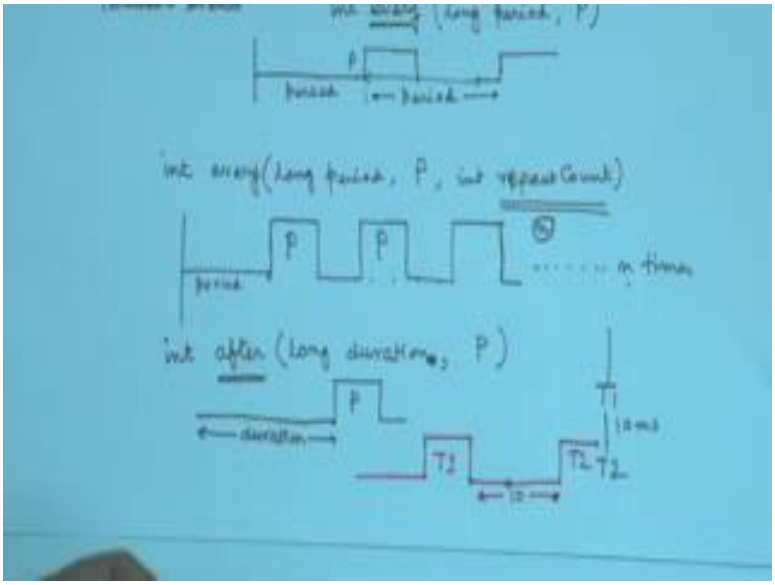
Embedded Systems Design
Prof. Anupam Basu
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 18
Controller Design using Arduino

In the last class we have discussed about interrupts, how Arduino handles the interrupts and also we started discussing on timers. Timer plays a very important role in any embedded system design, now there are different applications of the timers.

(Refer Slide Time: 00:47)



For example a timer can be used to create periodic events, so for that we have seen that in Arduino we have got this keyword every; we have seen long period say it is a P 1; that means, every starting from a time, every period there will be an event P and that event can take some time.

But again after the same period, so from here after period duration, we will have again that event taking place. This is periodic that is a occurrence if this period is from here to here right, so this is a same period; this part and the this part. So, that there by we will see when we will look at real time operating systems, the periodic processes follow this sort of nature and that will happen at every occurrence of the period. So, here internally the timer is keeping a note of the period and after every completion of that period; it is generating that event P whatever.

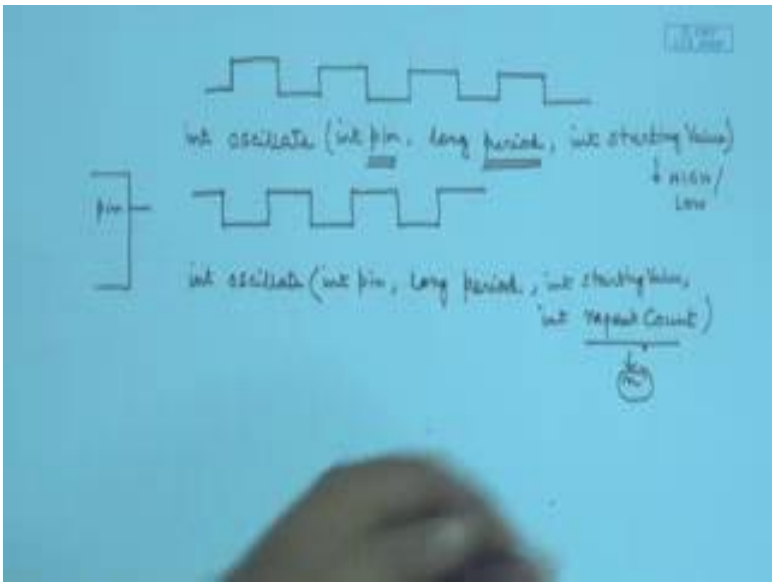
The other form can be int every again long period say P and int repeat count; now this means that here it is going on forever, but here it will go on for n number of times. So, if this be my time scale and this is period say 4; this event takes place and 1, 2, 3, 4; again this event takes place 1, 2, 3, 4; after this event takes place and this on this thing goes on n times; this is another application.

So, if you are asked to generate or actually implement this function, you should think over how you can yourself implement this function by a small piece of hardware maybe. What will you do? You will have some timer and also a counter, we will have a count and a timer and the timer is taking care of this period P and as this period elapses, there will maybe a pulse generated and this event; this P both of them I will make P period; let us call it P r d; this period after every period that is been counted by the timer a pulse generates and then this function is called and the counter is counting or counting down counter is actually counting up, so it will count up to n and whenever it is end; it will stop.

