Lecture –31

Design of Speed Control of a DC Motor using DAQ: Part 1

Electronic Modules for Industrial Application using Op-Amps

Welcome to the module. So, in this module, we are going to see working of a Dc motor, that means the controlling of a Dc motor, using data Acquisition device. In the last module we have seen, design of different signal conditioning circuits, as well as a driver circuit, that is required for the motor to drive it, using operational amplifiers. At the same time, we have also designed, different controllers, P PI controllers we have seen, interfacing the P PI controller to the DC motor in order to drive the DC motor to a required set point, and we have also observed, that depends upon what type controller we are using it,

Refer Slide Time (1:18)

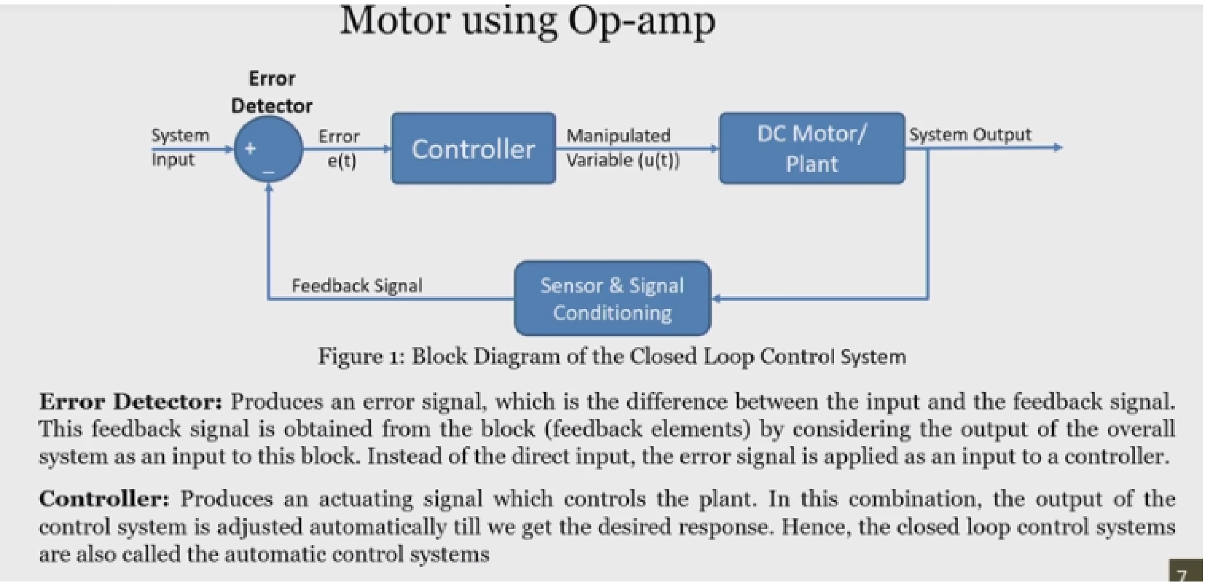
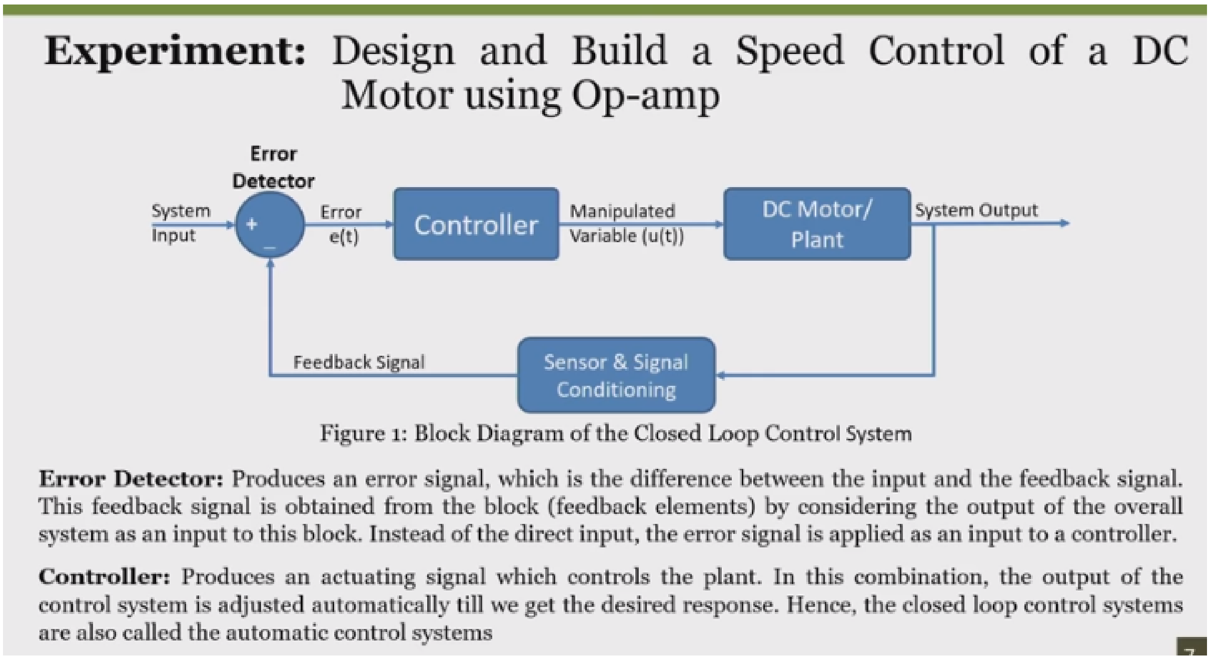


