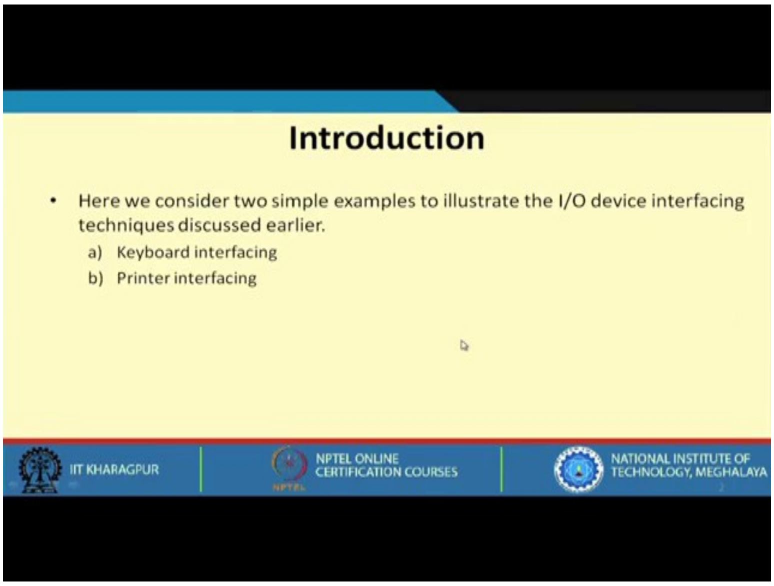
Computer Architecture and Organization
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 49
Some Example Device Interfacing

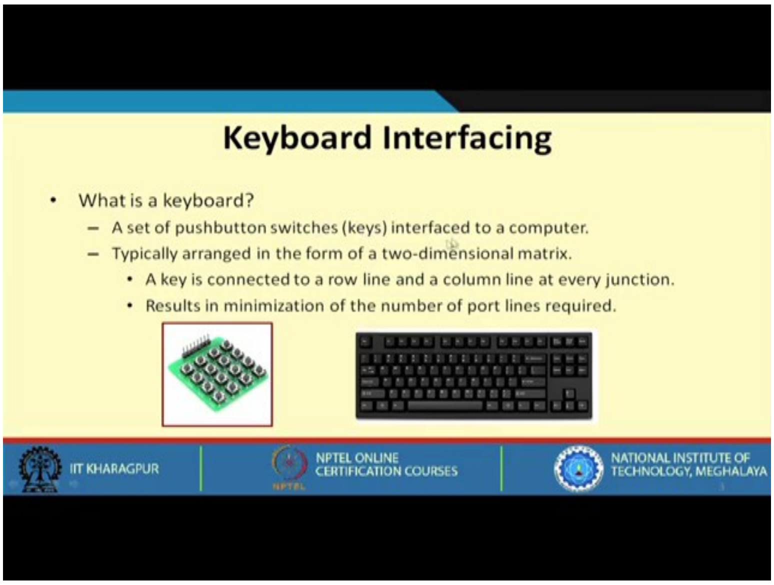
In this lecture we shall be looking at two examples for device interfacing. Deliberately we have taken some examples that are simple enough so that you can actually understand what is happening inside. And some of the IO transfer techniques that we have discussed earlier you can see how we can use them here.

(Refer Slide Time: 00:44)



One is a problem of interfacing a keyboard, and the second one is the problem of interfacing a printer. Let us look into this one by one.

(Refer Slide Time: 01:03)

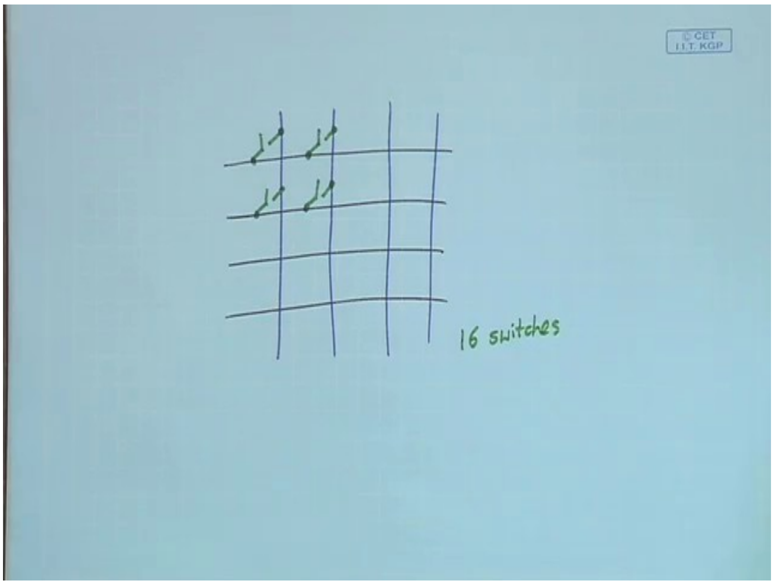


All of us use keyboards in our computer systems, in our mobile phones, wherever we have a mechanism for entering something. Even in our microwave oven we have a version of the keyboard, even with our TV remote there is a keyboard. With our air conditioning machine remote control there is a keyboard. Any device that allows user to give some input via pressing of switches is a kind of keyboard.

