Automation in Manufacturing
Dr. Shrikrishna N. Joshi
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

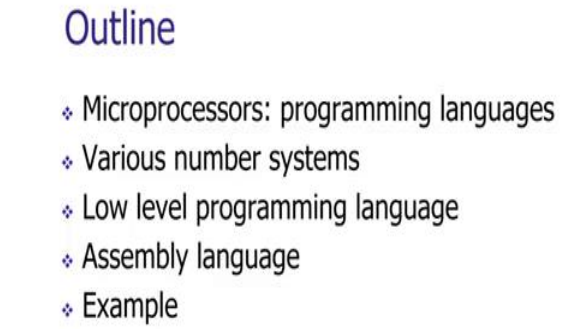
Week – 05
Signal conditioning and Microprocessor
Lecture – 05
Introduction to Microprocessor Programming

(Refer Slide Time: 00:28)

We are at the last lecture of Week 5 that is Lecture 5. In this week, we have seen various important aspects of development of an automated system that is a Signal Conditioning and Microprocessor Technology.

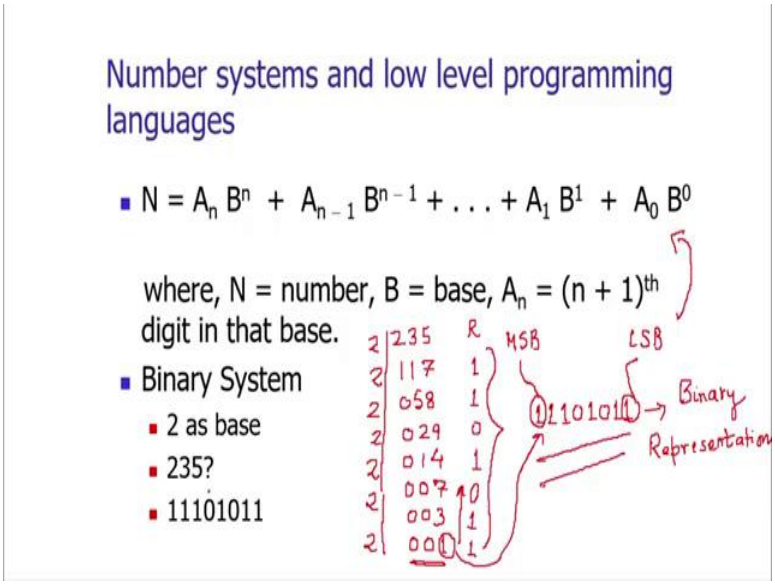
In this lecture, we will be studying how to program a typical microprocessor. My friends, microprocessor is a very vast area. There are lot of courses available on this technology. In view of the limited time we will be focusing on the Introduction to the Microprocessor Programming.

(Refer Slide Time: 01:11)



The outline of this lecture is as follows. At the start of the lecture, we will see what are the various programming languages available for microprocessors, what are the various number systems being used in the programming languages, then we will study the meaning of low level programming language, after that we will see what is the meaning of assembly language. To understand the syntax of the assembly language we will solve an example. We will carry out a case study.

(Refer Slide Time: 02:02)



At start of this lecture which is on introduction to microprocessor programming, let us see what are the various number systems being used in microprocessor programming. The basic number system is the binary number system that is often used in the microprocessors. As we know very well, the binary system has the base 2, and in general in our day to day life we are using the decimal system which is having the base 10.

In the slide, we can see a formula which is used to convert decimal number into a binary number. The binary representation can be find out by using this formula.

$$N = A_n B^n + A_{n-1} B^{n-1} + ... + A_1 B^1 + A_0 B^0$$

N is a number in decimal format with base 10 and B is the base in which we have to convert it.

Let us see how this formula can be utilized to convert a decimal number into a binary number. In the binary system, we are using 2 as the base. Now, let us try to find out is the binary representation of the number 235 into binary number. This is just a revision most of you may be knowing about the conversion of the decimal 2 the binary system.

Let us find out how we can convert the number 235 with base 10 into its binary equivalent. This number will be divided by 2 and try to find out the reminders 235. On dividing it by 2, we are getting 117 and the here we will just note the reminder, and the remainder is 1. Again, we will divide by 2. We will get here 58 and the remainder will also 1. Once again we are dividing it by 2. The answer is 29, but now the reminder is 0, fine.