So here, these have been given as a function, but ultimately when you compile; it will have to be implemented, so each of these can be synthesized. So, int after duration P; that means, it is after a particular duration this P will be called; that means, this is essentially like a delay, I can create a delay for this event after so much time then after it can be say I have done some task here say T 1, now T 1 has completed and after completion of T 1 say after 10 millisecond of completion of T 1; I want T 2 to start.

I can implement it in this way that after T 1, I put this int after the duration 10 millisecond P 2. Therefore, T 1 started here and T 1 was over here; T 1 was completed after that I put duration 10 then T 2 starts. So, this is the delay function sort of I am sorry this is not visible, so I put this delay in this form and after that it takes place.

(Refer Slide Time: 08:12)



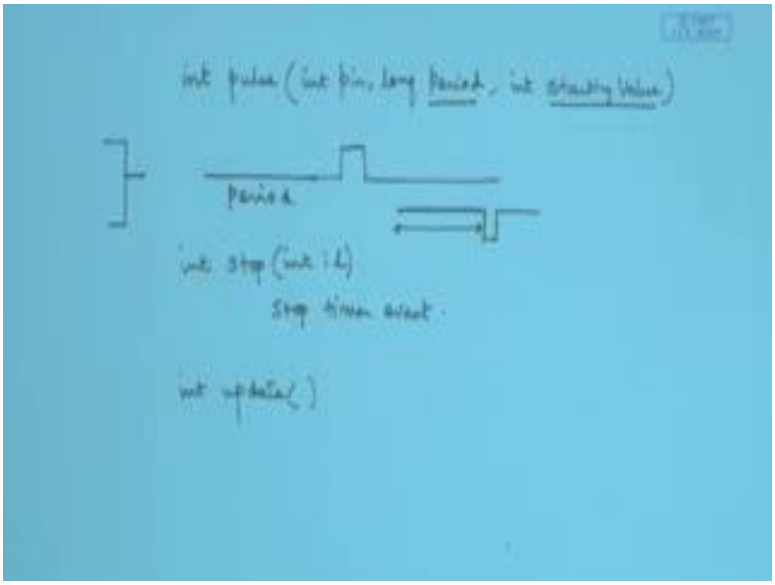
Next the other functions that we often need is to create an oscillation; that means, I need something like this, some pulse train to be generated; train of pulses. For that we have got say int oscillate; int pin starting value. Now one thing I would like to mention that is a little out of context of this timer that these functions I am showing with respect to Arduino, but it maybe that from 2 years Arduino is not available; something other, some other pulse is it is very popular.

So, maybe even today raspberry pile very popular, so for any one of those this exactly this function may not apply, but there will be a similar function which we will have to look up in the manual and the principle is essentially the same. So, we need to generate this pulse; that means, I am toggling the state of a particular pin; a particular pin is been toggled. So, there is a pin coming out of this pulses, some pin and I am toggling that every period milliseconds; every period milliseconds and the starting value it can be started from low to high initially it was low to high toggle.

So, if this can be either high or low, so the pulse train could also be like this the starting level was high and I create a pulse train like this that is also oscillating. So, now you can also now you have seen like in Arduino, we have got the pulse width modulator. Here of course, we are fixing the width of the pulses by the period, but by changing the period you can play with the frequency of the pulses. Another variety of this is int oscillate; the pin, the period, the starting value value and in addition there is another repeat count.

Again we have encountered this earlier; that means, it says how many times this pulse will go on, how long will this oscillation go on. This is as I specified as long as I explicitly do not stop it; it will go on, but here I am specifying some time and other things remain the same, so that is how we can generate the oscillations. In the last class we have also seen an application how we can write an interrupt code.

(Refer Slide Time: 12:31)



Another important thing is pulse; say if I use this function what will happen with this? Pulse; that means, I will just generate it once. So, with respect to a particular pin; I have got suppose it is starting value was low and after period I set the pulse sign. So, this is just a generating a pulse, similarly we have got int stop.

Student: All of the pulses run automatically at decimal body to the (Refer Time: 13:37) our starting value is low.

If I starting value is low then the pulse will go high, otherwise if my starting value us high I will have a negative figure like this.

