

Mechatronics
Prof. Pushparaj Mani Pathak
Department of Mechanical and Industrial Engineering
Indian Institute of Technology, Roorkee

Lecture - 36
Project using Microcontroller - ATMEGA16

I welcome you all to this NPTEL Online Certification Course on Mechatronics. Today, we are going to discuss the Project using Microcontroller, and I am going to talk about the ATMEGA16 microcontroller. This lecture is essentially the first lecture on the last section of our course, that is, the design of mechatronic system design. The intention of this lecture is that the student should be able to use the microcontroller, integrate it with the actuators, sensors and combine or design the desired system.

Here, we are going to talk about is using the ATmega16 microcontroller. How can we design a mobile robot? So, this is the content for this lecture. So, first, we will be seeing the use of or the design of a system for lighting of the different types of LED lights, and then, we will be seeing the use of ATmega16 microcontroller in the design of a mobile robot, a line following mobile robot.

(Refer Slide Time: 02:21)

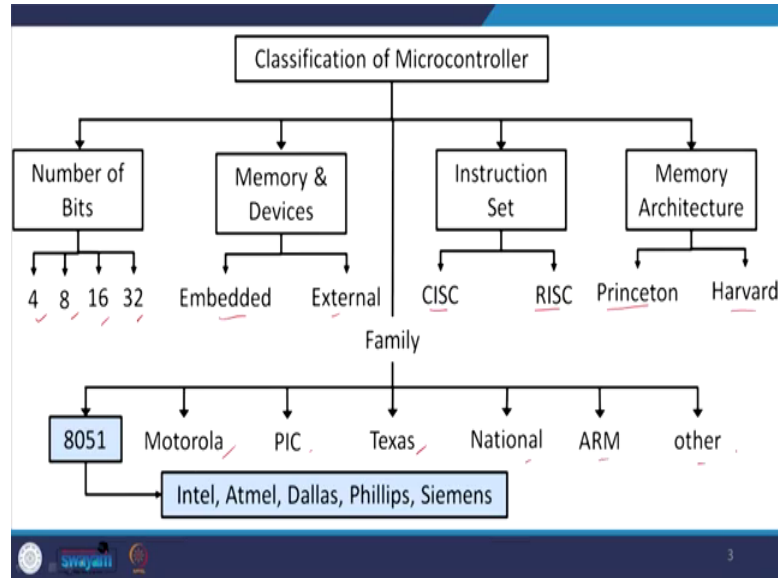
Microcontroller

- Microcontroller is a compact integrated circuit designed to govern a specific operation to control various electronic devices.
- A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip.
- Unique feature of microcontroller: Programmable

The microcontroller, as we have discussed earlier, it is a compact integrated circuit designed to govern a specific operation to control various electronic devices, and a typical

microcontroller includes, it has a processor, memory, and the input-output peripheral on a single chip and the unique feature of this microcontroller is that they are programmable.

(Refer Slide Time: 02:56)



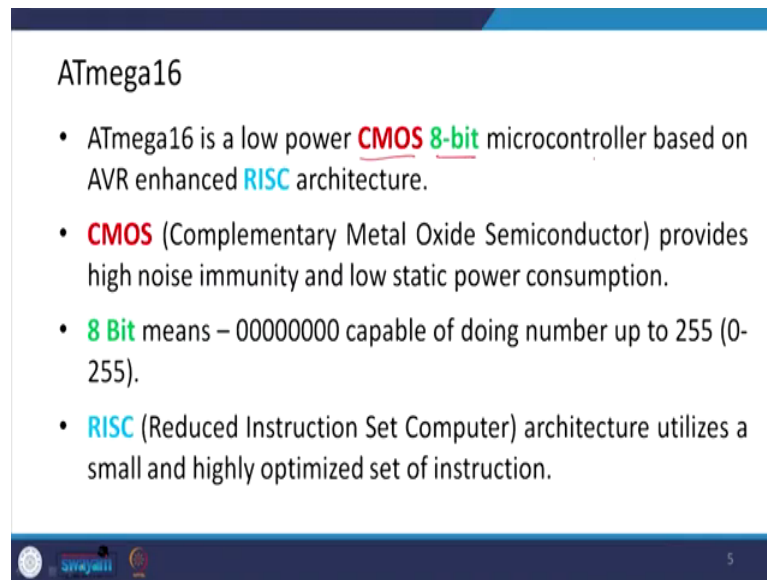
If we look at the classification of the microcontroller, based on the number of bits, the classification is there 4 bit, 8 bit, 16 bit, or 32 bit. Then memory and devices are embedded memory or an external one. Instructions set CISC type or RISC type. Then, a memory architecture, either Princeton or the Harvard type of architecture, and they could form the family 8051, which is Intel, Atmel, Dallas, Philips, Siemens, or they could be from Motorola, PIC, Texas, National ARM, or other types.

(Refer Slide Time: 03:47)

| Feature | Microprocessor | Microcontroller |
|-----------------|--------------------------------------|---------------------------------|
| Application | General Computing (Laptops, tablets) | Appliances, specialized devices |
| Speed | Very fast | Relatively slow |
| External parts | Many | Low |
| Cost | High | Low |
| Required Energy | Medium to High | Very low |

Microcontroller versus microprocessor, let us see some of them or some important basic features of that, details I have already talked about in my earlier lecture on the microprocessor and microcontroller. The features are from the application. The microprocessor is used for general computing purposes. For example, laptops and tablets and the microcontroller are used in appliances or specialized devices. Speed, this is a very fast microprocessor. The microcontroller is relatively slow. External parts, there are many in the microprocessor, there are few in the microcontroller. Cost is high, these are the low-cost devices, and the required energy is medium to high, and in the case of microcontrollers, the energy is required very low.

(Refer Slide Time: 04:58)



ATmega16

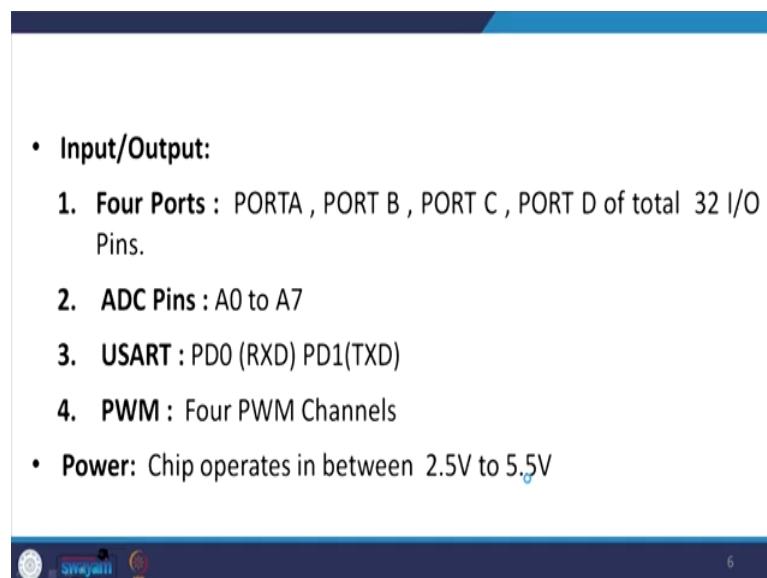
- ATmega16 is a low power **CMOS 8-bit** microcontroller based on AVR enhanced **RISC** architecture.
- **CMOS** (Complementary Metal Oxide Semiconductor) provides high noise immunity and low static power consumption.
- **8 Bit** means – 00000000 capable of doing number up to 255 (0-255).
- **RISC** (Reduced Instruction Set Computer) architecture utilizes a small and highly optimized set of instruction.