Figure 1: Block Diagram of the Closed Loop Control System

And we have also seen, that depends upon the game, that we have said to the controller the offset voltage is the error the offset error. The time it is taking to reaching it set point is always changes. Also, we understood, that the working of a closed loop control system, and the design of converting the encoder pulses output, which are digital into analogue output way, which linearly varies and maps, with respect to the system input. But in the previous thing, we have considered, the DC motor RPM has frequency divided by fourteen. But we do not know, whether it is going to give fourteen pulses per one revolution or seven pulses per revolution.

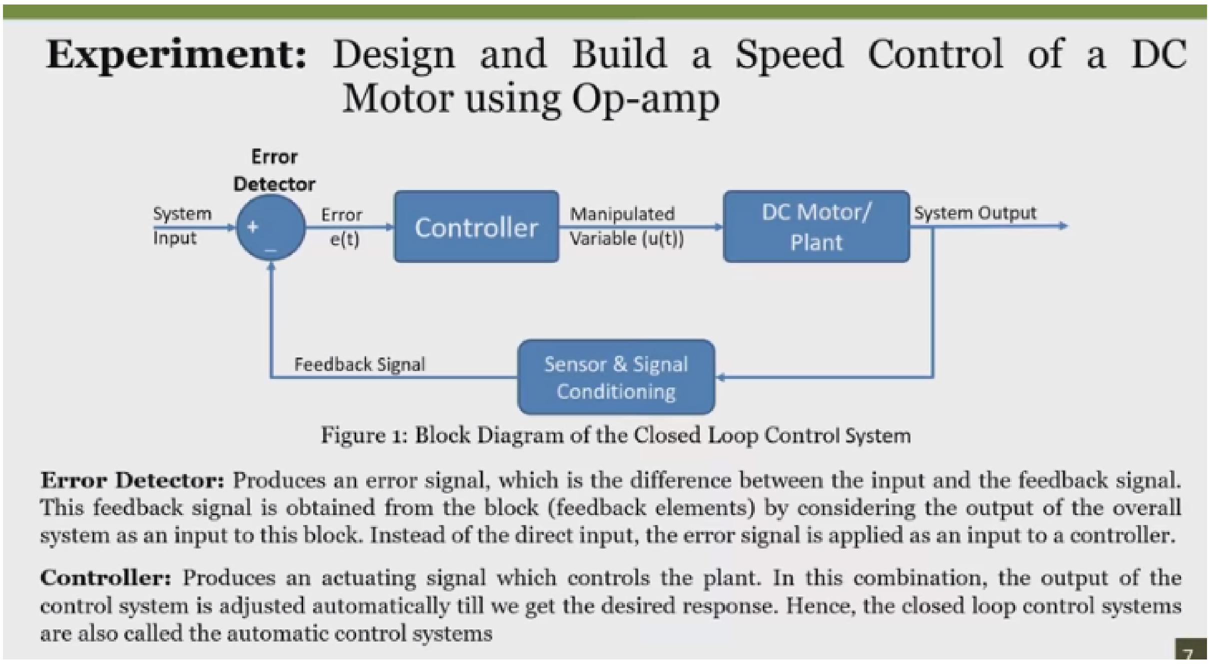
So, in this model what I am going to show you is, replacing a complete analogue controller, which we were designed, using operational amplifiers, with a data acquisition modules. So, we have seen, different data acquisition modules, their working the difference between the other data acquisition modules and everything. In this case I am going to use USB NAUSB DAQ. I ‘ll be interfacing Dc motor using NIUSB DAQ. Of course, as, any microcontroller, as well as data acquisition devices, cannot support the required current for the DC motor to drive it. As, we have already designed a signal conditioning circuit or driver circuit for the DC motor to drive, based upon the signals received from the micro controller, as well as, an operational amplifier, we’ll be using the same driver circuit. Interfacing the motor to the driver circuit, but the input to the driver circuit will be from data acquisition device. But how the data acquisition device will a provide a signal to that, because, since the data acquisition device will be interfaced to our PC. The PC the algorithm, which is running in the PC, which is being developed by using national instrument software lobule. So, we are going to design the same controller, But the controller will not be on board, it will be on software. So, that's why this is called software in loop simulation.