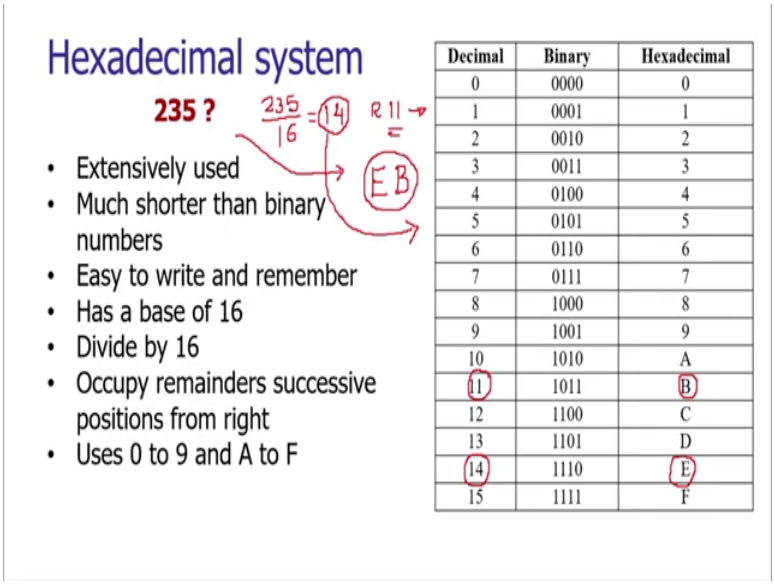
We need to carry out this operation till we get remainder 1 or 0, till we get this value as 1 or 0. We are dividing once again. We will get here 14, and this is 1. Further we divide getting the reminder 0. We will keep on doing this fine. Now, if we want to write the binary representation we will start from the bottom to the top.

We have to first see whether, what is the value we are getting here. If it is 0, it is fine. If it is 1, then we will be using that 1 over here. We will be starting 1, then 1 from then in the remainder column I will be starting from bottom to the top 110101 and 1. This 1 will come over here then 1101011. So, 11001011.

This is the binary number or binary representation fine. The bottom most digit is the MSB. It is the Most Significant Digit. We are calling this as the MSB, Most Significant Digit and the topmost digit is having the least significance. This is LSB Least Significant Digit. In this way we can simply convert the number 235 into its binary representation.

The same 235 the binary number can be converted into the decimal equivalent by using this equation. Well, this is quite popular and it is very simple or general number system that we use, but it is taking lot of bits. If we just look at how many bits it consumed 1, 2, 3, 4, 5, 6, 7, 8. So, 8 bits or 1 byte is required to represent a number with 3 digits that is a 235. It is sometimes it is combustion, it is not economical as per as the memory space and the speed is concerned.

(Refer Slide Time: 08:41)



Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

For this purpose, we are using the hexadecimal system. Hexadecimal means, the base will be of 16 and when we are having the base of 16, we are just trying to find out its hexadecimal representation.

Hexadecimal system is considered to be the efficient one in comparison with the binary representation. In the slide, we can see a table. The table is having the decimal numbers from 0 to 15, their equivalent binary representation it is shown in the second column.

The 0 is represented as 0 0 0, this is a 4 bits representation. Now, in hexa decimal system we are designating, we are representing 0 as 0, 1 as 1, 10 as A, 11 as B because after 9

we have to represent it. Why we have taken only 15 decimal numbers over here? This 15 decimal numbers may be the remainders during the division of the number during the division process, division of what? We are dividing the decimal number with the hexadecimal number that is 16.

When we divide the given number by 16 whatever the remainder number is, we have to represent that number by using certain system. In binary we are having 4 bits representation, but in hexadecimal till 9 we are giving the digits, but after 9 for 10 we are using the alphabets. If the remainder is 10, the alphabet is A. If the remainder is 11, the alphabet would be B, for 12 it is C, for 13 it is D so on and so forth. The hexadecimal system it is easy to write and remember.

It has base of 16; that means, we have to divide the decimal number by 16 and we have to occupy the remainders successive positions from the right. Whatever the reminders would be there, that will be used to represent it, but the representation would be from the right towards the left. To understand this process, let us use the same number that is 235. If I divide the number 235by 16, we will be getting 14 as the quotient and the remainder is 11.