5

Now, let us talk about ATmega16. It is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. A CMOS (complementary metal-oxide-semiconductor) provides high noise immunity and low static power consumption, and it has 8-bits, 8-bit means that you are going to have the eight numbers over here and capable of doing a computation up to 255.

Now, RISC architecture is the reduced instruction set computer architecture that utilizes a small and highly optimized set of instructions.

(Refer Slide Time: 05:48)

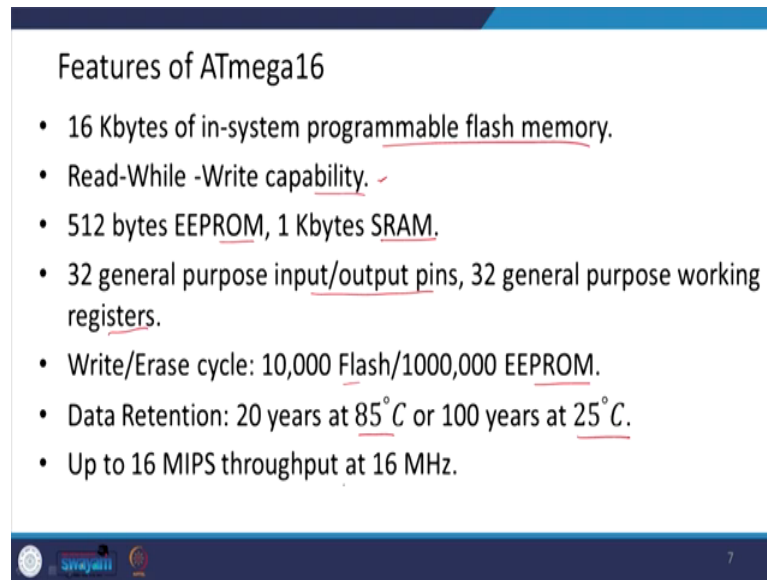


- **Input/Output:**
 1. **Four Ports** : PORTA , PORT B , PORT C , PORT D of total 32 I/O Pins.
 2. **ADC Pins** : A0 to A7
 3. **USART** : PD0 (RXD) PD1(TXD)
 4. **PWM** : Four PWM Channels
- **Power:** Chip operates in between 2.5V to 5.5V

6

Looking at the input and output ports of ATmega16, there are four ports, PORTA, PORT B, PORT C, and PORT D, of the total of 32 input-output pins. Analog to digital converter pins are there A0 to A7. USART PD0 that is RXD, PD1 that is the TXD. Four PWM channels are there, and the power is chip operates in between 2.5 volts to 5.5 volts.

(Refer Slide Time: 06:25)



The slide is titled "Features of ATmega16" and lists the following features:

- 16 Kbytes of in-system programmable flash memory.
- Read-While-Write capability.
- 512 bytes EEPROM, 1 Kbytes SRAM.
- 32 general purpose input/output pins, 32 general purpose working registers.
- Write/Erase cycle: 10,000 Flash/1,000,000 EEPROM.
- Data Retention: 20 years at 85° C or 100 years at 25° C.
- Up to 16 MIPS throughput at 16 MHz.

The slide also features a footer with a logo on the left and the number "7" on the right.

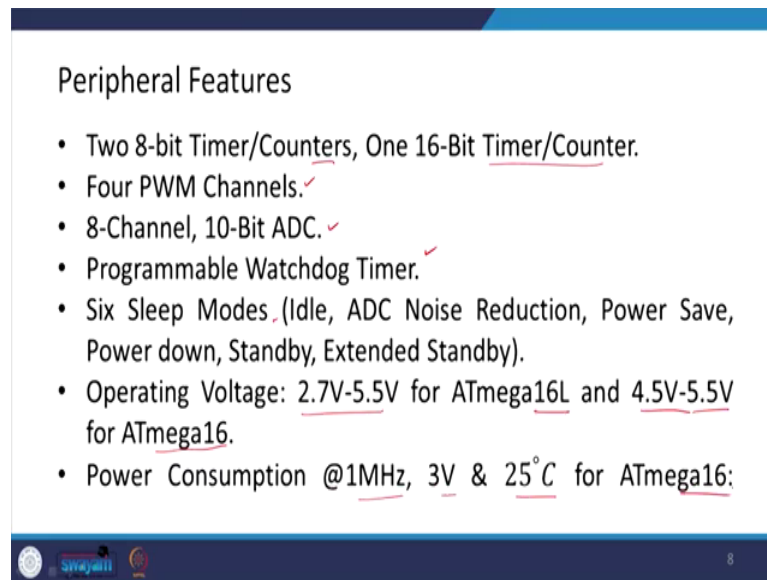
Coming to the features of ATmega16, there is a 16 kilobyte of in-system programmable flash memory and Read-While-Write capability. It has 512-byte EEPROM is there, 1 kilobyte SRAM is there, 32 general-purpose input, output pins are there, 32 general-purpose working registers are there.

Write erase cycle it is the 10000 flash or 1 million EEPROM. Data retention, 20 years at 85° C or 100 years at 25° C, up to 16 MIPS throughput at 16 mega Hertz. So, these are the features of ATmega16.

