

Computer Organization and Architecture: A Pedagogical Aspect

Prof.Jatindra Kr.Deka

Dr.Santosh Biswas

Dr.Arnab Sarkar

Department of Computer Science and Engineering

Indian Institute of Technology, Guwahati

Fundamentals of Digital Computer

Lecture - 04

Basic Elements of Processor

(Refer Slide Time: 00:47)

Module: Fundamental of Digital Computer

- Unit-4: Basic Elements of Processor
- Unit Objectives:
 - Objective-1: Illustrate the components of a processor (Knowledge)
 - Objective-2: Describe the different categories of registers (Comprehension)
 - Objective-3: Explain the tasks performed in an instruction cycle (Analysis)

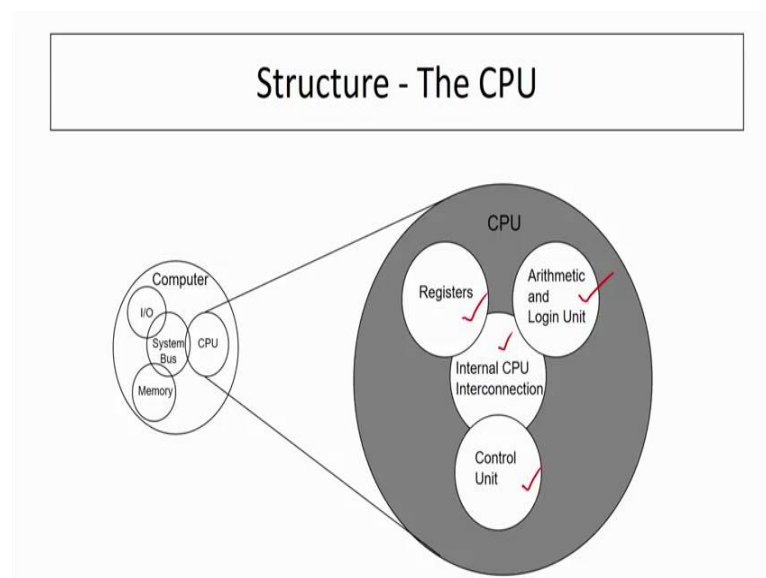
Hello everybody welcome back to the online course on computer organization and architecture. We are in the module fundamentals of digital computer. And now we are in unit 4, the unit 4 is related to basic elements of processor. So, as usual now first we are going to talk about the objective of this particular module. We have identified three objectives for this particular unit on basic elements of processor.

The objective -1 illustrate the components of a processor. So, we are going to see what are the elements we have in a processor and the level is your knowledge level. So, in this particular unit we are going to discuss everything in knowledge level only. So, we are going to just give idea about it, but in subsequent module these things can be covered in higher level maybe in the design level and synthesis level.

Objective-2 describe the different categories of registers. This is in comprehension level, what is register we are going to see and what are the different categories that we have we are going to look into it.

Objective-3 explain the tasks perform in an instruction cycle. So, already in our earlier unit, I have talked about the instruction cycle, how to execute an instruction. Now, we are going to see what are the different tasks or sub tasks that we are going to perform in an instruction cycle. So, this is in analysis level that means once you see an example of an instruction then at least you will be able to analyze what are the tasks that will be performed by this particular instruction, and finally, what is the effect that we are going to have after executing this particular instruction.

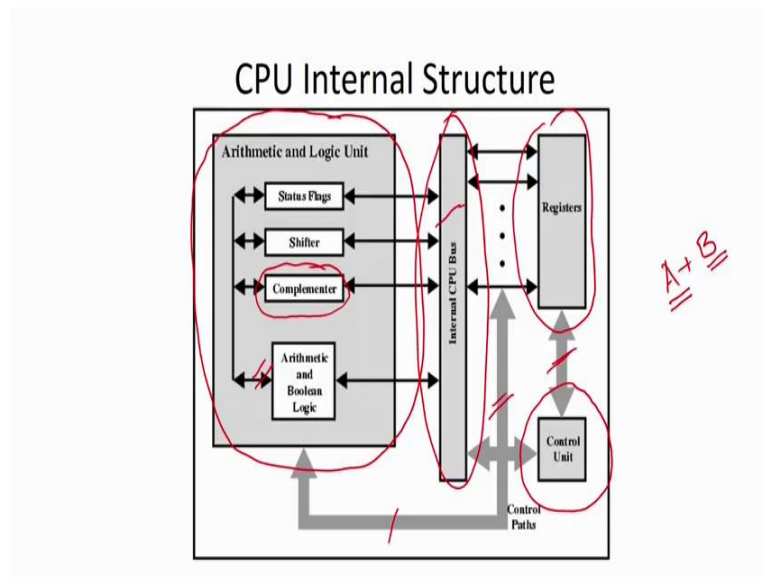
(Refer Slide Time: 02:10)