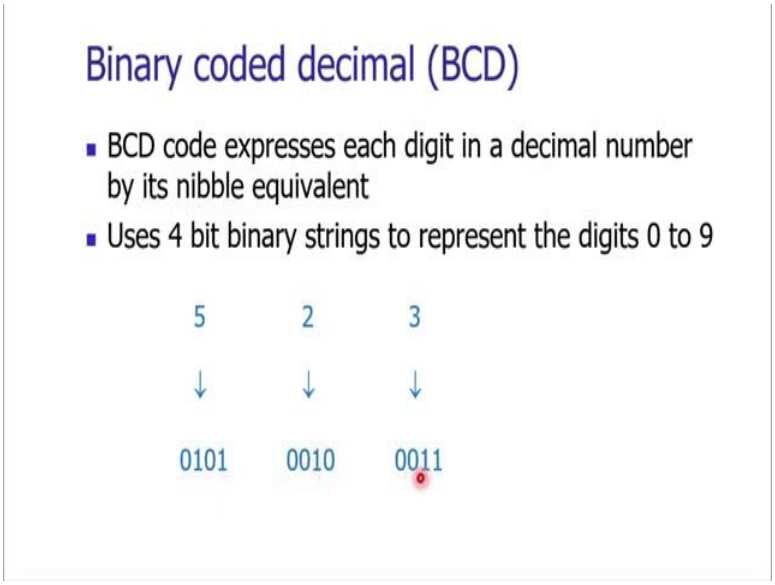
Now, what this procedure suggest you find out the remainder, the remainder is 11. Now, we refer the table and try to find out what is the hexadecimal representation if the remainder is 11. If we just note here for 11 as the remainder, then the hexadecimal representation is B.

Let us take, for remainder of 11 we got the hexadecimal representation of B that we will be writing here B. Then, we will see the quotient, if the quotient is more than 16 again we have to carry out the division operation, but the quotient is less than 16.

For that purpose, again we will be referring the table for the quotient less than 16 what would be the hexadecimal representation. Here for 14, it is showing the representation as E letter. We are using this and we will write here the letter E. We got the next letter as E. Thus. this is the hexadecimal representation of number 235 EB only 2 letters.

A complex 235 or the number 235 can easily be represented by using only 2 letters and this is easy and efficient as well. That is why this system is being used in extensively in the microprocessor programming.

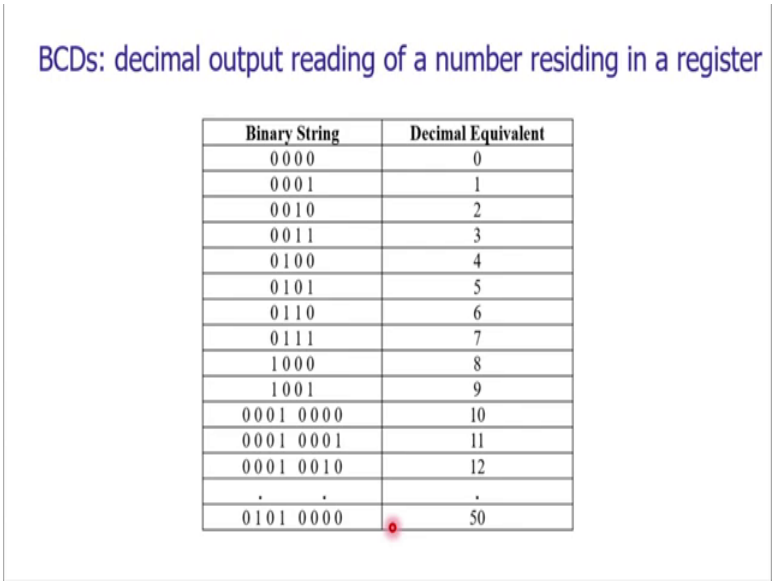
(Refer Slide Time: 15:11)



The next representation system is Binary Coded Decimal (BCD) system. In this system, we are representing, we are expressing each digit of a decimal number by its nibble equivalent. It uses the 4 bit binary strings to represent the digits 0 to 9. Let us take this example. If you are having a decimal number say 523, then how or what would be the binary coded decimal representation.

The first digit that is a 3, it will be represented in its binary format. The binary representation of 3 is 0011, 2 is having the binary representation 0010 and 5 is having the binary representation as 0101. To get this, we are following the same formula same methodology as we have seen in our previous slides. The 523 will be represented as 0101 0010 0011.

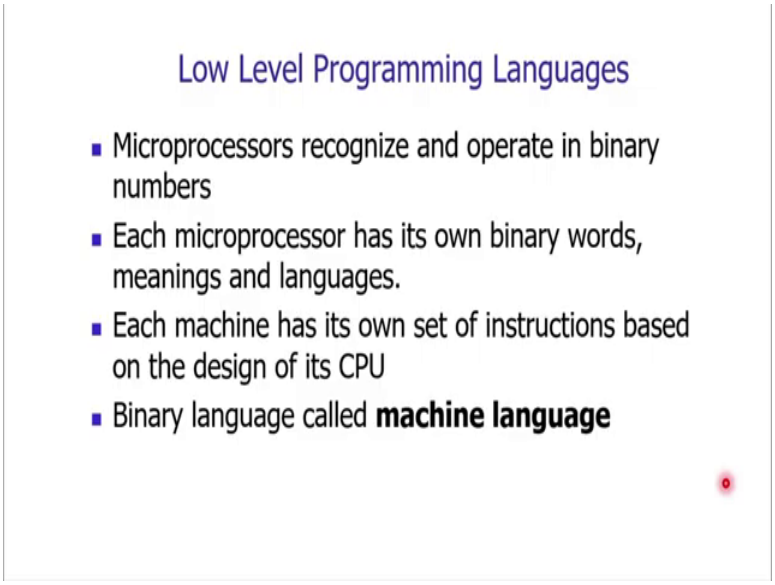
(Refer Slide Time: 16:52)