Refer Slide Time (3:45)



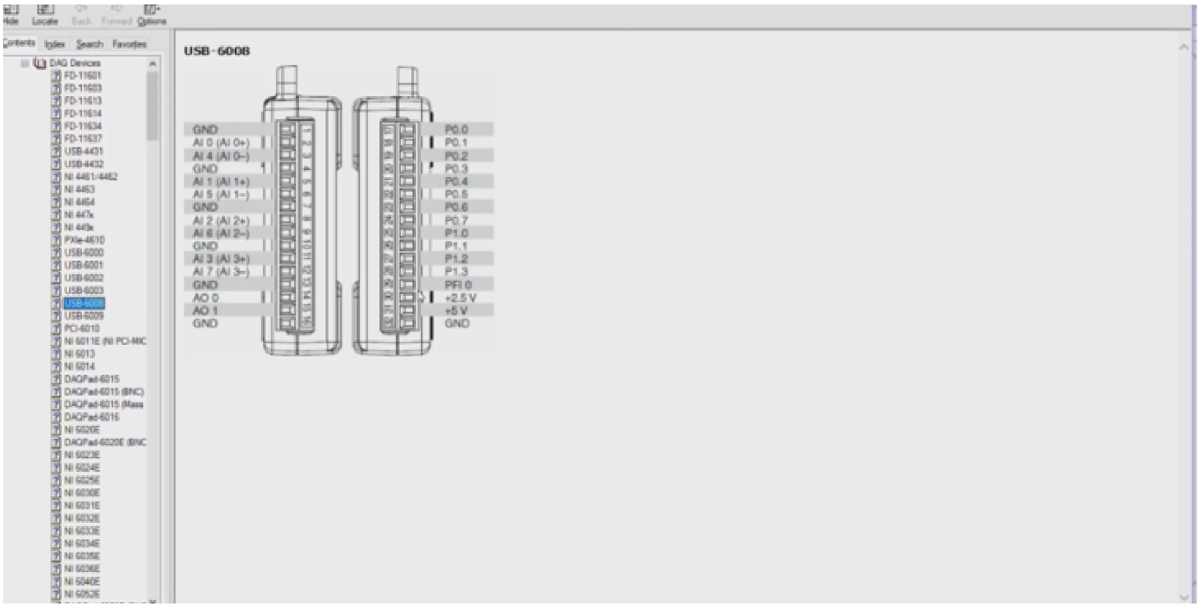
So, in a complete closed loop system if you look into the screen, if you look into the screen, in a complete closed loop system, the system input will be set from the PC. Error detector will also be set from the PC, and the controller is also, designed and developed a in a labue software, which is also, in PC. But this interface will not be direct right now. This will be connected through a USB DAQ whatever the signals based upon the controller based upon the ACE, that we are digitally providing, that will be converted into an analogue output by using data acquisition device, that will be given as an input to the driver of a DC motor, where DC motor is connected to the driver circuit. Then As, we already seen, the motor is already integra integrated or interfaced with the encoder, which gives a digital pulses, and since, we had using data acquisition device, which can also support a counting pulses as well as which can also support a digital pulses, as well as it can also acquire analogue input pulses. By looking into by acquiring the signal, either in a digital way or a counted way as well as by analogue you know, signal way. If you can understand how many revolutions or if you can understand, the rate at, which the RPM of the motor is rotating, such that, by we have a code or by writing an algorithm in an labue software to make our self-understand about how much RPM it is you know the motor is rotating, then giving it as an input to the negative at this point.