A keyboard in generic sense is nothing but a set of push-button switches, which are called keys. This is an example of a small keyboard where you can see that there is 16 keys, and this is a standard computer keyboard, which is also conceptually very same.

This example as it shows that the 16 keys we have organised in 4 rows and 4 columns. Now logically speaking we arrange the keys in this fashion only. Why? See there are rows and columns we connect a key at the junction of the rows and column.

(Refer Slide Time: 02:43)

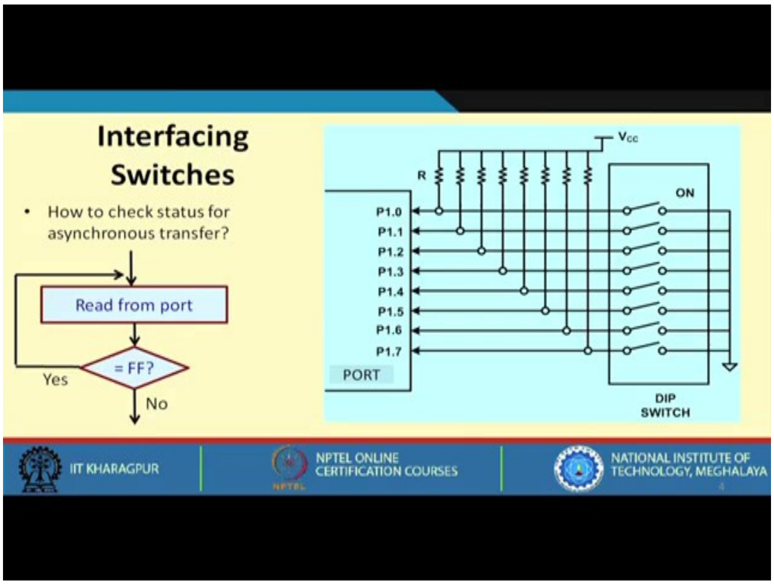


Suppose I have some row lines and also there are some column line, but they are not connecting they are on a different layer.

Now, what we do? We connect a switch between every row and column. There is one switch connected here, there is one switch connected here, one switch connected here, like this. In every junction there will be a switch. For a 4 bxy 4 array, there will be 16 such switches. Whenever you press a switch the corresponding row and column will be connected, this is the electrical property of the keyboard.

The computer keyboard that you see may look like a little longer, but inside if you see the layout the keys are again laid out logically in a matrix form, because there is an advantage of laying out in matrix form. As we will see later, the number of interconnection lines gets reduced.

(Refer Slide Time: 04:07)



Let us come to the problem of interfacing switches. Let us start with the simplest scenario where we do not have matrix kind of a layout. We have a set of 8 switches.

Let us see how we have connected them. There are 8 switches which are shown here. One end of the switches is connected to ground, and the other end is connected to the lines of an input port. This is an input port of the computer and the switch can be regarded as an input device. So, whenever you are pressing a switch the state of the switch will be coming as an input data.

Now, you see whenever you press a switch because it is connected to ground that line will definitely become 0, it will be connected to ground. But if you do not press it is open then this line will be floating. So, you really do not know what voltage will be coming here. To be sure that you are having some voltage here you connect some resistances on all the lines, this is these are called pull up resistances. And they are connected to a positive voltage, which can indicate logic 1.