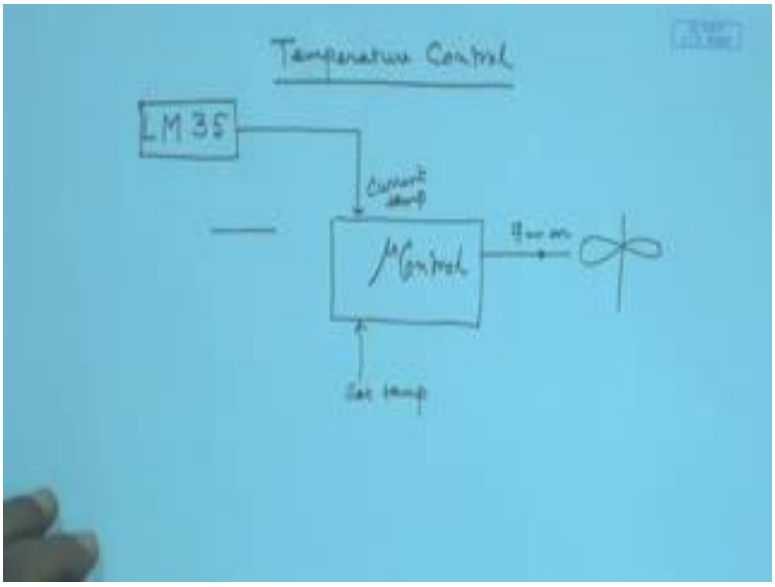
Student: (Refer Time: 13:49).

Yes s this starting value is what is a value now and from here I am giving the period and getting the pulse and stop is it is an ID of the timer. So, I can stop a timer say for example, in this earlier case we have seen that this thing was oscillating and every

function since now you can see that there is an int here, this function is therefore written in some integer and that is an ID of the timer.

Now since I have got the idea of the timer, I can; here I am explicitly specifying how long it should go on here I have not. So, therefore, I can explicitly make it stop using this sort of stop type of command and there is another one int update; this should be this updating the timer alright. So, timer we will come back to the necessity of the timer little later.

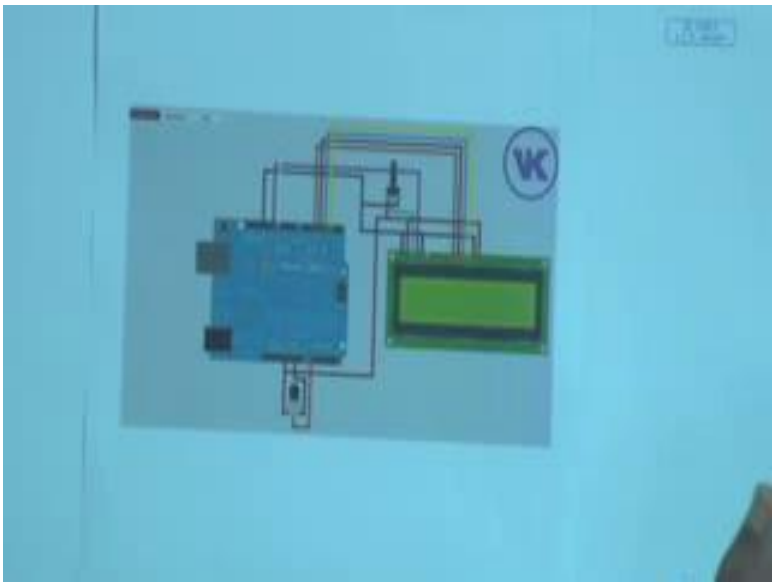
(Refer Slide Time: 15:35)



Now, we will move to an application of Arduino in the form of a controller design. So, we will start with a simplest controller design; say we want to carry out some temperature control which you can do very well yourself. You need first of all; a sensor, a temperature sensor and you should select the temperature sensor depending on your requirement and what are the requirements, I asked you to study what are the parameters that are required for a sensor and I also suggested some page.

So, I hope you have gone through that; now selection of the temperature sensor will depend on the range that I want to actually sense, the linearity of it, the resolution and many other factors. So, one very popular sensor for temperature is LM 35; comes as a small chip.

(Refer Slide Time: 16:47)



Suppose we select; now what do we want is; we want something like this say we have got an Arduino board and in Arduino board here we have got two pins; one is the 5 volt pin the supply, another is a ground. Now these two grounds and 5 volt are connected to this; this where I am holding the pin, this one is a LM 35 alright. So, this will sense, so there are two ends; there are 3 pins one is a 5 volt, one is a ground and there is a temperature value that is coming up from here and I am taking it to analog in; what will it give me at this point?