Refer Slide Time (5:20)



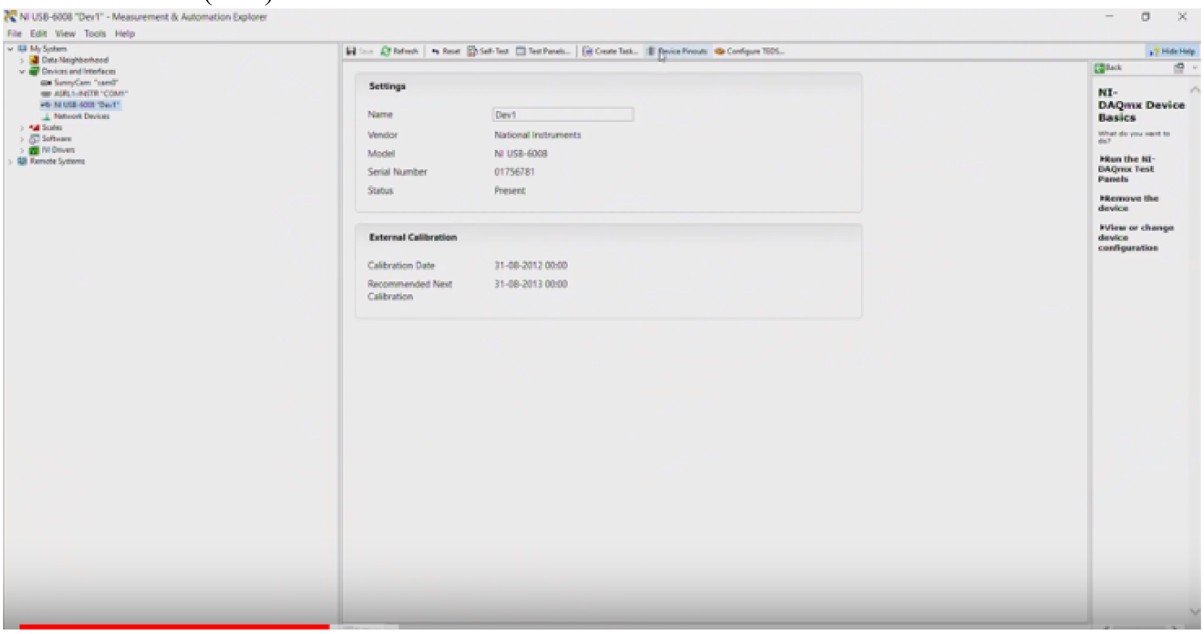
By proving the output at this point, we can easily realise the complete closed loop system by integrating the DC motor externally, and controller , which is sitting in a software side, and we can understand a complete working of understand the importance of operational amplifier , sorry a understand the importance of the controller , we can also, understand the working of DC speed motor control, in order to maintain the RPM of the DC motor by using data acquisition devices as well as by using a software. So, here if you observe carefully, this is this is our USB DAQ this USB DAQ NA6008. It is interfaced to PC using our USB cable, and if you see the connections to this DC motor. This is our DC motor which is connected to an encoder. So, this particular Part, the board, we have you know design our driver circuit and this is where we are interfacing the motor to the driver circuit. Right? Here we can see, whereas, when you look into that we know that how many number of analogue as well as a digital pins, that USB DAQ has. So, what we have done right now is, we have connected the clock terminal pin to okay. The encoder is powered by using USB DAQ. So, here we can see, this are the you know powering connection to the encoder, plus v_{cc} and ground. Whereas, the input to the driver circuit has been given in the 14^{th} terminal of USB DAQ. Which means that, it is an analogue output. Now, what we are going to do is, in order to understand how many number of pulses this DC motor is going to provide or give, per minute we will write a programme in a labue and we will acquire a signal will count the signal only for particular time duration. Let say, one minute or two minutes. So, that based upon how many number of pulses per minute we are getting gives you gives us a measure of RPM. Right?