What is the advantage of this? Here this system would be useful when there is a facility to combine or to use 2 registers of 4 bits capacity. In typically one bit in typical microprocessor with basic microprocessor, the each register is having a capacity of 4 bits. Then to represent a 3-digit decimal number we can use 3 registers or to represent a 2 digits decimal number 2 different registers can be joined together and the BCD can be applied there.

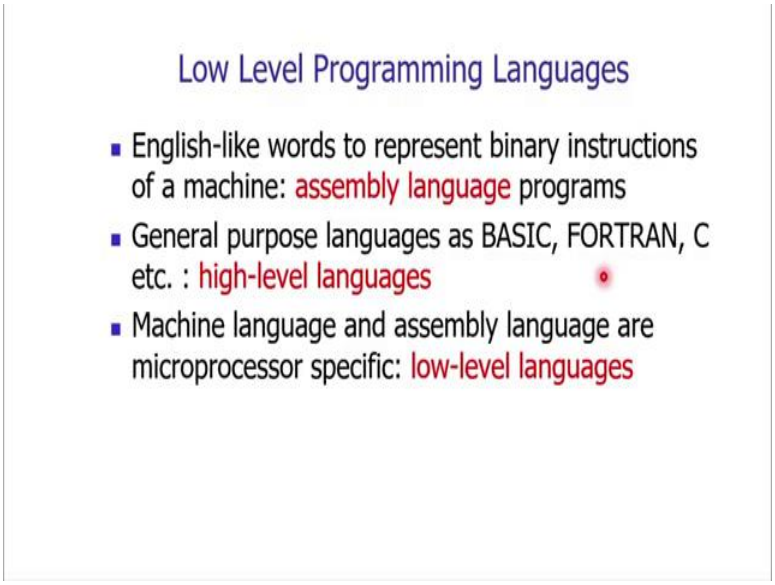
Here we can see from 0 to 9 only 1 register is sufficient, but to have the 2 digits we may need 2 different registers. The first register will store 0001, that is the binary representation of the digit 1 and the, and for 0 there would be another register. These 2 registers can be utilized to represent a decimal number.

(Refer Slide Time: 18:27)



Microprocessors recognize, they understand and they are capable of operation, they are capable of operating the binary numbers. Each microprocessor has its own binary words. These binary words have specific meanings and each microprocessor will have its own language, it will have its own grammar and syntax. In a similar way each microprocessor based machine will have its own set of instructions based on the design of its CPU.

(Refer Slide Time: 19:27)

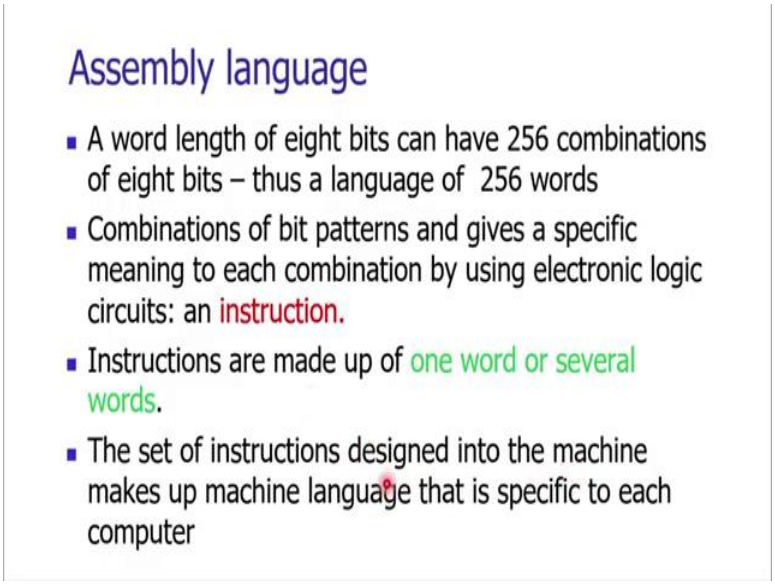


The binary language which is used by the microprocessor is called as the machine language, and when we use English like words to represent the binary instructions, that is

called as the assembly language programs. We also use various programming languages such as BASIC, FORTRAN, C.