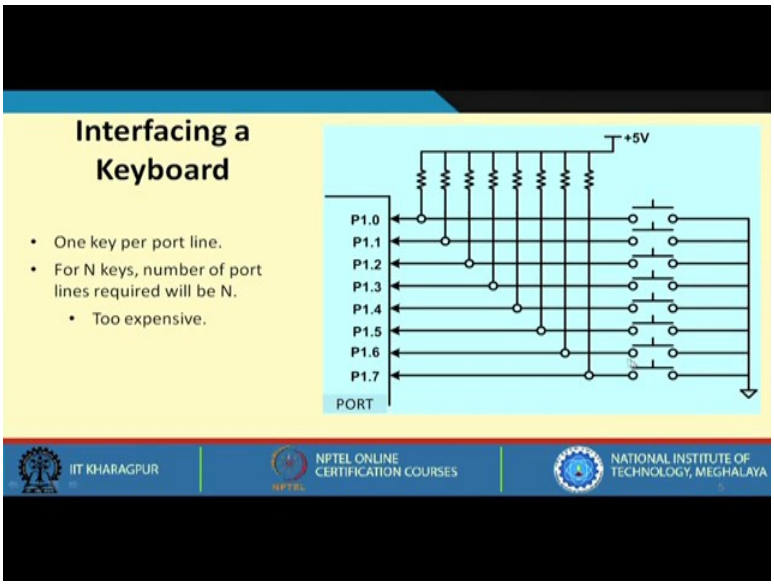
Now what will happen if the keys are not pressed, then this Vcc will come and appear here, this will be treated as a 1. So, if I have not pressed any keys and if I read this input port all these lines will be 1's, but if any of the key is pressed the corresponding input will become 0.

Let us see how we can check the status for asynchronous transfer. You recall for asynchronous data transfer or handshaking the CPU will be continuously checking the status of the device. If it is not, it will go back and check again. When the device is ready, then only it will come out and read the actual data.

Here checking for this status is very easy you see. If you have not pressed any key, if you read from the port you will get FF, all 1's. If it is FF you know that no key has been pressed, you go back and continue checking. But as soon as you see that it is not FF, this means one of the switches is 0, then you come out. The first thing you will have to check which of the bits is 0, that you can easily check. There are shift instructions, there are bitwise AND/OR operations. You can check the bits which bit is 0.

So, if you connect switches like this it is very simple. But there is one problem. Here you see we have 8 switches, with one 8 bit port you can connect them directly. Now I tell you that well I have 100 switches. Where do you connect them? You will be needing 100 lines, large number of ports, not only that, 100 connections. So, as the number of switches increases the complexity of connections and the number of ports will also increase, this is one drawback.

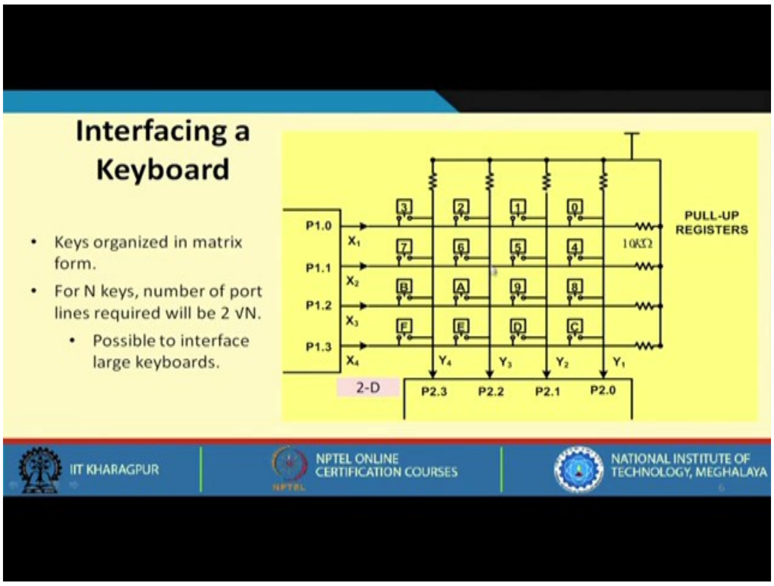
(Refer Slide Time: 08:14)



So, here what I am saying is that if there are N number of keys, number of port lines required will be N. So, if there are 8 bits per port then I will needing N/8 ports. And I will be needing one line for every switch; there will be N number of lines also.

This is too expensive to use and to layout, that is why people do not use although this is very, very simple. The way you check whether a key is pressed or not is very easy, but because of the complexity of wiring people do not use this.