(Refer Slide Time: 07:18)

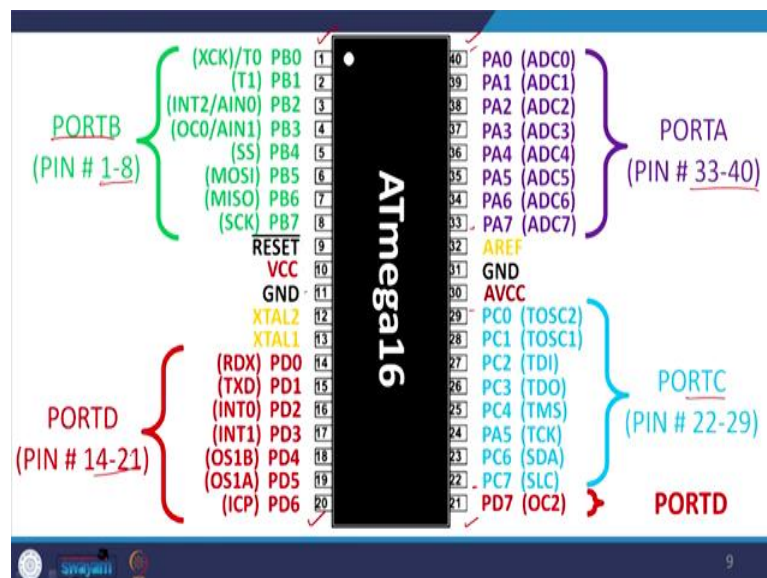
Peripheral Features

- Two 8-bit Timer/Counters, One 16-Bit Timer/Counter.
- Four PWM Channels. ✓
- 8-Channel, 10-Bit ADC. ✓
- Programmable Watchdog Timer. ✓
- Six Sleep Modes (Idle, ADC Noise Reduction, Power Save, Power down, Standby, Extended Standby).
- Operating Voltage: 2.7V-5.5V for ATmega16L and 4.5V-5.5V for ATmega16.
- Power Consumption @1MHz, 3V & 25°C for ATmega16:



The peripheral features are two 8-bit timers counters. One 16-bit timer counter is there in it. Four PWM channels, as I discussed earlier. 8-channels, 10-bit analog to digital conversion. A programmable watchdog timer is there. Six sleep modes that are idle, ADC noise reduction, power saving, power down, standby, extended standby is there. Then, the operating voltage is 2.7 to 5.5 volt for ATmega16L and 4.5 to 5.5 volt for ATmega16 and power consumption at the rate 1 mega Hertz, 3 volts, and 25⁰ C for ATmega16.

(Refer Slide Time: 08:11)



Now, let us look at the PIN configuration. So, here, you can see the PIN numbering from 1 to 20 is left-hand side and 21 to 40 is the right-hand side. We have PORTB and it has PIN 1 to 8 that is PB PORTB0, PORTB1, PORTB2, PORTB3 and so on. Then, the 9th is RESET, then this is the voltage supply, then you have the ground. Then PORTA is from 33 to 40 is PORTA0, PORTA1 so on. Coming to the PORTC over here so, you can see that PORTC, from 22 to 29 is again PORTC over here, and PORTD is 14 to 21 is here.

(Refer Slide Time: 09:46)

ATmega16 Pin Description


- ATmega16 have total of 40 pins from which 32 are I/O pins.
- VCC (PIN10): Digital Supply Voltage.
- GND (PIN11): Ground
- PORT A (PA0-PA7): Port A serves as
 1. The analog input to the A/D converter.
 2. 8-bit bi-directional I/O port, if the A/D converter is not used.

10

These things you can see in the catalog for ATmega16 also. So, ATmega16 has a total of 40 pins, out of which 32 are the input-output pins, as I discussed earlier. VCC is the digital supply voltage pin number 10 indicated like this. Pin number 11 is the ground indicated this way. PORTA that is pinned A from 0 to 7, so, that is, there are eight pins. So, this port serves as the analog input to the analog to digital converter. 8-bit bidirectional input-output port if the analog to digital converter is not used. So, we can use all the eight pins for the bidirectional input-output that is from PA0 to PA7.

(Refer Slide Time: 10:47)

- **PORT B (PB0-PB7)** , **PORT C (PC0-PC7)** & **PORT D (PD0-PD7)**: 8-bit bi-directional I/O port with internal pull-up resistors selected for each bit.
- **RESET**: Reset input a low level on this pin for longer than the minimum pulse length will generate a reset.
- **AVCC**: It acts supply voltage pin for PORT A and the A/D converter.
- **AREF**: AREF is the analog reference pin for the A/D converter.
- **XTAL 2**: Output to inverting oscillator amplifier.




Port B is again PB0 to PB7, PORT C is PC0 to PC7, and PORT D is PD0 to PD7 so, each 8 port is. So, an 8-bit bi-directional input-output port with an internal pull-up resistor is selected for each bit. REST, the reset input a low level on this pin for larger than the minimum pulse length will generate a reset. AVCC acts as a supply voltage pin for PORTA and the analog to digital converter. AREF is the analog reference pin for the analog to digital converter, and XTAL 2 is the output to inverting oscillator amplifier.



(Refer Slide Time: 11:38)

PDIP Vs TQFP

Plastic dual-in-line Package abbreviated as PDIP is a chip-package with rectangular housing and two parallel rows of electrical connecting pins.



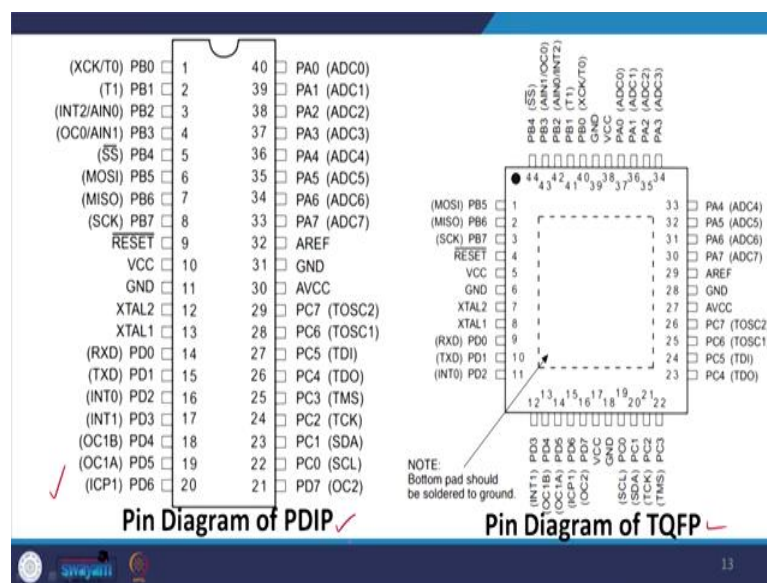
Thin Quad Flat Package abbreviated as TQFP is a chip-package with quad housing with "gull wing" leads extending from each of four side



Now, let us look at how these are available in the market. So, there are two forms, the PDIP and the TQFP is. The PDIP is plastic dual-in-line packaging. This is how it is available in the market. And this is a rectangular housing, and there are two parallel rows of electrical connecting pins, as you can see over here and there is in the backside.