Nowadays, we are using Python or Java. All these languages are called as high level languages. But the languages which are very specific to the microprocessor and they are used to assemble the hardware and to communicate with the hardware. These type of languages are called as the low level languages.

(Refer Slide Time: 20:15)



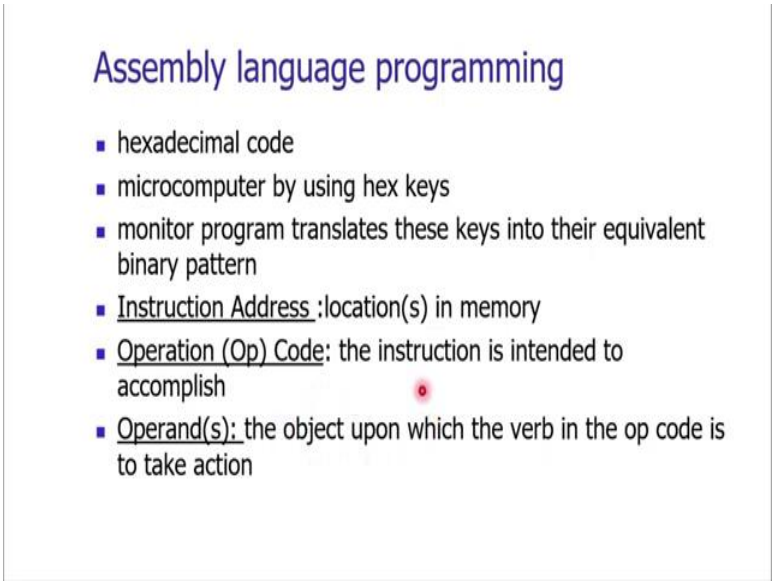
In assembly language, a word length of 8 bits is used. If the word length is 8 bits, so we can have around 256 combinations of 8 bits. Thus, the language will have around 256 words. We can have variety of combinations of this bit patterns and we can attach a specific meaning to each of this combination. And, by using the electronic logic circuits we can carry out variety of operations on that.

Typically, instruction will have a combination of various bit patterns, meaning is attached to that and we can carry out variety of operations on that using the electronic circuitry. The instructions are made up of one word or it may have several words. When we club together or when we have a set of instructions specially designed into the machine, that creates a special language a specific language pertaining to or related to that hardware that microprocessor hardware. That is called as the machine language of that hardware.

(Refer Slide Time: 21:45)

We can define the machine language as the binary medium of communication through a design set of instructions specific to each computer. The machine language would be very specific to each computer. In assembly languages, we are using certain set of words and these words are called as mnemonics. We will see, what is the meaning of mnemonics in our next slides.

(Refer Slide Time: 22:18)



The assembly language basically uses the hexadecimal code and the microcomputers are equipped with the hex keys. The keyboard what we used for operation of computers is

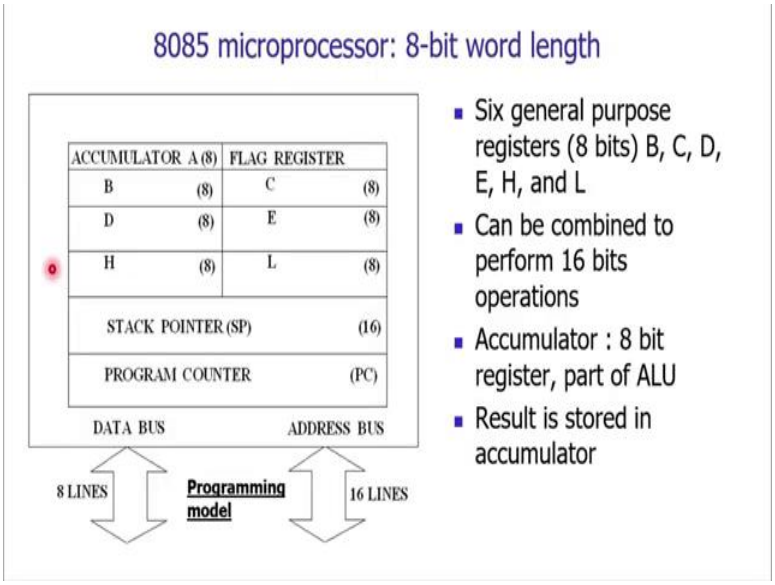
alphanumeric and some specific characters are there. In a similar fashion, we can have the hex keys based keyboard through which we can communicate with the machine. The monitor program translates these keys into their equivalent binary pattern.