(Refer Slide Time: 09:16)

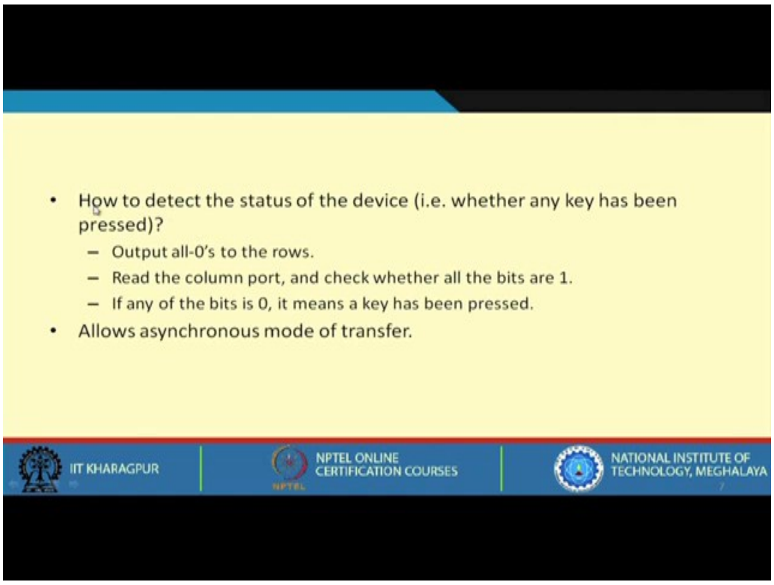


So, what we use? We use a matrix keyboard, just the kind of keyboard I just now mentioned. Here I have shown an example of a 4 x 4 matrix. There are 4 row lines, there are 4 column lines, there are switches connected between a row and a column.

Once you connect these 16 switches in a 4 x 4 matrix, you do not require 16 wires. You need 4 wires in the rows, and 4 wires in the columns. And as you can see that there are resistances to a positive voltage, the pull-up resistances are connected to the rows as well as columns.

Let us see how it works. The first thing is that the number of wires is getting reduced. So, for N number of keys if you can lay them out in a perfect square; if N is a square let us say 64, then you can layout as 8 x 8. So, we will be needing 8 wires on this side and 8 wires on this side. So, twice square root of N. So, even for 64 keys we need only 16 port lines, not more than that.

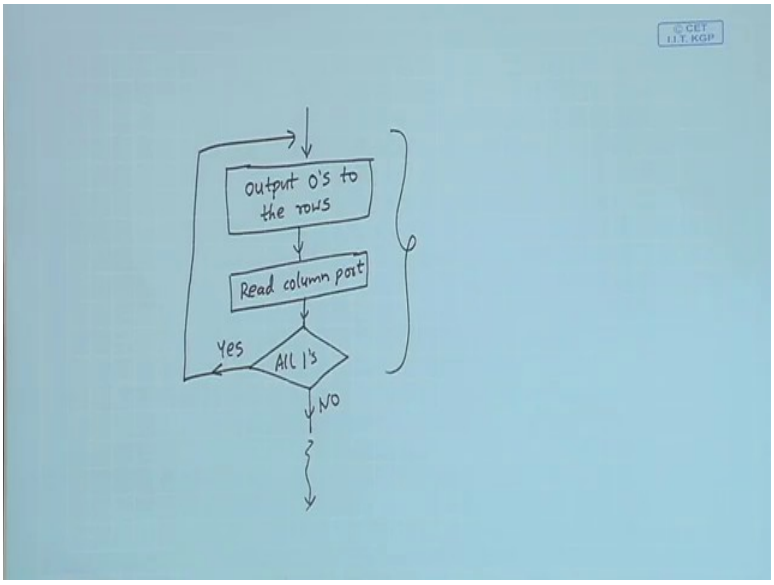
(Refer Slide Time: 11:21)



Let see how we can check whether the device is ready to transfer; that means, some key is pressed or not. How we can check that? How to detect the status of the device? Well, there is a simple way; you output all 0's to the rows then read the column port, and check whether all the bits are 1. If any of the bits is 0, it means a key has been pressed.

Look at this diagram again. You output 0 0 0 0 on this port line. So, these lines are 0 0 0 0. Now suppose none of the keys are pressed. So, the column lines they will all be connected to the voltage, they will get 1, 1, 1, 1. Now suppose this key A is pressed. So, this row and this column will get connected. This was 0 this 0 will propagate and this bit will be 0. So, if I read and if I see that this is not 1 1 1 1, which means at least one of the key has been pressed.