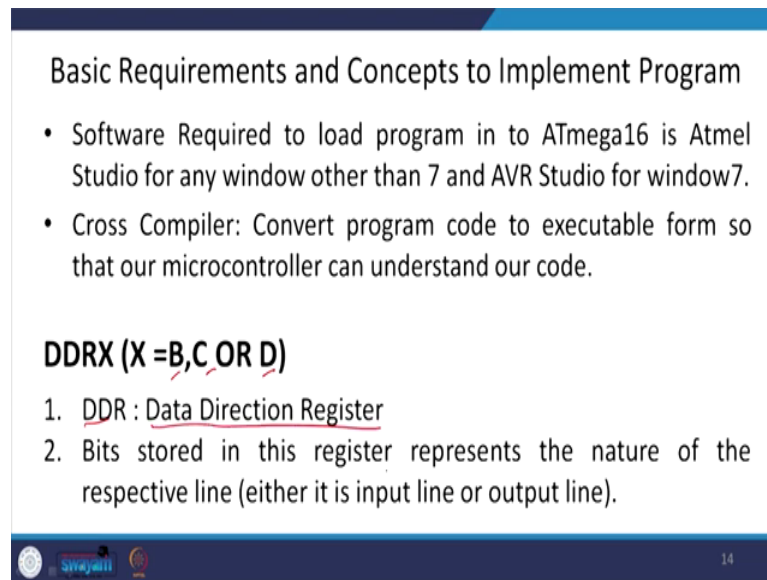
And the TQFP is a thin quad flat packaging so. The thin quad flat package is a chip package with quad housing with a gull-wing lead extending from each of the four sides. So, you can see that here you have each of the four sides you have the connecting points.

(Refer Slide Time: 12:37)



This is the pin diagram for the PDIP type, and this is the pin diagram for the TQFP type. So, you can see that you have pin connections on all four sides, and I have already discussed these various ports in my earlier slide.

(Refer Slide Time: 13:01)



Basic Requirements and Concepts to Implement Program

- Software Required to load program in to ATmega16 is Atmel Studio for any window other than 7 and AVR Studio for window 7.
- Cross Compiler: Convert program code to executable form so that our microcontroller can understand our code.

DDRX (X =B,C OR D)

1. DDR : Data Direction Register
2. Bits stored in this register represents the nature of the respective line (either it is input line or output line).

swajati 14

So, now, let us look at the basic requirements and the concept of implementing the program. The software requires to load the program into ATmega16 is Atmel Studio for any window other than 7 and AVR Studio for windows 7. There is a cross compiler that converts program code to executable form so that our microcontroller can understand our code.

And the DDRX, let us look at here X could be your B, C or it could be D. So, this DDR is what we call it as the data direction register. So, the bits stored in this register represent the nature of the respective line. Either it is the input line, or it is the output line.

(Refer Slide Time: 14:06)

3. If stored value is 0, it means that respective line is acting as input line (for connecting input devices like sensors).

4. Similarly if value is 1, it means that respective line is acting as output line (for connecting output devices).

5. Ex: **0b**-Binary, **0x**-Hexadecimal, Number like **15**-Decimal

Pin A3 to Pin A0 can only be used for output

Pin A7 to Pin A4 can only be used for input

DDR(A) = 0b 0000 1111

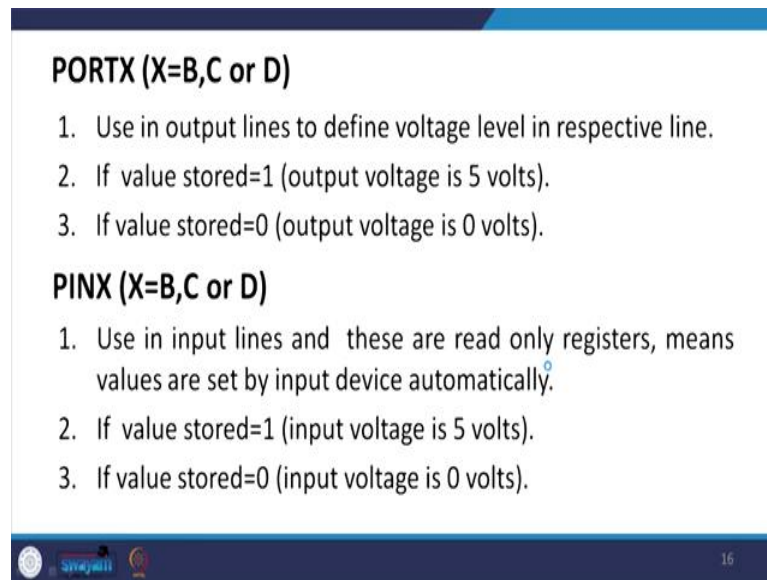
Port A

If the stored value is 0, it means that the respective line is acting as the input line for connecting input devices like the sensor. Similarly, if the value is 1, it means that the respective line is acting as an output line for connecting the output devices.

An example could be given as DDR, this A stands for Port A, and this is 0b so, 0b is for the binary, and we could have the 0x for the hexadecimal that is there.

And then, we have a pin as A7 to pin A4 can only be used for the inputs over here. So, pin A3 to pin A0 can only be used for output. So, these four are that is A0, A1, A2, A3 here, and this is A4, A5, A6, A7. So, these are used for the input, and these are used for the output because we are talking about Port A so, the corresponding pin numbers are there.

(Refer Slide Time: 15:45)



PORTX (X=B,C or D)

1. Use in output lines to define voltage level in respective line.
2. If value stored=1 (output voltage is 5 volts).
3. If value stored=0 (output voltage is 0 volts).

PINX (X=B,C or D)

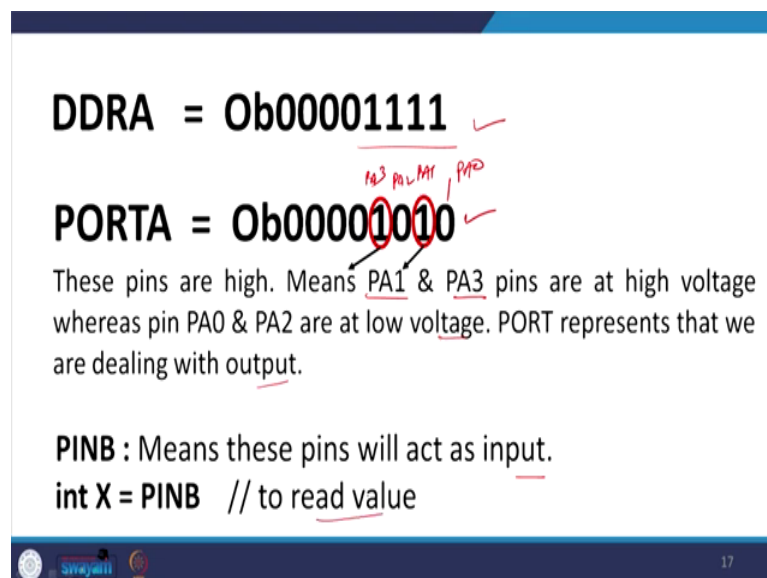
1. Use in input lines and these are read only registers, means values are set by input device automatically.
2. If value stored=1 (input voltage is 5 volts).
3. If value stored=0 (input voltage is 0 volts).

16

Now, for PORTX, X for B, C, or D. Use in output line to define voltage level in a respective line. If a value is stored is 1, then the output voltage if 5 volt, and if the value is stored is 0, it means that the output voltage is 0 volt.