When we click on these keys, that the meaning of the keys will be translated into the binary pattern and that binary pattern will be utilized by the microprocessor to take the necessary action. What information we need to carry out the operations that the microprocessor level, we need the instruction address that is a location in the memory of a data on which we have to carry out the operation.

We should know about the operation code, what kind of operation we need to accomplish, what kind of operation we need to carry out on the data. And, the operand the information about the operand, the object upon which the verb the op code is to take the action.

We need to find out the object, the instruction what to be carried out or the task what to be carried out on that object and where the object is lying, where the object is located. The location, the instruction about the process and the object itself: this 3 information are needed to write simple assembly language program.

(Refer Slide Time: 24:16)



To understand the programming let us take simple example. In the slide, we can see a 8085 microprocessor. It is very basic, very simple configuration to understand. My

friends, nowadays we are using very high end microprocessor, the programming is very complex, but for the basic understanding we are using 8085 microprocessor.

The 8085 microprocessor is using 8 bit word length and the architecture of this 8085 microprocessor can be seen in the slide. It has 6 general purpose registers B, C, D, E, H, L. These are the general purpose registers, general purpose memory elements B, C, D, E, H, L. These can be combined to perform 16 bits operation.

As mentioned, in BCD we require more number of bits to be processed. In the situation we can combine this 8 bits registers and we can carry out the processing on the higher bits representation numbers. This 8085 microprocessor is communicating with the outside world through data bus and the address bus.

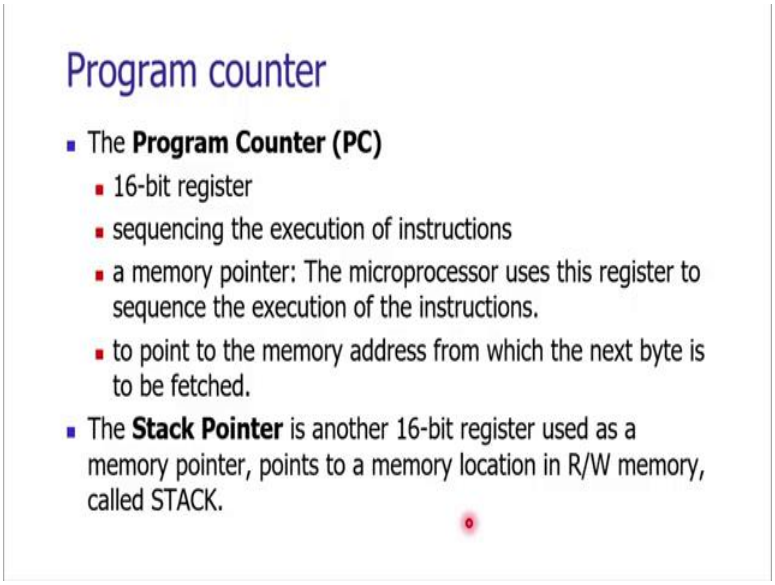
The meaning of data bus and address bus we have seen in our previous lecture. Well, the general purpose registers are programmable, it means that a programmer can utilize them to load or copy data from these registers and to do this we have to instruct we have to carry out the instructions.

For example, MOV i.e. move command, MOV D E means I want to move the data which is located at register D to E. Meaning is that we can easily transfer the data which is available at D to E. That means the registers are programmable. We can say these registers are nothing, but memory locations, but we are using them in the ALU that is Arithmetic Logic Unit in the CPU, that is why they are called as the registers. Accumulator is also a register. It is of 8 bits. This is basically used to store the result of the arithmetic operation.

If I say add H and L, the data of H and L will be added together. The results of this addition will be stored in the accumulator A, and from this location it will be transferred to its destination in the memory. In addition to the general purpose registers and the accumulator register A, we are having a flag register.

The flag register gives the status of the result. It has various flip flops, and by setting and resetting this flip flops we can give the status of the result. We can define the signature of the result, whether it is positive or the negative by using this flag register. There are various flags being used such as sign, parity, carry so on and so forth.