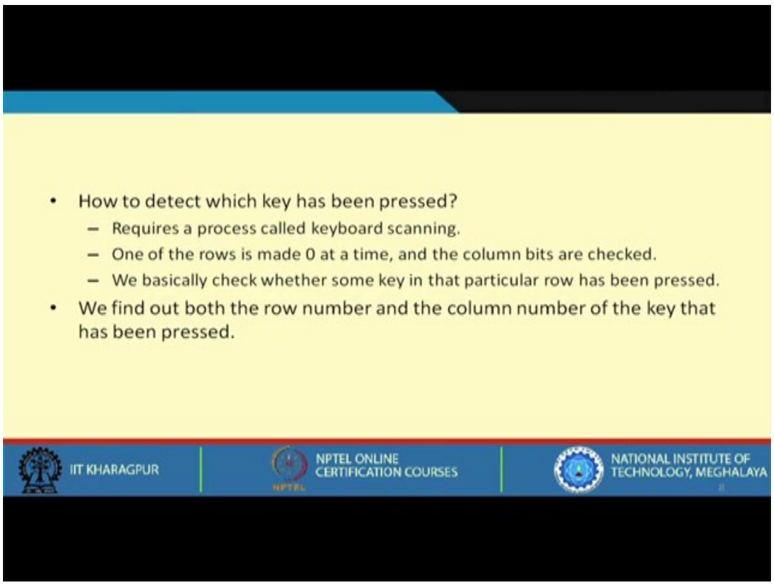
(Refer Slide Time: 13:01)



I will show in the flow chart of the code. We output 0’s to the rows. Read column port then you check here whatever you are reading are they all 1’s? If you see it is all 1’s, it means none of the keys are pressed. In that case we have to go back and check again. And if any of the bits are 0, some key iss pressed you can go ahead to the next step.

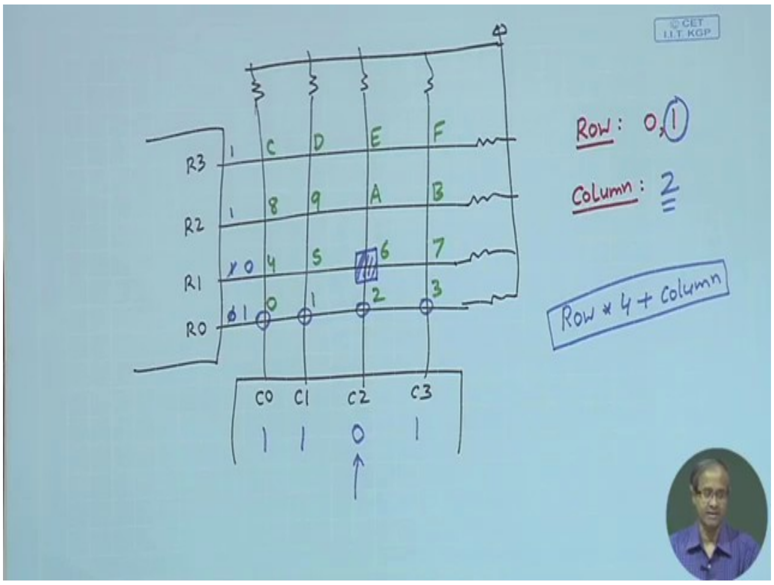
This is how you can check the status of the device.

(Refer Slide Time: 14:14)



The next question is, how to determine exactly which key has been pressed? For this purpose we need to carry out something which is called keyboard scanning. Here one of the rows is made 0 at a time, and the column bits checked. I shall illustrate with an example.

(Refer Slide Time: 14:47)



Suppose I have 4 rows and 4 columns. There is one port connected on this side, one port connected on this side. And there are pull-up resistances, they are all connected to the positive supply voltage.

I want to check whether a particular key is pressed or not. Let us follow the convention this is my row number 0, row 1, row 2, and row 3. And this is my column 0, column 1, column 2, and column 3.

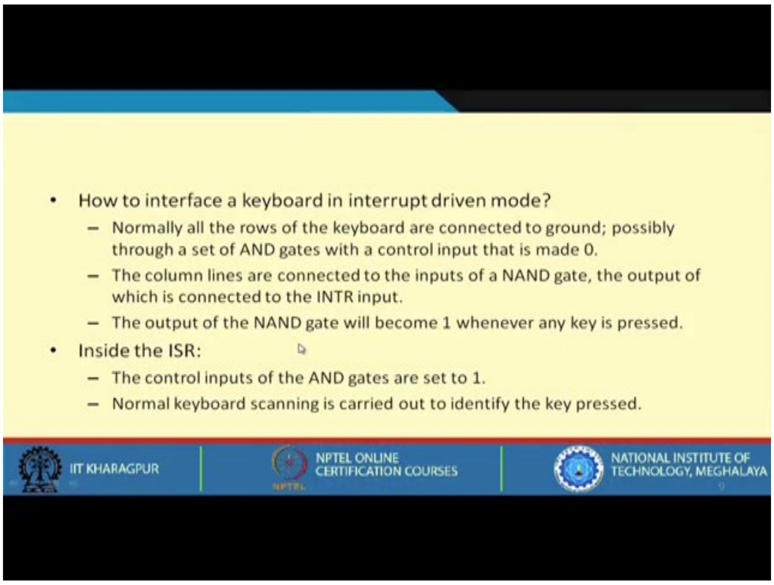
What I do is as follows. I keep 2 counters one counter I call it as the row counter, this is just a variable or a register. And another which is called the column counter. Initially I set the row counter to 0. I activate row, meaning I output 0 on this row, but all others are 1. Then I read the columns and see that whether any of the bit is 0. If I see any of the bit is 0 it will mean that one of the keys which are connected to the row number 0 must have been pressed, because of that this 0 is coming here.