Refer Slide Time (8:11)



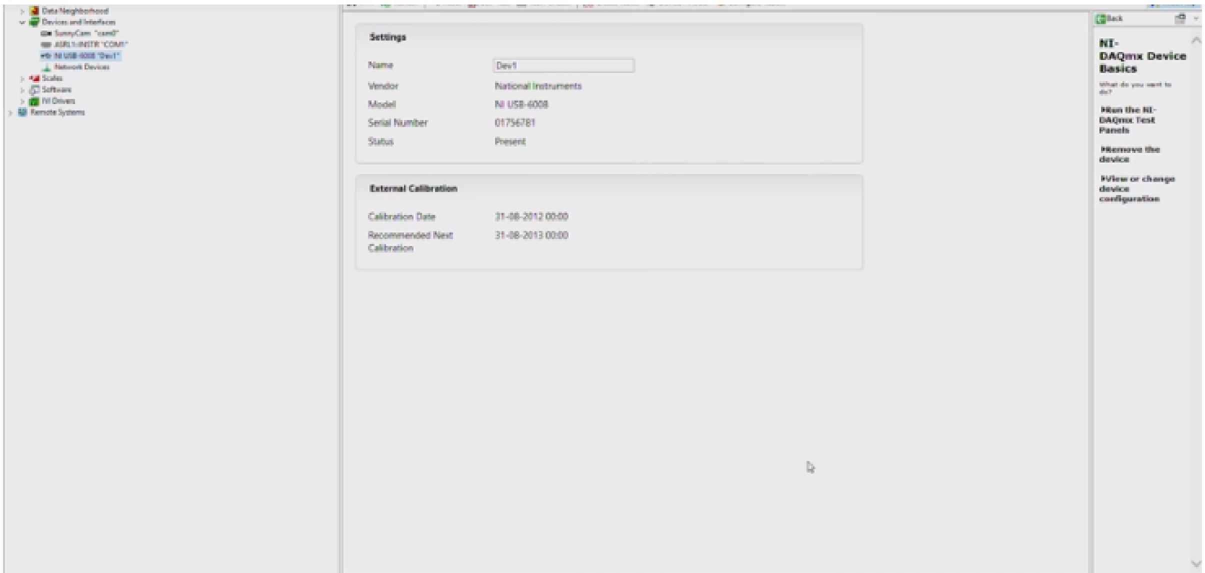
So, for that case we have to understand, which is our counter? So, if I look into the data sheet of USB 6008. So, here if you observe, we can see here, that pf5. the 29th terminal. So, it is our counter pin. So, let me connect, let me connect the clock pulse is, the output of the clock pulse is in this case c1 I am considering it. Either you can go with this c1 or c2.c2 I am taking it. So, I am giving it to a counter input. So, to understand whether it is counting to not? What I will do is that?

Video Start time (9:07)



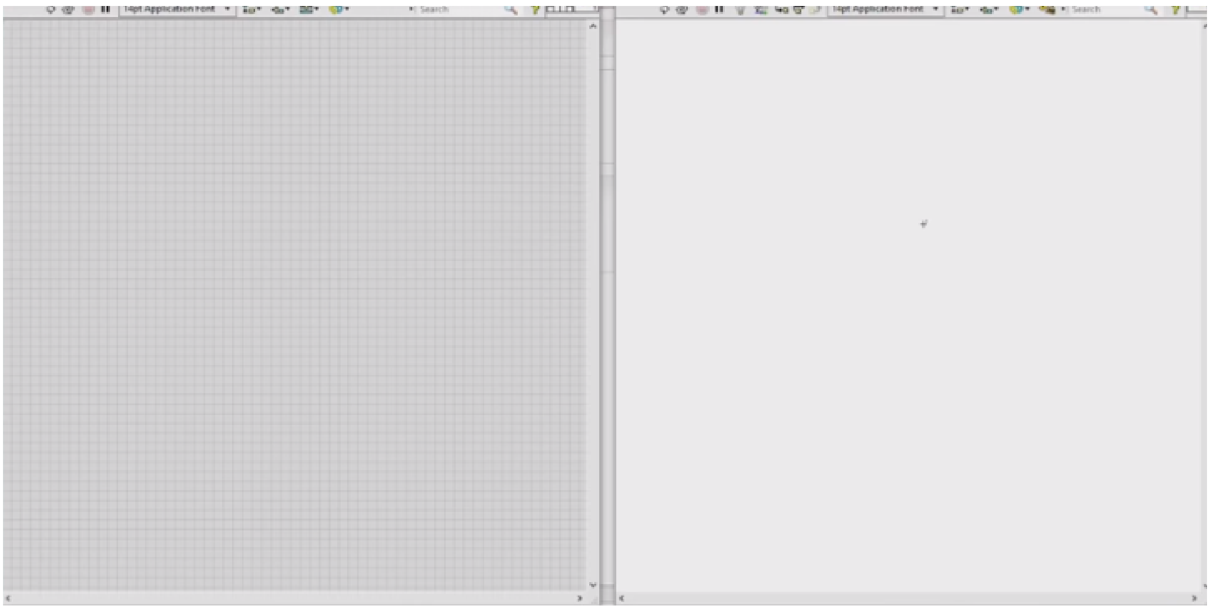
I'll create a test pin. So, I'll open test panels. Since everything is powered using USB DAQ. I can provide analogue output of one volt. So, let me update. Then I'll go to a counter input. So, here we can see, pf0 start. So, one thing it is clear that, depends upon the RPM, depends upon the input volt is provided to the, driver circuit, we can understand the motor is rotating. But this is not clear to us to understand about how many numbers of pulses it is coming.