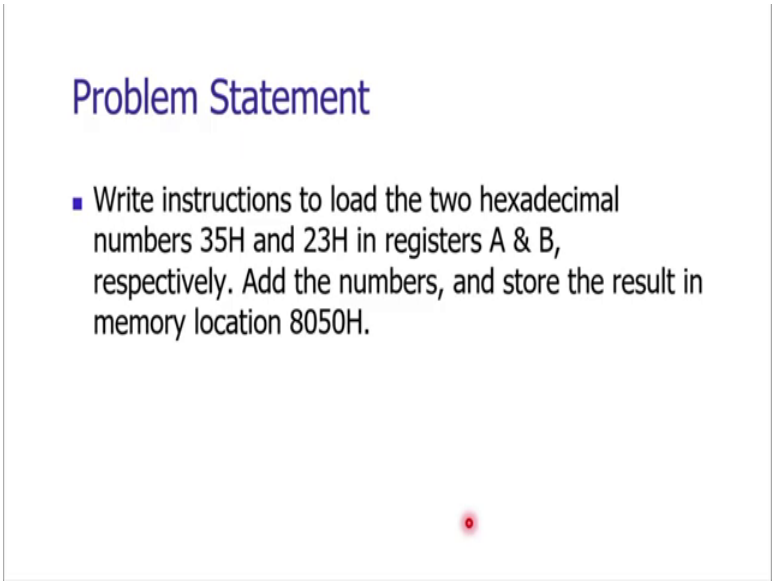
(Refer Slide Time: 28:35)



Now, what is the significance of program counter? Program counter is a 16 bit register and it executes the sequence of instruction. Whatever the instructions are written in the program, that would be executed as per the sequence given by the program counter. The name itself suggest, it is counting the steps and it is suggesting or it is instructing the processor to go for the next step to execute the instruction. It is also called as the memory pointer.

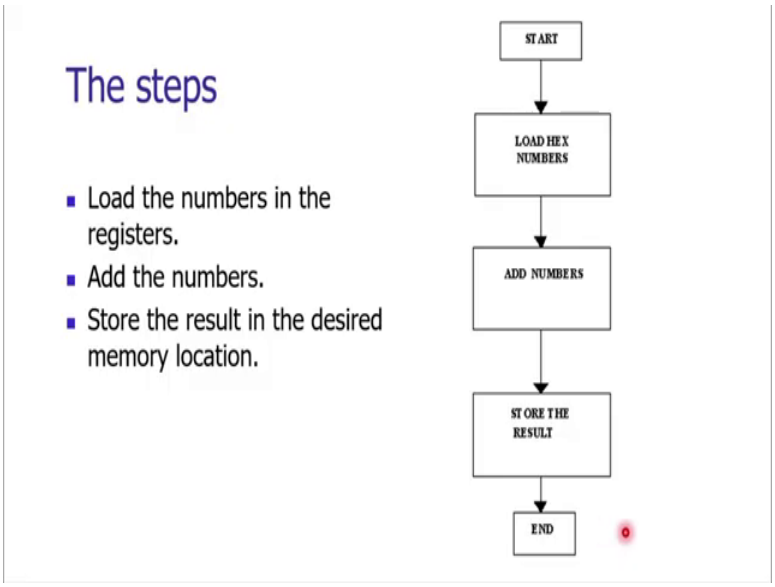
The microprocessor uses these register to sequence the execution of the instruction. It is used to point out the memory address. From that location we need to fetch the next byte of the information. The stack pointer points to a memory location in the read write memory that is called as the stack. The beginning of stack is defined by loading 16 bit address in the stack pointer or the register.

(Refer Slide Time: 29:53)



Now, let us write a simple program and the objective of this program is to load 2 hexadecimal numbers 35H and 23H at registers A and B. And then, we need to add these numbers and we have to store the result of this addition at a location 8050H.

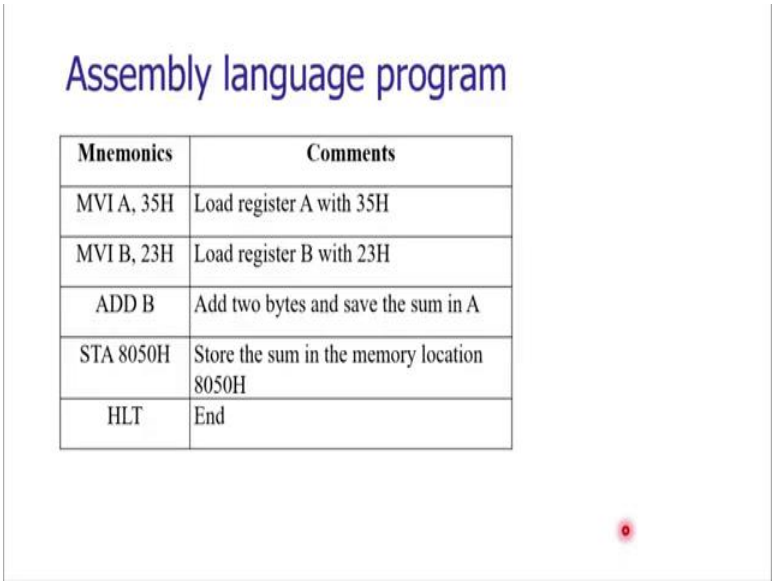
(Refer Slide Time: 30:18)



Let us see how to write a simple program. The various steps can be seen in the slide. First of all loading of the numbers to the respective registers, then addition of the numbers and then storing the result at the desired memory location. For that purpose we

can develop a simple flowchart like start, loading of hex numbers, then addition and storage, and storage of the result. After that the program will be ended.