So, we have seen that top level view of your computer structure. So, in this particular case, you are saying that we are having four basic element CPU, I/O, memory and they are connected with system bus. So, this is the computer that we can see components of a computer. Now, in this particular unit, we are going to talk about the processor CPU central processor unit. Now, if you look at it then you can bubble up this particular bubble of CPU and we are going to see that basically it is having three major component one is your arithmetic and logic unit, then we are having the registers which are nothing but the storage unit and along with the control unit. Controlling unit is having the responsibility to synchronize the operation inside the processor and all those components are connected through this internal CPU interconnection. Because

sometimes or in many a time you have to transfer information from one component to the other component like the transferring information from registers to the ALU or maybe the result from ALU to the registers. So, for that we need to have the interconnection, so we are having the internal CPU interconnection.

(Refer Slide Time: 03:21)



Now, this is the internal structure, it is very generic one, in most of the cases you are going to see this is the internal structure of a processor, but it may varies from processor to processor, but in general we are going to have those components. So, just see what are basic components that we are having, we are talking about the registers, these registers are nothing but the storage element. So, here we can store our information. Another component we are having called ALU - arithmetic and logic unit. So, this is the main processing element inside the processor, here we can perform some operation like in ALU inside that we are having a component called which have a have arithmetic and logic operation.

So, some of the arithmetic operation you can perform like addition, subtraction, multiplication, division etcetera along with that we are having a provision to perform some operation Boolean operation like AND, OR, exclusive OR etcetera. So, these are the basic operation that we can perform along with that sometimes we have some other functionalities also one is known as your complementer basically it is going to complement a given number. Another one we are having a shifter, already we have talked about the shift registers that means, sometimes we have to shift the contents of a register either it is your left shift or right shift. So, for that also

we are having this particular shifter. And along with that we are having status flag will going to see or going to elaborate what is those particular status flag, but already we have discussed some of the issues related to those particular status flag when we talked about the arithmetic operation inside the ALU. So, we are going to see how what is the effect of those particular component.

Along with that now say we are having registers and arithmetic and logic unit; now we are having an interconnection of this particular component. So, here we are having an interconnection network which is basically bus we will see what is bus now. So, basically it is a connecting wire. So, from this registers we can place the information in this particular internal connecting wire bus from that it will go to the ALU. ALU will perform the operation and result will be transferred to again through bus to the registers if we want to store the result.

Now, we have to now perform this operation in a systematic way. Just say that if I want to add two numbers $A + B$, then what we are going to do first we will transfer A from this register to the input of the ALU then we will transfer B from the register to the ALU then I am going to perform the addition operation. Whatever result we are getting after performing the addition operation, we are going to transfer it to the registers. So, for that we need appropriate control signals. So, this control unit is having the responsibility to generate these particular control signals. So, these are the control lines connected to different component.

Now, what is the control signal? So, first I am going to say the transfer the A to ALU. Now, we are going to select the register A inside that ALU we are having several registers in register bank. So, control unit will generate appropriate signal to select this particular register A, then we will transfer the information to the input of this particular ALU one of the input of this particular ALU. Then control unit is going to generate the next signal, so that it is going to select the register B. And after selecting the register B we will transfer the information to the ALU again. Now, after getting both the information A and B inside ALU now what is the responsibility of the control unit, to generate the signal to trigger the addition operation inside this particular ALU, because we are having several operations inside ALU and already we have discussed it already explained how we are going to select a particular processing element.