But if I see it is all 1’s, this means in row 0 no key has been pressed, then what we do? Then we increment row to 1, and activate the next row. N, we make this as 0 and make the others as 1. You do the same thing, you check whether the columns any of these wants 0 or not. In this way suppose I had pressed this key, this key was pressed. So, in row 1 it will get detected. I am reading something which will be 1 1 0 1; I get this data, because this was being shorted 0 will be coming here.

Now that row number I have identified, there has been a key pressed in row number 1. Now in whatever I have received I will check the bit which bits 0. I check bit number 0, 1, and 2; bit number 2 is 0. So, I set column counter to 2. So, I have got row number and column number. I can define a key code like this row x 4 + column. This is typically done in a 4 x 4 keyboard.

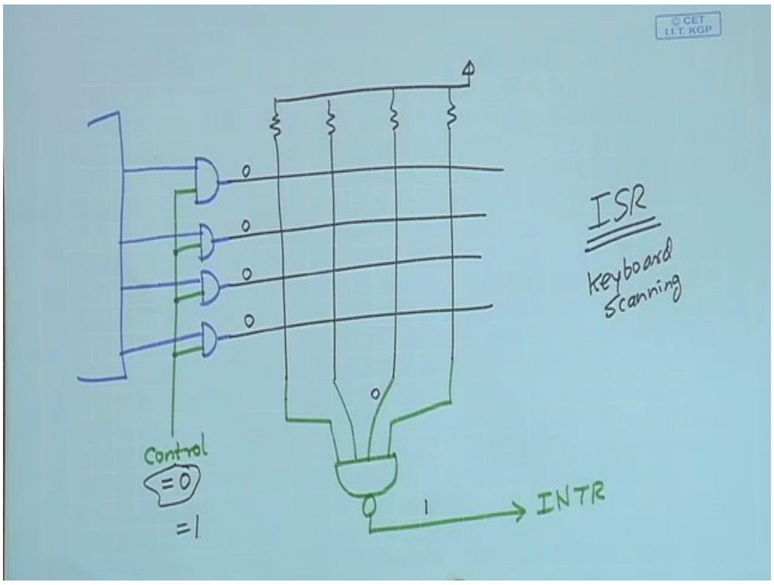
You see using this kind of keyboard scanning method we can identify both the rows and columns. And once you identify both the rows and columns, we can identify which key has been pressed. But before you start this process you have to do this. You have to first check whether a key has been pressed or not. If the key has been pressed, then you move on to the keyboard scanning. This is actually how the thing works.

(Refer Slide Time: 19:29)



You can make some small changes and make the keyboard interface work in an interrupt driven mode also. See in the earlier case just for checking whether any key is pressed or not, you are outputting all 0's and your checking whether the column is all 1's or not.

(Refer Slide Time: 20:10)



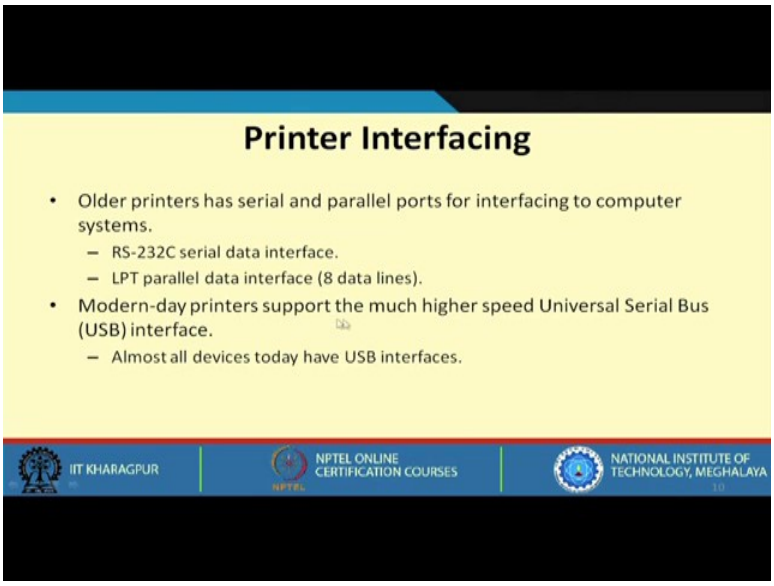
Now, here what we are saying we will do something a little different. I am again drawing the diagram, but I am not showing the switches and the resistances just the rows and columns; I am showing 4 rows and 4 columns. What I am saying is that we will be connecting 4 AND gates and the output of the AND gates will be driving the rows. And the port will be driving one of the inputs of the AND gate. The other input of the AND gate will be made common, this we can call some kind of control. This will also be controlled by the CPU; this will be connected to some other output port.