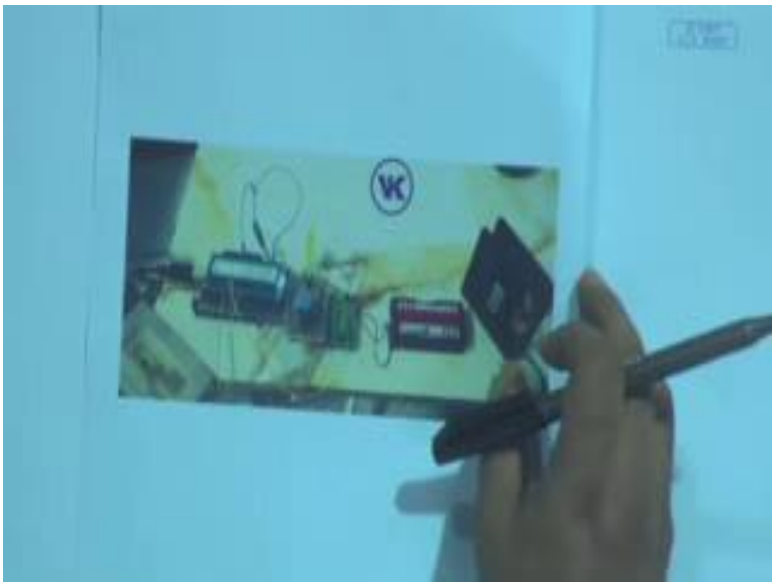
Student: Voltage value.

It will give me some voltage value, it will not give me the temperature; it is a sensor that is sensing the temperature and returning an equivalent voltage value which is analog. So, that is coming at this analog point, now I can do on the other side, I can have what should I say.

Student: LCD (Refer Time: 18:10).

Display LCD display and all those this can be connected, but actually we want to, so this would suffice. I connect the LCD display, it would suffice if I had just rate the temperature and display it, but since I want to control it; I also need some actuator and the based actuator will be a fact say simplest, not the best simplest actuator will be a fact. So, as soon as we have temperature going beyond the level, we will put the fan on.

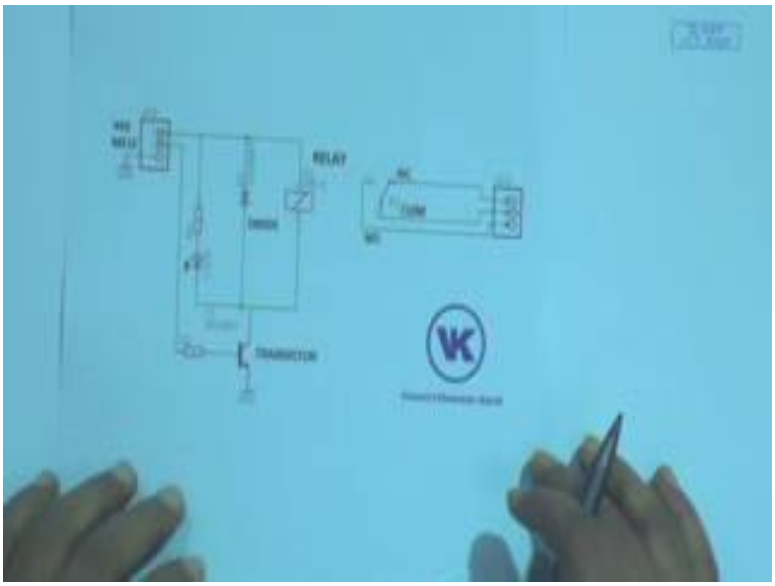
(Refer Slide Time: 18:53)



So, that is our objective and the simple setup for that would be something like this that I hope it is visible; there is a fan here and which is given by battery and here is my board with Arduino board with LM 35 and everything. So, whenever the temperature goes beyond the level; I will set up the fan that is my objective. So, simply speaking so here I have got a sensor say LM 35 that comes to the Arduino board, my micro controller and I have already told it some set temperature range; if it exceeds that if the current temperature.

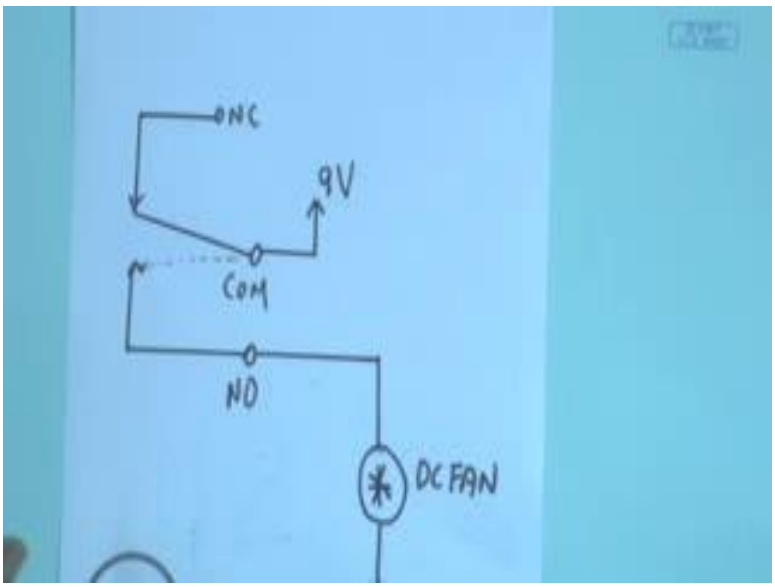
So LM 35 is sensing the current temperature, if that exceeds the set temperature then I actuate say fan on. So, there will be a fan here and that fan will be on that is what I want to achieve simple.