So, control unit is going to generate the signals, so that it is going to trigger the particular addition module of this ALU and finally, it is going to get us the result. So, once we are getting the result, then what we are going to do we are going to transfer it to some of the registers. So,

again control unit is going to generate the signals. So, the responsibility of the control unit to generate the appropriate control signal at the appropriate point of time, and these control signals are given through this particular control part or control bus. And through this data bus we are transferring the data from register to ALU or from ALU to registers. Ok so, these are the internal structure of a processor; in most of the processor we are having those components, but the placement of the components may vary from design to design.

(Refer Slide Time: 08:11)

Registers

- CPU must have some working space (temporary storage) -- Called registers
- Number and function vary between processor designs
- One of the major design decisions
- Top level of memory hierarchy

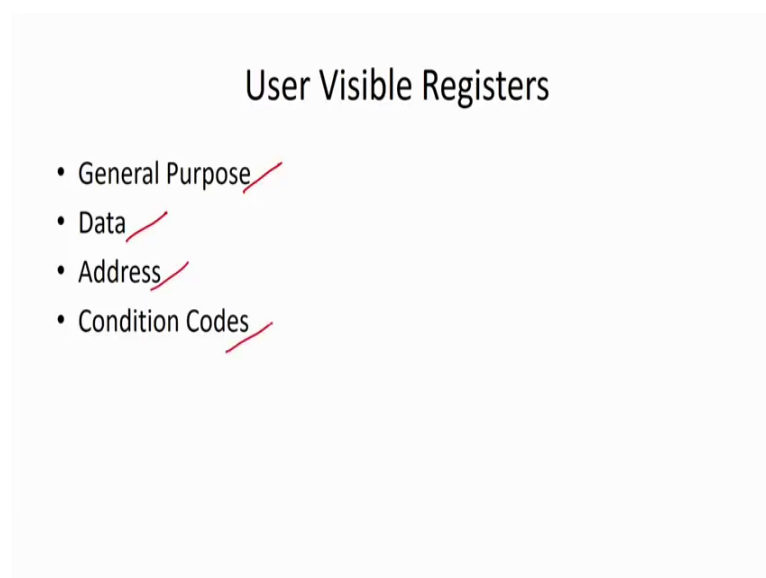
Now what is the first component that we are going to talk about it is the register. What is registers? These are nothing but some storage space. So, CPU must have some temporary working space and which are known as your registers. So, first we are going to take the information and put it into this particular registers. So, after that once we have the information inside the processor that now processor can carry out the work according to our requirement. So, the number of functions vary between processor designs. So, the what are the number of process register that we may have inside a processor, what are the functions of those particular register, it depends on design of the processor. So, it varies from design to design. So, this is one of the major decision in processor design. And we have to see how many registers we are going to keep and what are the purpose of this particular registers and secondly, we can consider this as a top level of memory hierarchy.

What is the memory? It is a storage element, so I can store information. So, this is top level we are saying that this is inside the processor itself processor is going to work with the information

available on these particular registers. Now, where from we are going to get that information into those particular registers if you see the working principle or Von Neumann stored program principle so that means, you must have storage element, which is the memory module, so that will be connected to the processor. So, this is the next level of storage in the memory hierarchy.

So, in a memory hierarchy, top level we are having the register where we can store our information, but where from we are going to get information to that particular memory or registers it is from the main memory. So, main memory is the next level. So, we can have a hierarchy of memory. And in this particular course, we are going to see what are the different level of memory that we have in a computer. So, just I am mentioning that it is a top level of memory hierarchy. So, registers are nothing but a temporary storage unit which are placed inside the processor itself and processor is going to work with the information available in the register.

(Refer Slide Time: 10:21)



So, what are those registers? So, registers can be categorized into different part, one is known as your user visible registers. Why we are saying this is user visible, because these are the architectural issues and we know that there are some register inside the processor, and while writing our program we can use those particular registers, so that's why we are saying this is the user visible registers. And these are these can be used by a programmer while writing his program. Maybe when I am going to talk about the writing a program we will see in what level basically we are mentioning about it. So, these are the registers and user or a programmer can

use. Again in this user level registers can be categorized into four different categories one is your general purpose, data registers, address register and condition codes.