For PINX, if X could be B, C, or D and this is used in the input line and those, are read-only register means values are set by input devices automatically and if a value is stored is 1 means the input voltage is 5 volt and if the value is stored is 0, it means that the input voltage is 0 volt.

(Refer Slide Time: 16:41)



DDRA = 0b00001111 ✓

PORTA = 0b00001010 ✓

These pins are high. Means PA1 & PA3 pins are at high voltage whereas pin PA0 & PA2 are at low voltage. PORT represents that we are dealing with output.

PINB : Means these pins will act as input.

int X = PINB // to read value

17

So, DDRA could be like this, and PORTA could be like this. So, here, these pins are high means that PA1 and PA3 are high voltage, whereas PA0 and PA2, this is your PA0; this is your PA1, this is PA2, and this is PA3.

So, PA1 and PA3 pins are high whereas, PA0 and PA2 are low voltage. PORT represents that we are dealing with the output. PINB means this pin will act as input, so int X is equal to PINB so, to read the values.

(Refer Slide Time: 17:47)

```
Example: Program to Glow LED
#define F_CPU 12000000UL // To Define frequency
#include <avr/io.h>
#include <util/delay.h> // Directory for Delay
int main(void)
{
  DDRB=0b00011110; // 1-Pins will act as output pins of B
  while(1)
  {
```

Now, let us look at a program to glow the LED. So, as I said earlier, we are going to see two examples, one on the glowing of the LED and one on the control of or the design of the line following robot using ATmega16. Here we can define a frequency over here so, this is there, then we can include certain files, to include directory for the delay we can have include function like this, and then, we can have this DDRB, 1-pin will act as output pin of B, and then, we could have while 1.

(Refer Slide Time: 18:47)

```
PORTB=0b00011000; // Glow LED connected to PB3 & PB4
_delay_ms(1000); // Delay of 1 sec
PORTB=0b00000110; // Glow LED connected to PB1 & PB2
_delay_ms(1000); // Delay of 1 sec
PORTB=0b00011110; // Glow LED connected to PB1 to PB4
_delay_ms(1000); // Delay of 1 sec
PORTB=0b00000000; // Do not glow any LED
_delay_ms(1000);
}
```

PORTB can write it like this for glowing LED connecting PB3 and PB4. So, this is 0, 1, 2, and so, this is 3, this is your PB0, this is PB1, this is PB2, and this is your PB3, and this is your PB4. So, glow LED connecting PB3 and PB4 and then, we can have a delay of 1 second. And then, we can have a command for go glow LED for PB1 and PB2 over here, then we can have a delay of 1 second and then, go glow LED for PB1 to PB4 so, 1, 2, 3, 4 all these values are 1 so, it will go LED connecting PB1 to PB4, then again delay of 1 second and do not glow any LED. So, all these values are going to be 0 over here, and then, we can have a delay of 1 second, and then finally, we can close that.

(Refer Slide Time: 20:04)

```
Program of Line Follower Robot
// Sensor 1 & sensor 2 are connected to PC0 & PC1 respectively.
// Motor 1 connected to PB1 & PB2 ; motor 2 connected to PB3 & PB4.
#include <avr/io.h>
int main(void)
{
    DDRC=0b00000000; //Optional
    DDRB=0b00011110; //Define PB1,PB2,PB3,PB4 as output lines
    int sensor=0; //Variable to store status of sensors
    while(1) {
```

Next, let us look at the program of the line follower robot. Now, here for this line follower, as you can see in the comment line, sensor 1 and sensor 2 are connected to PC0 and PC1 respectively of the Atmega16 microcontroller, and motor 1 is connected to PB1 and PB2, and motor 2 is connected to PB3 and PB4. Here again, we can see this one. First is include function, we can start the main program, we define the DDRC here, then DDRB that is defined PB1, PB2, PB3, PB4 as output line and then variable to store the status of the sensor so, that we define over here.

(Refer Slide Time: 21:16)

```

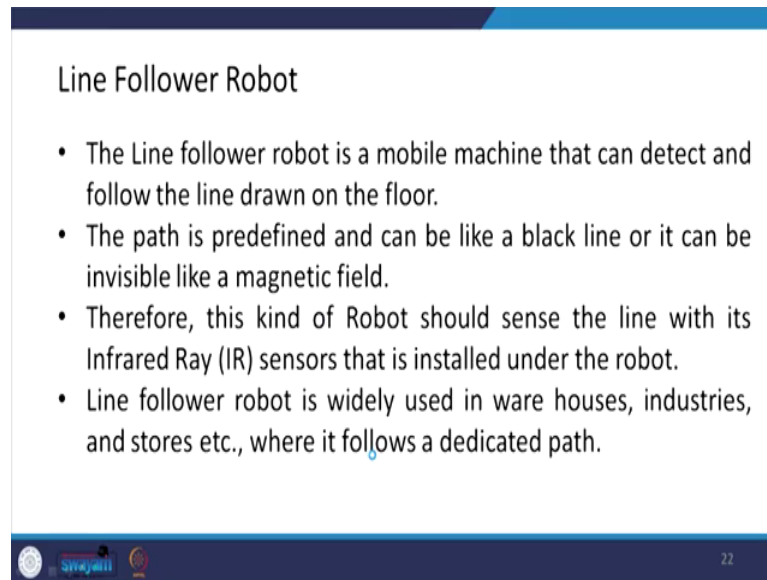
sensor= PINC & 0b0000011; //Statement to store value from pin
                             register to variable
if(sensor==0b0000011) //Conditional statements
{PORTB=0b00001010;}
else if (sensor==0b0000001)
{PORTB=0b00000010;}
else if(sensor==0b0000010)
{PORTB=0b00001000;}
else //Condition when both sensors are off
{PORTB=0b00000000;} } }

```

Then, the sensor is PINC and this one. So, statement to store value from pin register to a variable and then, we can see that if the sensor is these two are that is 1, then the PORTB here that is it should be 1 and 1 over here. If you look at the earlier one so, your motor is connected to PB1 PB2, and motor 2 is connected to PB3, PB4, and sensor 1 and sensor 2 are connected to PC0 and PC1 over here. So, PC0 and PC1 so, are the sensor values. So, these are the sensor values, and these are the actuator values. So, what does this means that is PB1 and PB3. So, PB1 and PB3 are my motor 1 and motor 2 both are being there 0, 1, 2, 3. So, both these motors will be running. So, this is what is meant by it, and if the sensor is 1 that is one sensor is 1, then you have to use that is the motor of PB1 I has to run, maybe I will discuss the robot structure and logic, then maybe this could be understood further.

And if the sensor is this one that is if your second sensor value is 1, then your this that is PB3; PB3 has to run, and otherwise, if both the sensors are off, then no motor should run. So, this is what is meant by it.