Video End Time (9:58)



So, what I'll be doing is that, I write a code and I'll be providing voltage from software. So, that depends upon the voltage we will see, in one minute how many number of pulses the motor is rotating.

Video Start time (10:18)

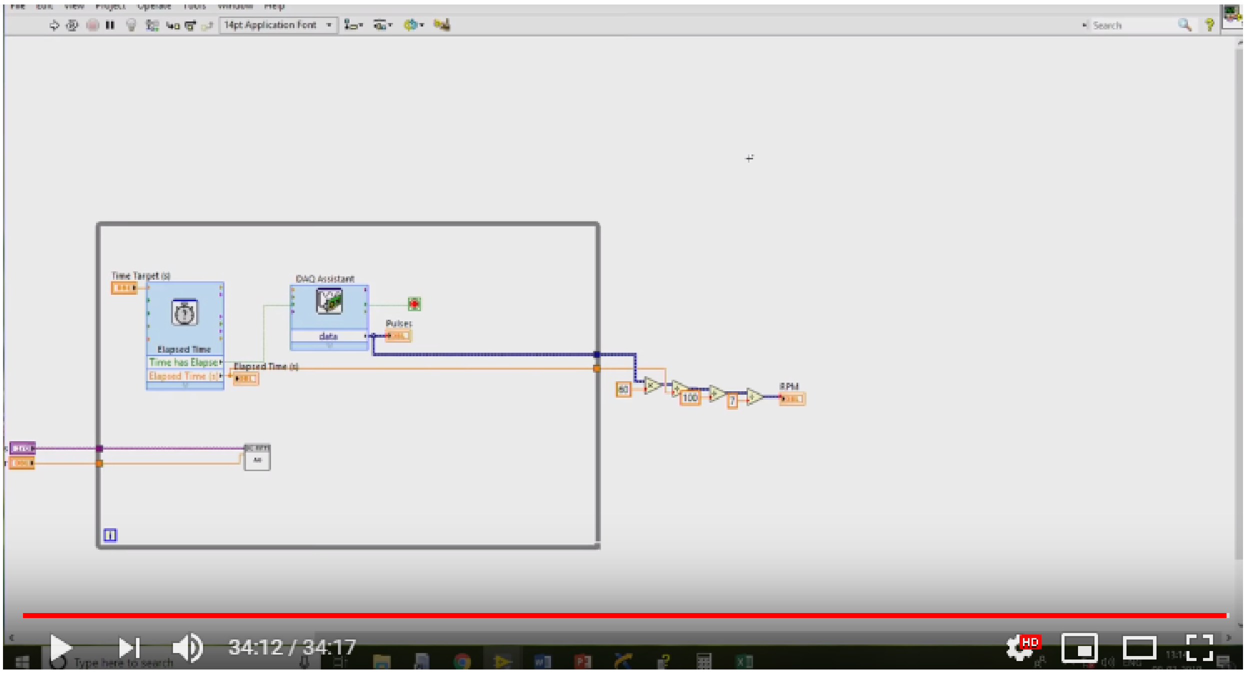


So, for that, let me open labue. So, since our intention was to, record. So, what I'll be doing is that, I'll be creating an express vi to acquire the counter signal. So, I'm going to input. Data system device, and I have to acquire counting input than, that should be edge count. So, since it is USB 6008.It can support one counter signals. It has a one part of counting signal. So, here we can a, we can see, whether we need following edge are rising edge. then, to understand whether it is working or not?