(Refer Slide Time: 20:29)



Now, the fan a quick connectivity of the fan will be something like this; how would I put the fan on, the best way to put the fan on is to have a relay. What is a relay? A relay can be normally open or normally closed NO or NC; a normally open relay on being switched will close; a normally closed relay on being switched will open. So, here we can see that when I want to drive the fan; I want to make the relay closed.

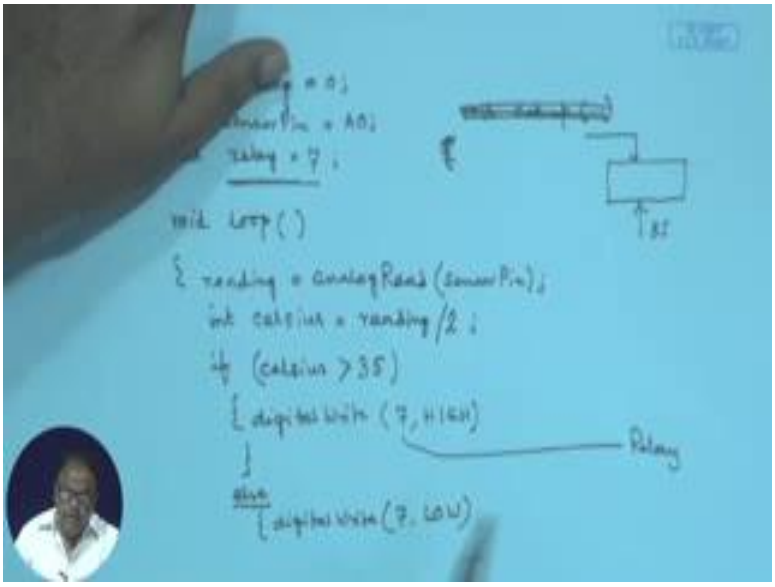
(Refer Slide Time: 21:35)



So, typically so here what we show is a there is some LED's etcetera and ultimately the relay is connected here to show you in a better form the schematic would be this that see

I have got the DC fan here and the DC fan is connected to a relay which is; this is normally open and it is normally closed. Now here is 9 volt; here is it is not, it is closed over here; now when I put it on then this one will be connected. Therefore, the normally open relay will close, so this will be connected on setting something from somewhere I put that relay on, so this fan is starts' rotating that is my scheme. Now this scheme I will have to implement using my program and so since we know; I will just write the relevant part of the program.

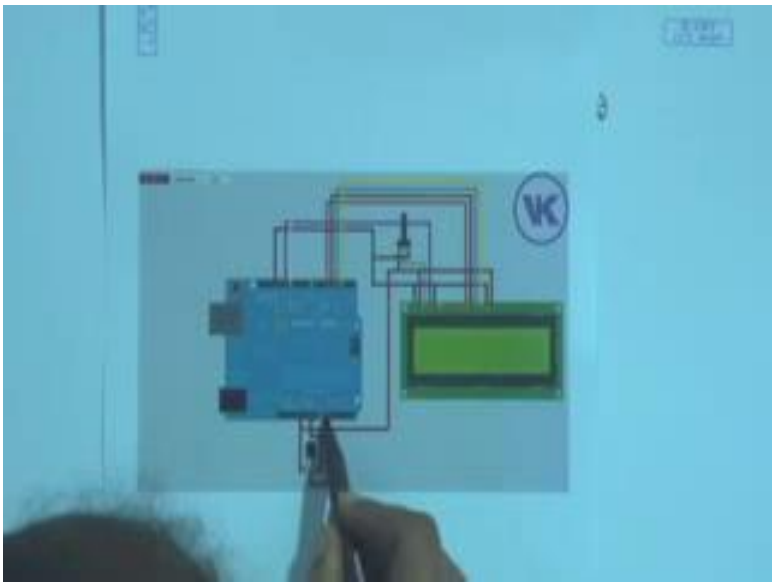
(Refer Slide Time: 22:40)



The whole thing I am not writing, but say I am just giving some initializations int will explain it in a second. So, I am setting the relay the variable name to a value 7; now let us come here. So, first thing that I will do, I can set up the LCD but here I am skipping the LCD, so I need not print any message here, let me just quickly relay does some come here. Now most of these control things will be in an infinite loop, so it will continuously sense on to that. So, there is a loop going on I am not setting the cursor and so first I am taking the reading through the analog read, this function we are already familiar with analog read with the sensor pin.