(Refer Slide Time: 24:00)



The slide is titled "Line Follower Robot" and contains a bulleted list of four points. The slide has a dark blue header and footer. The footer includes a logo on the left, the word "swayam" in the center, and the number "22" on the right.

Line Follower Robot


- The Line follower robot is a mobile machine that can detect and follow the line drawn on the floor.
- The path is predefined and can be like a black line or it can be invisible like a magnetic field.
- Therefore, this kind of Robot should sense the line with its Infrared Ray (IR) sensors that is installed under the robot.
- Line follower robot is widely used in ware houses, industries, and stores etc., where it follows a dedicated path.

The line follower robot, as I said, is a mobile machine that can detect and follow the line drawn on the floor. The path is predefined and can be like a black line, or it can be invisible like a magnetic field. Therefore, this kind of robot should sense the line with its IR or the infrared sensor that is installed under the robot. Now, the line follower robot is widely used in warehouses, industries, stores, etcetera, where it follows a dedicated path.

(Refer Slide Time: 25:03)

List of Equipment's Required

- Atmega16 Development Board.
- Two Motor, two Wheels and Caster Wheel
- Chassis & Fitting Packet, Screwdriver & 40 Pin Wires
- 2 IR Sensor, 1 USB Programmer and Battery Snipper.



Atmega16 Motor and Wheel Pin wires and Chassis USB Programmer IR Sensor

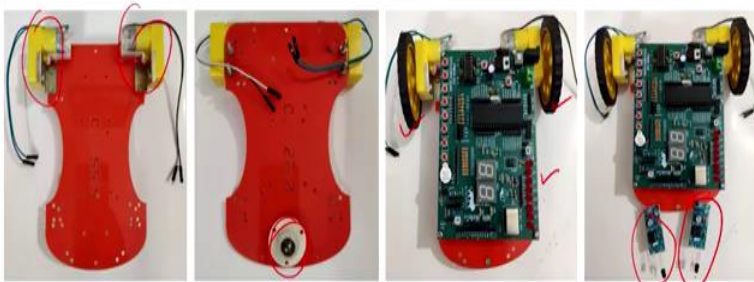
23

What are the types of equipment required for making this line follower robot? So, as I said, we are going to use the ATmega16 development board. We are going to use this one ATmega16 development board, the two motors here, the two wheels, and the caster wheel. So, we have a motor here, and this is a wheel. The caster wheel is not shown. I will be showing it in the maybe next slide. Then, a chassis and fitting packet, screwdriver and 40 pin wires and the 2 IR sensors, 1 USB programmer, and the battery snipper. So, you can see pin wires and chassis, pin wires and a chassis is here, USB programmer is here, and the IR sensor is there. So, we require two such IR sensors.

(Refer Slide Time: 25:50)

Connection Guide

- The step-by-step connection guide for Line Follower Robot with ATmega16 using Analog IR Sensor is as follows:

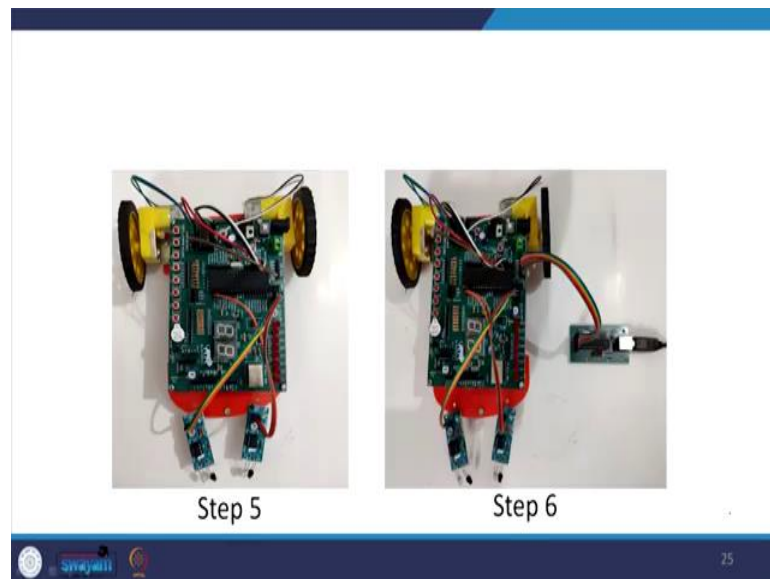


Step 1 Step 2 Step 3 Step 4

24

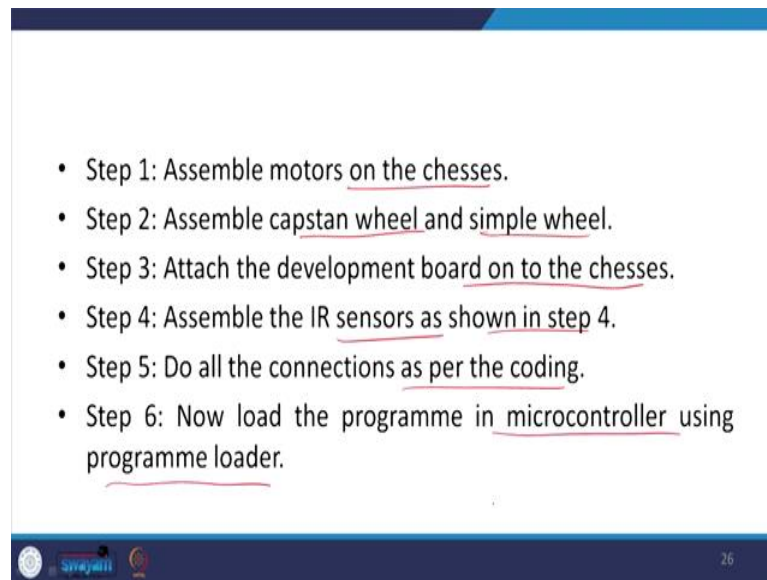
To connect step by step connection can be explained. In the first stage, what do we do? We fix up the two motors on the chassis over here. So, that is done, and then we could connect the caster wheel over here and here. You can see that I connect the two other wheels, and I have the ATmega16 microcontroller over here, and here, I connect the 2 IR sensors. And in this IR sensor, you can see that there is one transmitter and one there is the receiver.

(Refer Slide Time: 26:36)



Here, you can see that we do all the wiring, and then we connect it through the USB program finally.