(Refer Slide Time: 30:53)

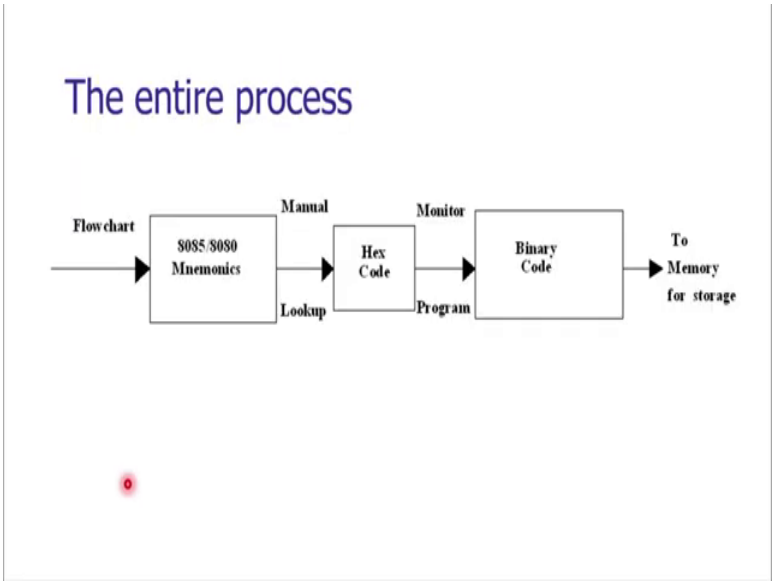


The corresponding assembly language program for this operation can be seen in the slide. These are the mnemonics. Mnemonics as we have seen the words special words of the 8085, which are designating the specific operations.

Here, the loading operation is represented by a word MVI; MVI A means, we have to load the register A with the value what is the value is 35H. Similarly, we have to load the register B with the value 23H. Then, the next mnemonics is ADD B.

We are adding B to the A directly no need to write the letter A over here. Then we have to store to store STA, the sum will be stored at 8050H. In this case whatever the sum would be there of this operation, that sum will be stored at the location of A only. The result will be stored at accumulator A, and that result will be transferred to the location 8050H. At the end, we are halting it we are saying the program is completed. For that the mnemonics is HLT halt.

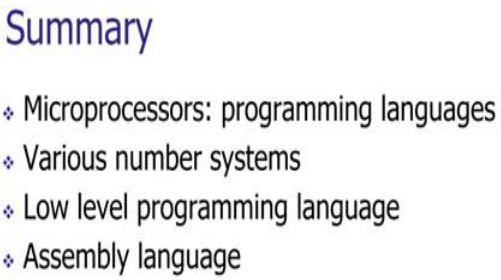
(Refer Slide Time: 32:36)



This entire process is a very simple. It is having set of mnemonics as per the given microprocessor in this case we are having 8085. We are looking up hex codes, then we are converting that hex codes by using the monitor program into binary code. The binary code will be stored in the memory of the microprocessor. In a similar way, we can write a simple program.

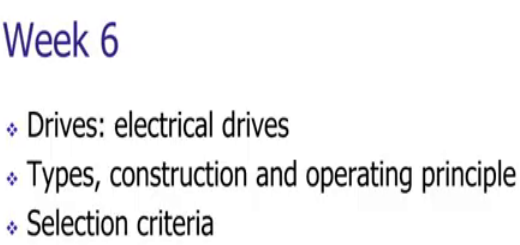
This is just an introduction to the programming of microprocessor. If you are interested you refer the book by Ramesh S Gaonkar. The name of the book is “Microprocessor Architecture Programming and Applications with the 8085”. Finally, let us summarize the lecture 5 of week 5.

(Refer Slide Time: 33:43)



In this lecture, we have seen various programming languages which are used in microprocessor technology. We saw various number systems being used in the microprocessor programming. The meaning of low level programming language and assembly language have been studied. At last we carried out a simple case study on programming of a typical microprocessor.

(Refer Slide Time: 34:14)



So, with this we end the week 5 of our course. In week 6 of this course, we will be studying various drives particularly the electrical drives which are used in the automated

systems, types of these drives, their constructional details and operating principle. We will also see how to select these electrical drives, what are the criteria of selection of these drives till then.

Thank you goodbye.