Now, where did I connect that sensor to the analog input that I already shown you that we are connected into the analog input is 0.

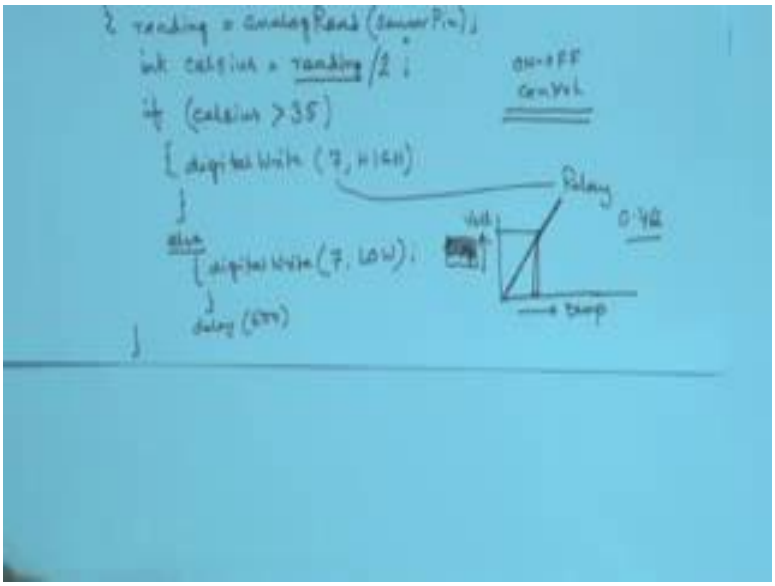
(Refer Slide Time: 24:47)



The sensor was connected LM 35 to the analog 0; this pin, so I am adhering to that. So, I am taking analog read and here now I can use my variable name; sensor pin. Now this is important just see int, Celsius is reading by two; twice, this twice this term. If I will come to this, if Celsius is greater than 35 say I have set it up with the set temperature with 35 and from here I am sensing it, if this is greater than 35 then digital write 7; high, 7 is the pin number where I am sending the high; that means, I am closing the relay and the fan will start to rotate and this will go on, but it is going in a loop, else digital write 7 low.

So, where is this seven connected; seven is connected to the relay you see here, so digital write 7 low and then I give a delay.

(Refer Slide Time: 27:15)



Then I give a even delay of 500 millisecond and close complete, so that is the loop. Now this delay is temperature is a sluggish process, so 500 millisecond means half a second it is good enough, but for faster process is you wanted it quickly. So, I think is clear what I am doing here, I am reading the sensor pin which has been connected to analog a 0 then there is a puzzle here why I am doing this I will come to that.

If this Celsius value is greater than 35 then digital write then I write the high to the fan, the fan goes on after 500 millisecond I again check it, if there is a still more than 35; I again keep the fan on, otherwise I will put the fan off and again I will go on checking. So, what is this happening here it is an on off control this sort of control is known as a on off control. Now why is this? The reading that I am getting; what is that reading that I am getting?

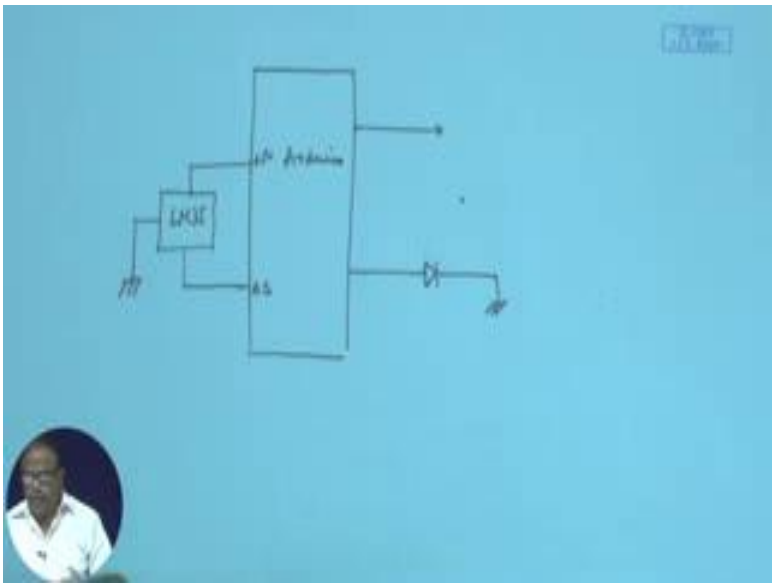
Student: Voltage value.