This is one change you do, and the another change I am suggesting is that you use a NAND gate, and connect the columns to the gate inputs. And the output of this NAND you connect to the interrupt input of the processor. Just think what happens here. You recall these are all pulled up to positive voltage, and when you are checking whether the key is pressed. So, under normal conditions control will be set to 0.

Control 0 means what? One of the input of the AND gate will be 0, so these lines will always be 0 0 0 0. Now suppose no key is pressed, then all these lines will be 1 1 1 1, the output of the NAND will be 0. So, there is no interrupt. But if any key is pressed because you are because you are applying all 0 by default that particular bit will become 0, and correspondingly the output of the NAND gate will become 1, and an interrupt will be generated. So, the processor will know that some key has been pressed; it will jump to the ISR. And inside the ISR the keyboard scanning routine will be there.

So, when you are inside the ISR you are doing keyboard scanning, you will have to change the control to 1 during that time. Because now whatever you are sending over the port that should come to the rows. With this hardware modification you can have interrupt driven keyboard interface. Whatever I have illustrated is written here exactly. The rows and keyboard are connected together, columns are connected, output of the NAND gate will be 1 whenever a key is pressed, and inside the ISR the control inputs of the gates are set to 1, and normal keyboard scanning is carried out to identify the key pressed.

(Refer Slide Time: 23:42)

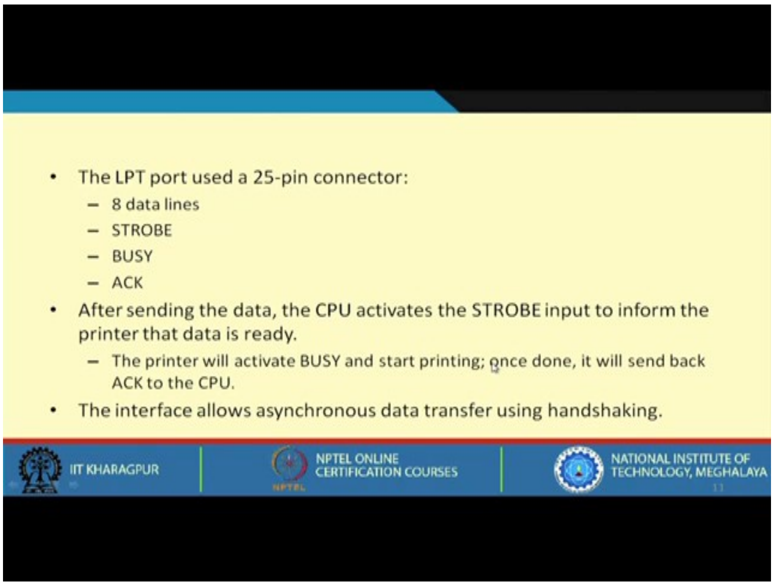


Now, let us very briefly look at a printer interfacing problem, means how some of the IO transfer method we discussed earlier can be used there. If you think of the older printers which are available may be 5, 10 years back, they had either serial or a parallel interface it was called LPT. This parallel interface was almost universally used for printer interfacing.

In the parallel interface there are 8 parallel data lines, but nowadays in the computers you will not see the serial ports and parallel ports anymore. They have become obsolete, but in the old computers you may still see find some serial ports RS232C, and parallel LPT ports available.

Now, almost all printers today support an interface called universal serial bus or USB. USB has become really universal, almost all the devices today have USB interfaces including our mobile phones, cameras, mouse, keyboard, everything.

