connected with analog output and we also have USB data acquisition device, I plan to show you the acquiring using C DAQ as well as generating using another data acquisition which is USB device. Similarly, even with USB device, it is able to acquire as well as generate the same signals.

So when look into the pinout, we have a different types of connections here. so when we look into that, one side, it has a total of 16 analog pins on one side, where as another 16 pins on the other side. So one side, they are written with the name of analog, other one as written with the name of digital, which means that what are the analog that we require can be connected at this side, and digital blocks can be connected as this side, and this is connected to the PC using USB cable.

Now when we look into the pinouts of this particular device, here we can see the pinout configuration of complete device. So it contains -- the first channel is ground. It contains a total loss, 7, 8, total of 8 analog channels. Out of 8, 4 channels can be used as differential input whereas remaining 8 can be used either a differential -- used as single ended input. So in this case, I'll be using AI0+ as well as AI0-, such that the AI0 pin, the output -- sorry, since we require to generate on output signal, we have to go A0O. So AI indicates analog input, AO indicate analog output. So that means that it has -- it not only having analog input, it can also generate a signal by using two channels which are at 14^{th} pin and 15^{th} pin, and on the other side if you see, which is from 17 to 32 pin, it has a total of 8 digital channels, along with that it also has power supply to the sensor, which is 5 volts, as well as it also has voltage, which can generate voltage of 2.5 volts and ground. So that means the sensor, the voltage required for the sensor can be easily powered by using this 5 volts tool.

So what I'll be doing is that since we understood the Dev1 can support for analog output generation, I am using the USB DAQ, which is named as Dev1 in the PC will be used to generate it. So in order to generate it, the output from the manipulator variable has to be right. So I have created a task, which contains this physical channel, then I'll create error control and output of this will be connected here, then create indicator. So the detail explanation of this particular block can be easily seen here by pressing Ctrl+H or going to the help window. The detailed help -- this is just a summary and if you require the complete detailed of this particular block, you can click on the detailed help. Here it gives information about the module, what we have selected in LabVIEW.

So in this case if you see, the bold indicates the compulsory terminals to be connected, whereas -- the bold, it is required, until and unless these particular terminals are connected, the device cannot operation, the device cannot work, whereas the hidden text if you see, it indicates, it's

recommended or optional, whereas other sets which is not even bolded as well as which is not even hidden, these are nothing but recommended. So even if you connect, it will work even; if you don’t connect it will work. But when you see that, it doesn’t mean that it will not pass any data, but default it contains some particular value.

So, for example, say when we consider auto start, auto start specifies this VA automatically starts the task if you do not explicitly started DAQmx start tier. So in case even if you connect, it will work; even if you don’t connect, it will work, but if you don’t connect, it will consider some default settings, which is already available and it will work like that. So since we require to generate analog output, single sample, dbl way, so create constant true, then data. So I have to write the data. So what I’ll do is that I’ll create control, so whatever the data that I pass at this particular block, the same will be generated at the AO terminal, which we have specified here in the USB DAQ.

So let me rearrange, so this is output data now just data operation make current value default. So similar to the previous one, I will also give a pin connection, so that the same VI can be used as a sub VI in another program, and let me go to layers, delete it, create a template with since it is sub VI and this is for manipulated variable, which is indication of writing a data. So I have generated it. So this is manipulated variable generate. So this is also done, but this has to interfaced to -- this has to be called in main VI. Now I’ll go to -- I’ll create a sub VI, I’ll select a VI, so which is MV Generate, I connect it here, so the output whatever it is being generated has to be sent to -- so this is output data, so this is an input, which should be connected at this point, yeah, so this I will connect it here.

But we cannot directly connect, the reason is that, since it is simulation, the system can even provide infinite amount of data as an output, but in realize, the data acquisition device cannot provide very high magnitudes. So each device has limitations with respect to the operating voltages as well as current. So similarly, even with USB devices, the output voltage it can generate is of 0 to 5 volts. So a simulation can go up to more than 5 volts, even with the positive as well as the negative in order to not to have any conflict between the values, we can create some kind of threshold such that if the input voltage is greater than particular value, send the output as a maximum.

Moreover, if you recall, the driver circuit, the maximum voltage in order to not to acquire more amount of current not to damaging the circuit, so we can restrict the input voltage connected to the driver should be not more than 1 volt, so that by creating some logical icons here, logic saying that if the input voltage is greater than 1 volt pass the maximum here, such that the generated output voltage will be not be greater than 1 volt, which both the driver circuit as well as the data acquisition device will be in a safe zone.

So let me add all the required components. So now if you see, the complete system is connected. So I have added with the logic whichever I told you, so that when the input voltage is greater than 5 volts, since the data acquisition module or USB data acquisition module cannot generate a voltage more than 5 volts as well as the less than 0 volt, so voltage which shows more than 5 volts will be shown with 5 volts. Similarly, any voltage lower than 0 volts will be shown as 0 volts, that particular logic has been designed. Moreover, if the simulation generates more than 5 volts, the system will automatically switches off.

Moreover, any stop button pressed from the output side from the front panel, even the system will switch off. Similarly, I have also added the display unit in order to visualize both the input as well as output signals in the same graph, so that it understands how much time it is taking the system to respond with respect the controller could able to reach temperature on the plant to the required setpoint. So with this the software side we have done. Now we have to make the connections in the circuit board.