The reading that I am getting is a voltage value, now the sensor itself has got a mapping of the temperature sorry the temperature actual temperature or let me say there is a characteristic, but if the temperature is something then it generates the voltage value; for a particular temperature there is some voltage. So, depending on this slope I am assuming it is very linear it is not always that linear for the all the range, but I am assume for the range that I am considering is linear; the slope matters. So, given this actual

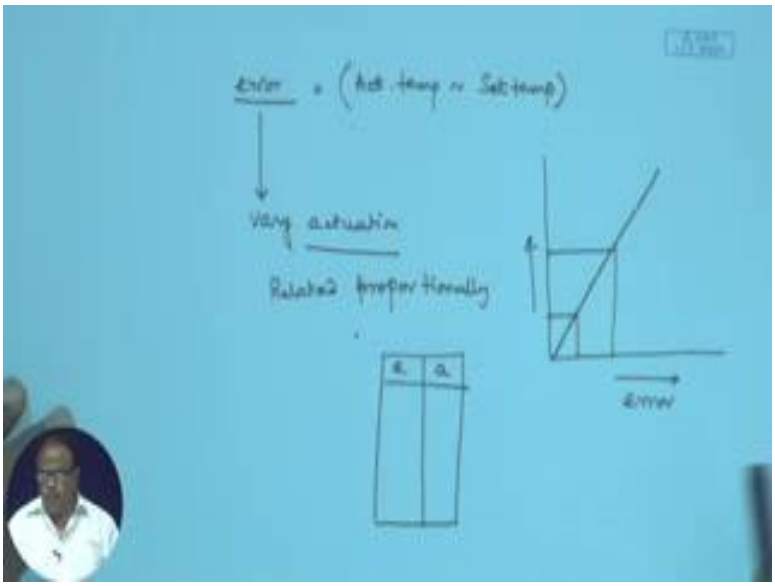
voltage, I have to come to this temperature and actually for LM 35; it is around something like 4666 etcetera.

So, I am dividing it by 2 approximating it 3.5, so depending on this is a scale factor. So, depending on the sensor that we use, we will have different scale factors and that is why this divided by 2. Now here it is a very simplistic control I would say, this is I am just doing on off and I am not doing any sort of proportional control or anything, how much say if it be 36 then also I take one of; if it be 37 then also, if it be 45 then also I will do the same thing, but that may not be a always desirable.

(Refer Slide Time: 30:51)



(Refer Slide Time: 31:45)



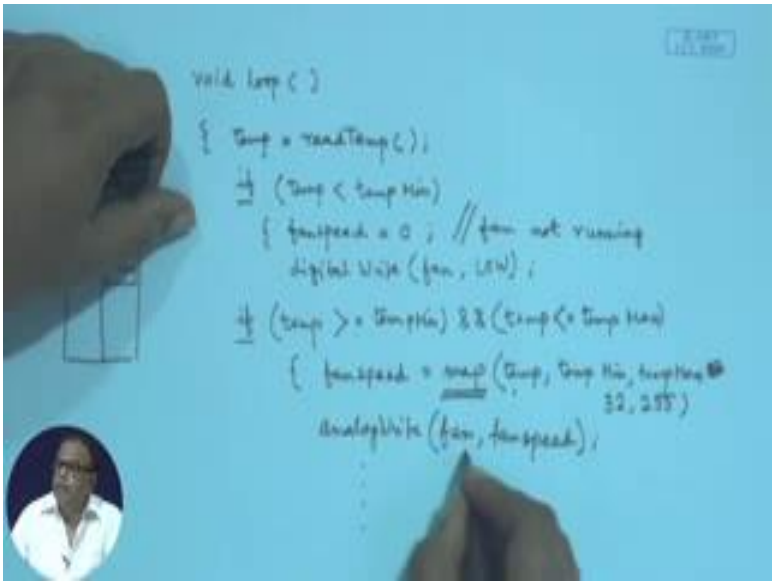
So, if I had the something to do like, so I have got the Arduino board and I have got the LM 35 connected to that some sensor, this 5 volt then there is a ground and I am putting it to some analog input say A 1, I connected it to A 1 and then I am driving the fan from this point and also I am writing the LED that if I can modify the program very well that if the temperature goes beyond the particular level; I will put this LED on. So, that we can easily extend the rewrite the program to extend this no issue, now if I actually want to do some mapping, so the other thing is that depending on the error.

So, I have got what is the error; the actual temperature difference, the set temperature there is a actual error; there is a error that I am getting. Now; if I depending on the error how much is the error; depending on the error I put the delay or say it maybe the delay or it may be the speed of the fan whatever; I vary the actuation, this actuation and the error this can be related proportionally, in that case we call it to be a proportional controller. So, we have got something like this I have got the error here and the actuation signal and here and that is somehow proportional. If the error is less, my actuation will be less, if the error is more the actuation will be more.