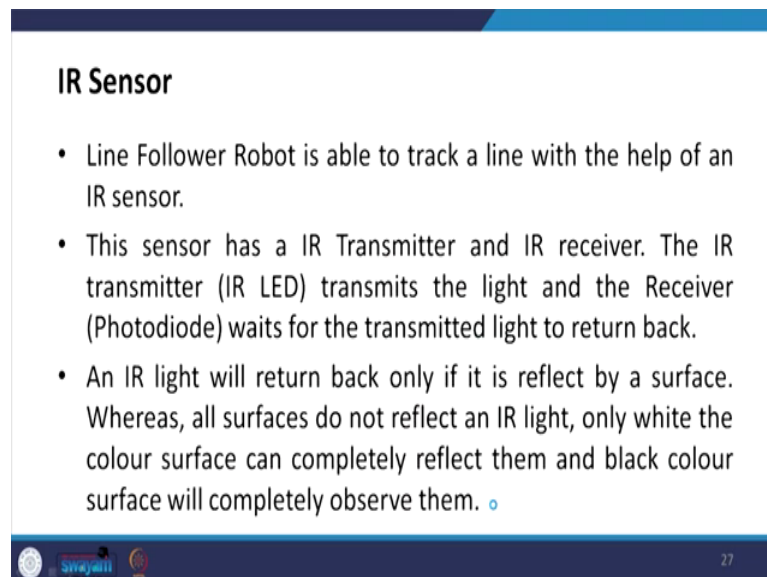
(Refer Slide Time: 26:49)



- Step 1: Assemble motors on the chasses.
- Step 2: Assemble capstan wheel and simple wheel.
- Step 3: Attach the development board on to the chasses.
- Step 4: Assemble the IR sensors as shown in step 4.
- Step 5: Do all the connections as per the coding.
- Step 6: Now load the programme in microcontroller using programme loader.

So, the steps are to assemble the motor to the chassis, as I explained. Assemble capstan wheel and the simple wheel there, then attach the development board on the chassis, then assemble the IR sensor as shown there, make all the connections as per the coding, now load the program in microcontroller using the program loader.

(Refer Slide Time: 27:14)



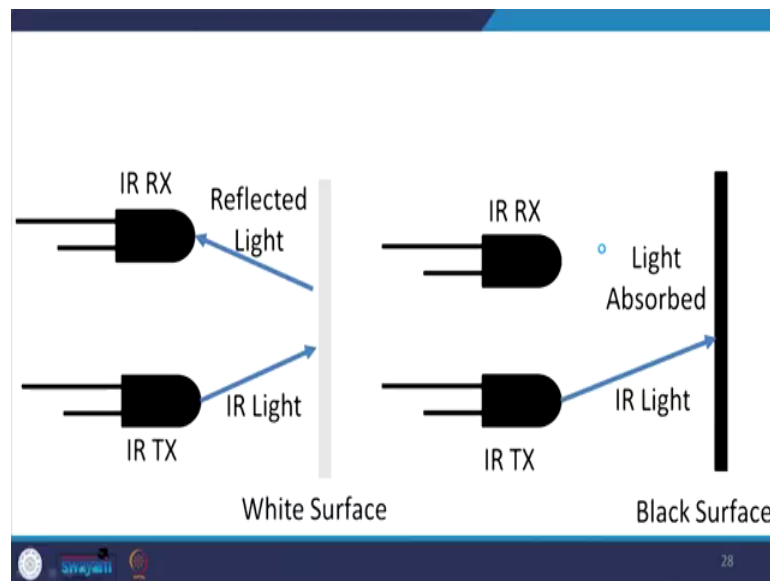
IR Sensor

- Line Follower Robot is able to track a line with the help of an IR sensor.
- This sensor has a IR Transmitter and IR receiver. The IR transmitter (IR LED) transmits the light and the Receiver (Photodiode) waits for the transmitted light to return back.
- An IR light will return back only if it is reflect by a surface. Whereas, all surfaces do not reflect an IR light, only white the colour surface can completely reflect them and black colour surface will completely observe them. ◦

Now, IR sensor, although I have talked about it in my lecture on the sensors, for the sake of completeness, I will just briefly explain here how we are using this sensor in the line follower robot.

The line follower robot is able to track a line with the help of an IR sensor. This sensor has an IR transmitter and an IR receiver. And the IR transmitter transmits the light and the IR receiver, which is nothing, but a photodiode, and the IR transmitter is nothing but an IR LED. The receiver waits for the transmitted light to return back, and it detects. An IR light will return back only if it is reflected by a surface, whereas all surfaces do not reflect an IR light. The only white surface can completely reflect them, and the black color surface will completely observe them.

(Refer Slide Time: 28:35)

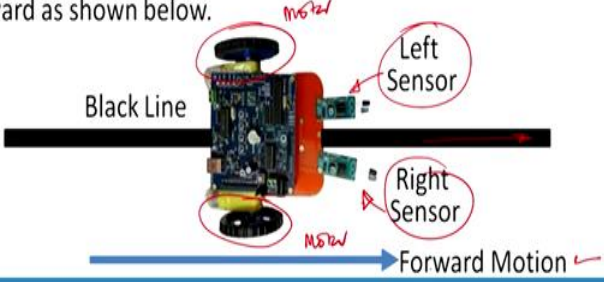


So, here, if you have a wide surface, you have the IR transmitter so, IR signal is transmitted, and the white surface will reflect, and it will be sensed by the IR receiver whereas, if you have a black surface, then the this IR transmitter will be sending and this will be received by or absorbed by the black surface completely. So, the IR receiver is not going to receive anything.

(Refer Slide Time: 29:14)

Both Sensors on White Surface

- These two IR sensors will be placed one on either side of the line. If none of the sensors are detecting a black line then microcontroller instructs both the motors to move forward as shown below.



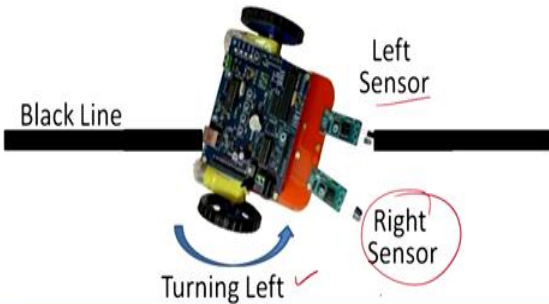
The diagram shows a mobile robot with a microcontroller board and two IR sensors. The robot is positioned on a white surface with a black line. The left sensor is on the white surface, and the right sensor is also on the white surface. The robot is moving forward, as indicated by the blue arrow labeled 'Forward Motion'. The left motor is labeled 'M672d' and the right motor is labeled 'M572d'. The left sensor is labeled 'Left Sensor' and the right sensor is labeled 'Right Sensor'. The black line is labeled 'Black Line'.

Now, if both the sensors are on the white surface, so, this is how we are asking our mobile robot to follow this step. So, this is the forward motion direction, this is your left sensor, IR sensor, and you have the right IR sensor over here, and you see, and here, you have the right motor, and here, you have the left motor, motor, and wheel all right. Here when both sensors are on a white surface, as you can see. So, these two IR sensors will be placed one on either side of the line. If none of the sensors detect a black line, then the microcontroller instructs both the motors to move and forward, as shown here. So, this is the first condition.