Here we can see, its keep on increasing. But until unless we stop and run the programme the counter cannot acquire cannot reset the previous measured value. So, what I'll be doing is that, am pressing okay. But idea was to understand what is a time it is taking or to understand about how many number of pulses it is going to generate per second. So, that means the system should automatically stop, when the input time is greater than or equal to one minute. So, because RPM rotations revolutions per minute. So, I'll remove this stop button. Instead of that I'll go with elapsed time. I'll set this elapsed time to one second. So, here we can see, we can also set a target time from front panel. So, what I'll be doing is that, I 'll go to the front panel I'll create a control. So, from this we can create, how what is our target time. So, this is in second. if I'm mention one, which means that, it has to run for a second. Ten, since it is an elapsed time timer. If you look into the help in do. It says that indicates the amount of time that, has elapsed. Since the specified start time. So, once you start the system. So, once it reaches to the time, then it automatically provides a signal saying that, if the time as an exceeded based upon the if the time exceeds based upon the time that we have set it. Then automatically it triggers the output as high in time has elapsed terminal. So, what I'll be doing is, that

this particular terminal I'll be connected it to this. So, what happens? So, when when this particular when the time has been elapsed, automatically this blocks provides a true signal to to the counter. Then automatically data acquisition system will stop, and it'll stop the complete loop, and in order to accurately visualise. So, we can also, set how much time it has been taken elapsed, before the stop of execution. So, What I'll do doing is, that I'll create an indicator of this point. and in order to see how many number of pulses it has been counted, so I will set control, sorry indicator, numeric indicator. So, it gives us the data of pulses which gives pulses, but this is not but this is not our actual RPM. So, if I want to find out an RPM, we have to do a calculation, so the thing is that these are the number of pulses that we are getting with respect to particular time, but RPM gives us how many rotations per minute. So, we have to measure with respect to 60 seconds, but what are the pulses that we are getting only 4, for this particular time allot set time. So, what I'll be doing is that I will take, I will take a multiplier, I'll multiply this with 60, so 60 indicates seconds, Right? So, so our idea was that, for x seconds for x seconds we are getting this minute pulses, whereas for 60 seconds how many pulses it will be equal into, so 60 into the number of pulses we got, we got divided by elapsed time. So, I will take a division operation, I'll go to numeric divide connect from here to here, so whatever we'll get will be RPM. So, I'll take another digital output block and I'll make it as RPM, I'll take another indicator. So, I'll make it as RPM, irrespective of how many number of seconds it is deducting, so it gives us RPM, by right now if you observe this can only acquire the signal, but if I want to even control the input voltage. So, what we can also say is that we can take another data acquisition device in order to control the input signal power to the driver. So, I'll take another express VI, I'll go to input or output I have to make it as an output, I am taking it as an output, data assist or for our simplicity we can use an simple measurement IO. So, what I'll be doing is that I will take, I'll create a new VI, I'll call that VI into that this VI is meant for acquiring the signal, so for that I'll take low level VI's, so I go to NI DAQ's in measurement IO, in this case I can create a channel, I'll create a channel but this channel is meant to provide analog output of voltage, but which to channel? So, we have to provide a control which indicates the specific channel, what are the voltage that we set has to be given to this particular channel, so this one I'll name it as the given a00, so a00 indicates, a0o, sorry! ao0 indicates analog output 0 channel, so let me zoom it, increase the font size, then what I'll be doing is that I'll take another low level VI, this is for whiting it, so this is for one channel one sample, I'll say data, so I'll take con, I'll take control, I'll take a numeric control and connect it here, this numeric control is input to the driver, input voltage to driver circuit, that means which means that for DC motor, this all be connecting is as input side, data and this dot should be connected here, error to error, error in error control, then error output. So, so in order to call this as a sub VI, in order to call, in order to work as a sub VI and to be call from another VI, we'll also have to set the terminals. So, what I'll be doing is that I'll take each connection, each pin and I'll be connecting it to this, here we can see, so depends up on the data type, it will automatically changes that particular pin to that default color, so now I can also have an option, I also have an option to edit the icon. So, what I'll be doing is that I'll go to a template, first let me delete this then go to a template our basic template, so in this I'll be writing Dc motor, DC control, DC-RPM control whereas this one is A out, so let me create a folder, DC speed control, so the name of this folder I am saying it as this file as Analog underscore out for DC motor. So, I'll be calling that VI here and the input is to the VI control, so let me keep it outside, so that at a time it can only pass the single data, so the physical channel I can keep it inside, but