I am not saying that this is a best mode of control, but this is a very useful mode of control. We will see later that there can be better forms, but if I want to implement this sort of proportion and for the sake of argument; let us assume that this is absolutely linear, in that case I can very well; I can compute an equation if I know the loop of this

or I can very well keep a table of the error versus actuation, I can keep a table. Now if I do that, how would I proceed to modify the earlier control algorithm? The earlier control algorithm was purely on off, but what I want to show here is even with a very simple micro controller like Arduino, I can establish such proportional and other sophisticated controls as well.

(Refer Slide Time: 34:23)



For example; see again I am just starting with this loop, so I read the temperature by some function here I am using a function read temp fine; I am getting the temperature. Now here if the temperature is less than; now here I am setting two things one is the max temperature, other is the min temperature. If temperature is less than the min temperature then fan speed is 0; that means, a fan is not speed; fan not running, it is less than the minimum and then I digital write fan low.

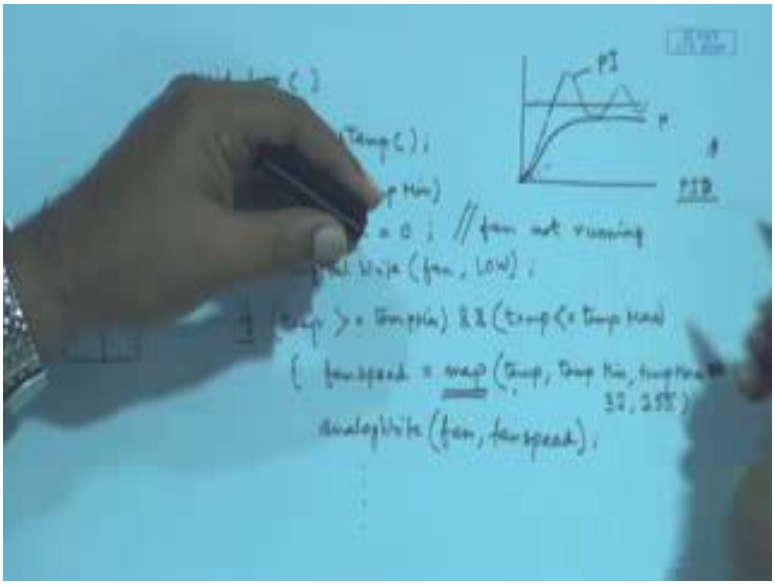
Now, if temperature is greater than equal to temperature min and temperature is less than or equal to temperature max; that means, it has not exceeded the alarm condition, but it is in between these two, then I can say fan speed; now this is what I was talking about is map. Map is a function that is taking the temperature; temp min temp max I am sorry this continuing 32 comma 255. Now; that means here this map function is taking this actual temp and for temperature min my fan speed as been set to some value in Arduino and for max it has been put some value.

So in between this range, it will return your fan speed, so that is what I was telling that there is a table from which I can find out that what is the temperature and accordingly I find out what is the fan speed. So, and then I can do analog write; if you like you can write the LCD also if you like; fan, fan speed. Now one thing, so in that way; so this is just an a example it will go on it will tell you how I can also achieve the proportional control using this sort of table. Now here there is one thing that I have not mentioned that is a fan, it is actually the fan speed is a value and that is going to the fan pin, this is a pin ID of the fan, where the fan is connected.

So, that should be given in the before I mean the setup, so I hope this is clear. So, accordingly see now there are other modes of control which we will discuss later like proportional control, sometimes leads to some problems like we; what will happen is that you want to have the temperature to say; the particular value and as you try to achieve proportional control, you never reach that value; there is always an error in between.

So, in order to do that that is known as the steady state error, but I am trying to achieve this value, but I am never reaching that value there is always small error. Now if I want to compensate for that error, sometimes it happens that it over shoots the level. We try to compensate it by adding an integrator, we will see that later and by that we will do an over suit that we also do not want.

(Refer Slide Time: 39:54)



Therefore you want to damp that over shoot, it will look something like this that here is my desire state and by proportional control I gradually arrive at some point like this and this error is there, but whenever I want to compensate this; this is proportional. If I want to do sorry I mean along with that if I want to do it integration then it rises first, but it over shoots; it goes on like this.

Now that is how again I do not want, so this is known as the PI control and in order to do that a damp it, we again add a differentiator which leads to the PID controller which is very sophisticated PID, Proportion Integrated and Differentiate. We will see that, but here is a very simple example with Arduino, now we can achieve the proportional control.