(Refer Slide Time: 30:23)

Left Sensor on Black Line and Right Sensor on White Surface

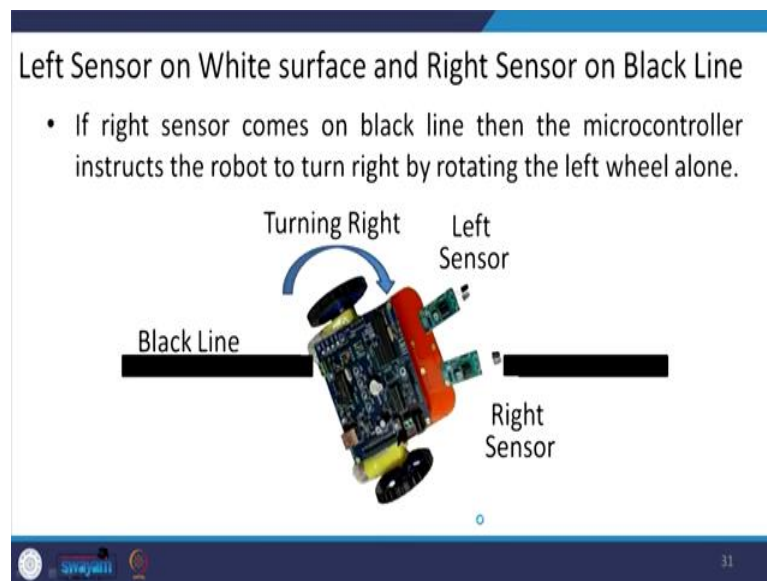
- If left sensor comes on black line then the microcontroller instructs the robot to turn left by rotating the right wheel alone.



The diagram shows the same mobile robot as in the previous slide. The left sensor is now on the black line, and the right sensor is on the white surface. The robot is turning left, as indicated by the blue curved arrow labeled 'Turning Left'. The left sensor is labeled 'Left Sensor' and the right sensor is labeled 'Right Sensor'. The black line is labeled 'Black Line'.

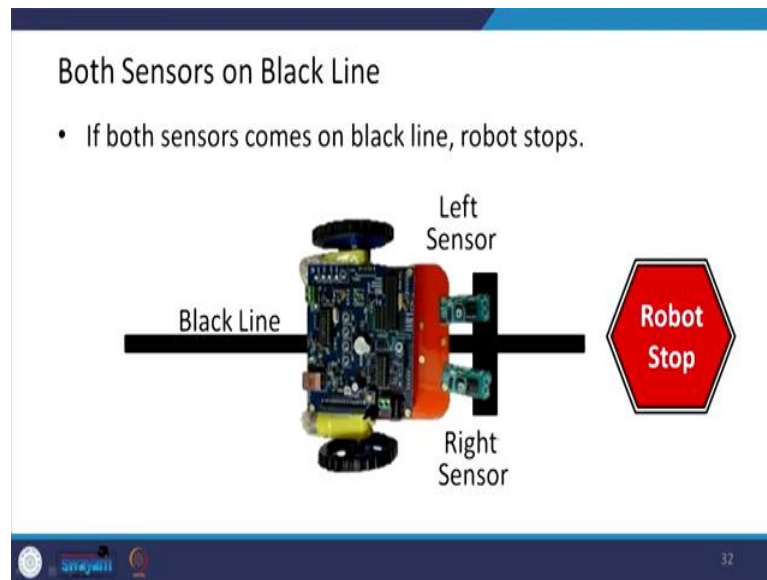
Then, the left sensor is on the black line, and the right sensor is on the white line. So, this right sensor position is the same. It is there on the white surface, whereas the left sensor has come upon the black surface over here. Now, if the left sensor comes on the black line, then the microcontroller instructs the robot to turn left by rotating the right wheel alone. So, if this left sensor is there on the black line, then this motor is actuated so that it turns towards the left direction. So, this is there.

(Refer Slide Time: 31:19)



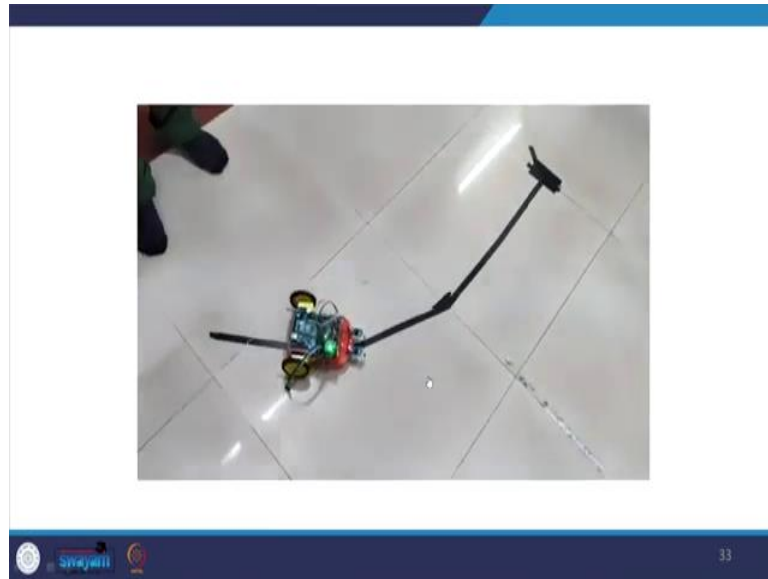
Then, if the left sensor is on a white surface and the right sensor is on the black line. So, if this is there, then it means that the robot has turned left so, it needs to be turned right. So, that is, the motor of the left side is being activated, and this instruction is given by the microcontroller. So, if the right sensor comes to the black line, then the microcontroller instructs the robot to turn right by rotating the left wheel alone.

(Refer Slide Time: 32:08)



If both the sensors are there on the black line, then the robot is stopped. So, your right sensor is there on; the left sensor is there on the black line, and the right sensor is also there on the black line, then both the motor stops and your robot stops. These three conditions are programmed what I was explaining to you over here in this over here. So, this was the first condition, this was the second condition, this was the third condition, and this is the stoppage of that one, the stoppage of the robot. The first one is both the IR sensor are there on the white surface so, both these motors are running, moving there and here, 1 is on the white surface, and here again, 1 is on the white surface. So, only the single motor is being actuated, as I explained to you.

(Refer Slide Time: 33:49)



With this, we can see that our robot is able to follow; our robot is able to follow the path. So, here you can see that this is the black line which the robot is supposed to follow, and this will follow the path, and finally, it will stop over there. So, you can see that the robot follows the path, and here, it will stop when both are there in the black strip. In this lecture, what we have seen is the various feature of the ATmega16 microcontroller, and then, we have seen u integration of actuator that is the DC motor to the microcontroller, integration of the sensor to the microcontroller, and then, we have seen the programming of the microcontroller for the robot to follow a straight-line path.

With this, I would like to thank you for trying to do this simple experiment. It is really very interesting, and it will be testing your skill of whatever you have learned in the mechatronics course here. So, the lecture of today and the beyond are on the design of the various type of mechatronic systems, which you will see through.

Thank you.