even outside is also, Fine! Whereas the data, the data has to be passed through this system should also be before the start of execution, so now, so let me set the physical channel to a00 and the input voltage to the driver, say 1 volt, so the time target is 1 second and here we can understand what is the elapsed time as well as how many number of pulses it has giving, everything we can see here, Okay? So, let me run , so initially this is channel where we are providing the analog output voltage and whereas the channel for data assistance we have already done that so now what will be doing is that I'll be running the DC motor with 1 volt and we will see what is an RPM and how many number of pulses it is doing it. So, When I run the program, Okay? Stop. So, the maximum value, requested value is 10, the maximum value is 5 and, Okay stop. So, when we looking to our USB DAQ, we know that USB DAQ can only provide an analog output voltage of 0 to 5 volts So, but whereas the default max value is of 10 volts. So, we have to replace the default with the 5 volts and the minimum value is of 0, create constant, so 0. So, now let me run, Okay? So, here we can see, so when the input voltage to the driver is of 1, the number of pulse will be that is 2, 1, 2 and the RPM of the motor is 12672, but with this we cannot understand, let me change to input voltage to 2, 3, 4 and 5, Yeah make sense, now so, we can see that repeatedly if you keep on running it, it is giving somewhere around 212, 213 pulses. So, which means that we can understand that 212 is a number of pulses it is going to provide when the input voltage given to the motor is a 5 volt, but it cannot gives us how many number of pulses, you know it is being generating it, for that we have to understand, we have to create a timer any way we already have a timer here, and for 1 minute, 1 second we have to see how many number of times the motor is rotating it. So, let us count how many number of you know revolutions it is making it, if you look into our we are saying it as an RPM, this is an RPM of an encoder but if you see that we have a total of again reduction ration of 100. So, what we have also to do is that the output of this has to be divided by 100. So, this case was RPM of a motor, irrespective of how many number of time that we are name, you know measuring it, the RPM of a motor is 117.39, so let me run once again, so this is the actual RPM, but but we do not know whether this is an RPM or not, the reason is that we have to divide the RPM with how many number of pulses that the encoder is rotating, so in order to understand we should know in in a for 10 seconds how many number of pulses the motor is rotating. At the same time, we also count whether it has been divided by 10 seconds or not. So, let me do it, so what I'll be doing is that I'll be counting manually, so will start 1, 2, 3. So, it is take, it is taking only three complete revolutions, so that means for 10 seconds it is of three complete revolutions, so that means for 60 seconds, it is approximately of 18 RPM, so we can here, we can see that here we can see, for 10 seconds we have manually counted three number of full revolutions, which means that for 1, for 60 seconds which is 1 minute, it is of 6 in a 3 into 60 divided 10, so which is nothing but 18 pulses, so but we are getting somewhere around 117.5, so that means 18 pulses corresponds to 117.5 RPM. So, If I divide 117.5 divided by 18, 117, 117.5 divided by 18, again divided by 18, so we have to use 7 as its not a 14, the 14 is the complete number of clock pulses we get from both c1 and c2, whereas 7 is from this single each channel. So, so which means that in order to actually find it the RPM of our DC motor, the output of the output this point ahs to be again divided with 7, so which means that for one revolution, one complete revolution of a DC motor, we are getting 7 pulses from the encoder, so I'll divide this with 7 to get in actual RPM, so since this calculation everything should be you know, it is easy to keep it outside, so what I'll be doing is it, I am keeping it everything outside, so let me run once again for 10 seconds, Right? So, 16 point. So, one one thing is clear that so keep it 1 second let me run, so here we can see 16.2. So, that means for 1 voltage the RPM of a motor is 16.6, so for let me change input voltage 2, so here we can see the RPM is 42, for 3, 69 somewhere around 70, for 4, 97, so for 4 it is 96.69, 97, if I change it to 5,124, 125. So, again for 1 v got, for 1 volt, for 1 volt, we got somewhere around 16 RPM, let me change to 2 volt, we got 43, let me rewind second time?, so 43, 44, whereas for 3, so we are getting somewhere around 72, whereas for 4 volt, 99 and for 5 volt its somewhere around 125, 126. Now, if I plot a graph between these two, yes! this is completely a its linear and we can see we can also plot a tot line and we can see it's a value to, so that means the slope of the line is 27.3 and the offset is 10.7. So, that is clear which is also equal to what we have done in our the previous consideration, but only thing is that there we have consider 14, but actually speaking it is only 7 pulses per revolution, Okay? So, now we understood how to do the calculation everything, but what we have to do, our intention was to design the controller in labue, interface the motor the using data acquisition, which we have already done, but we have to understand with respect to different controllers and with respect to the gain parameters of your controllers, how exactly the motor is going to control it

Video End Time: (34:16)


14pt Application Font
Time Target (s)
Elapsed Time
Time has Elapsed
Elapsed Time (s)
DAQ Assistant
data
Pulses
60
100
RPM
34:12 / 34:17
Type here to search