(Refer Slide Time: 25:22)



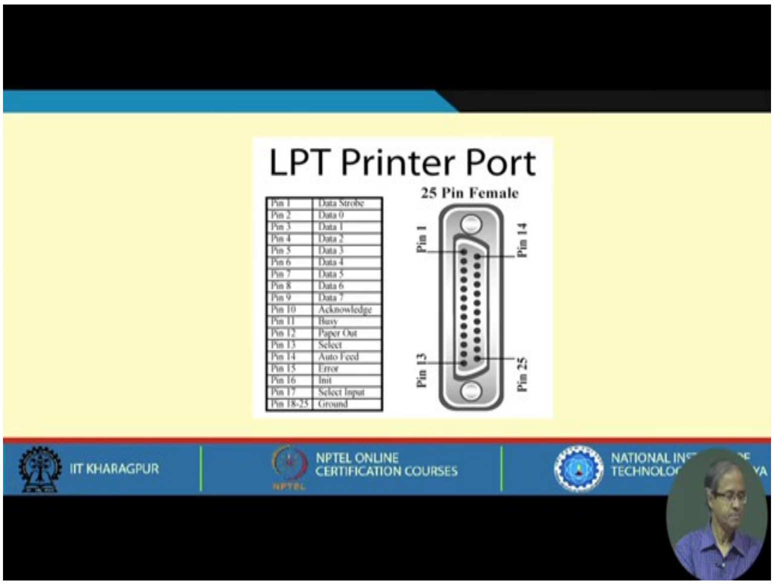
This LPT port used to have a 25-pin connector. All the 25 pins are not used of course, there are 8 data lines. There are power supply lines, positive supply, negative supply and a ground. And there are 3 interesting signal lines: a strobe, a busy, and an acknowledgement, just using which you could carry out asynchronous data transfer with handshaking.

Well how it was done? Well, the printer is an output device. So, CPU will be writing some data to the printer; the data to be printed. After sending the data over the parallel port over this 8 data lines, it could send one byte at a time; CPU will activate the strobe input also indicating to the printer that I have sent one byte of data. The printer can read that 8 bit of data, and during the reading the printer will activate the busy signal, that is going back to the CPU.

Now, after the printer is done, and it is ready to accept the next data, the printer will be sending back the ack signal. Just using the signal you can see you can have handshaking, you can have asynchronous data transfer. There are some additional signals like busy that was printer specific; of course, in a normal handshaking the acknowledgement is enough, but this strobe and busy are required because this allows 2 way handshaking, because

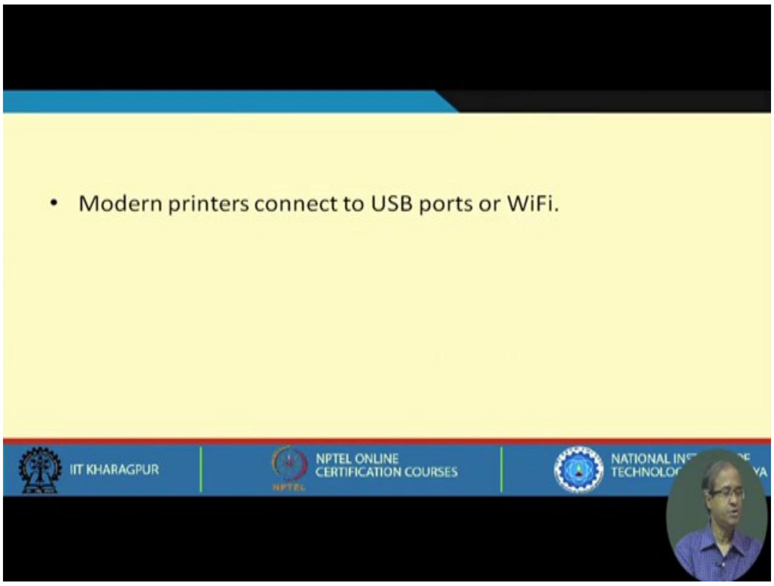
CPU knows that the printer is ready, printer will also know that the CPU has sent a data, both ways.

(Refer Slide Time: 27:23)



The printer port actually looks like this.

(Refer Slide Time: 28:08)



So, this was interface which was quite popular in the early days, but as I had said today if you look at the printers, they have either USB ports or the more sophisticated ones have Wi-Fi connection. They can connect to the wireless and from a computer system you can print wirelessly. What actually I have tried to show through these two examples is that any kind of devices you see, they will have some built in features to carry out some kind of handshaking on asynchronous mode of transfers.

You see newer technologies and newer standards are involving, one of the main objective is to make the interconnection simpler. Earlier the parallel ports carried 25 wires, but now the USB connections if you see there are only few wires. We will be looking into the details of the USB interface later. But here our objective was to look at a couple of examples, but if you take any other device, you will see that some similar features or facilities are there.

With this we come to the end of this lecture. In the next lecture we shall be working out some examples following which we shall be looking into some standard bus interfaces like USB.

Thank you.