So, general purpose register, it can be used for any purposes we can keep or we can store an information or any kind of information inside the general purpose registers, but if we are going to say it is a data register then what will happen we can store only data that we are going to work on. So, data register can keep data only. When we talk about the address register basically this is going to keep the information about the address. Now, what is an address? So, we already have mentioned that our computer works on Von Neumann stored program principle. So, for that we are having a storage unit which is known as your memory unit; now in the memory we are going to store our information.

Now, how to get this thing? So, for every memory location we must have to assign an unique address likewise say if I want to reach you at least I must know your address. So, if you are a student and you are staying in a hostel that is I must know the room number that you are staying, then I have to know in which hostel you are staying; of course, from outside world, I must have to know in which institute you are there. So, like that all those components is going to tell me your location and I can go to your location. Like that when we are going to take going to take an information from a memory we must know this particular address.

So, we are going to give some address to each and every location, and this address register can keep only the addresses of this memory location. Secondly, when we are talking about addresses several I/O devices will be connected to the computer. So, to identify each and every I/O devices, we must have to assign some addresses. So, those addresses can also be kept in our address register. And we are having one more registers called condition codes. So, we will elaborate it what does it means and what information we are going to store in condition code.

(Refer Slide Time: 13:04)

How Many GP Registers?

- Between 8 - 32
- Fewer = more memory references
- More does not reduce memory references and takes up processor space
- Large enough to hold full address
- Large enough to hold full word
- Often possible to combine two data registers
 - C programming
 - `double int a;`
 - `long int a;`

So now, one of the major decision while going to design a processor is the number of registers, how many general purpose register we are going to keep. This is a major design decision because the performance of a processor depends on this particular number. If we are having a pure registers then what will happen, there will be lot of memory references, we are going to keep one going to bring one information from memory working with that particular information. But if we are having a fewer number of registers in all we are keeping some information in all the registers then what will happen when we are going to bring another information to the processor, we cannot keep in any of the register because all are occupied. Then we have to transfer some of the information from register to the memory to make a vacant space and in that particular vacant space, we can keep this particular new information.

So, if having a very less number of registers then what will happen that memory reference will be more; that means, lot of information has to be taken from memory and intermediately we have to store into the memory. But again it doesn't mean that if we are going to increase the register they are going to put more number of register again it is not going to reduce the memory references drastically, it will reduce the memory references up to a certain extent. So, it is not like that we are going to work with a very less number of register or very big number of registers. In typical way it is observed that in most of the processor we are having a register somewhere between 8 to 32, 32 is also in some higher side. So, it is somewhere near to 8 only or in general I can say that register number of register varies with an 8 to 32

Now, when we are talking about the registers, when we are going to talk about this storage element, temporary storage element instead of processor, we must have to store some information, and those information should be stored properly. So, for that we have to see what will be the size of those particular registers. If we are talking about the address register then at least it should be able to store the full address of the memory location or full address of the I/O devices. So, registers address registers has to be long enough or large enough to keep the complete information of a memory location or a device. And what will be the size of a data register, it should be long enough to or large enough to hold a complete word.

So, if we are going to talk about a processor say we are going to see that it is a 16 bit processor; that means, we can work with 16 bit at a time so that means, the size of the register should be at least 16 bit. When we talk about a 32 bit processor then we are going to work with a 32 bit data, so that size of the register should be your 32 bits. Also it is possible to combine two data register together to work with more information. So, like that in C programming, if you are accustomed with the C languages you will find that sometimes you are going to use that double integer. So, integer is a data type we can have a double integers. So, in case of doubling integers, we are going to increase our length. So, in some cases depending on the processor, it may happen that sometimes we have to combine two register together to keep a double integer inside the processor. So, this provision is also you have to provide while designing the processor.

(Refer Slide Time: 16:33)

Control & Status Registers

- ✓ Program Counter
- ✓ Instruction Register
- ✓ Memory Address Register
- ✓ Memory Buffer Register

So, these are the general purpose register, where we are going to store information data and addresses and these are visible to the programmers. So, while we are going to write a program, we may use those particular register to store our information, but there are some registers or storage elements inside the processor which are known as a control register or status register, these registers are not visible to the programmer. Directly we cannot use those particular registers, but for the better operation of the computer or processor we have to have those particular registers. So, programmer cannot use those register, but to control our operation while designing we are going to use those particular registers.

So, some of those registers is one is your program counter, second one is your instruction register, third one is your memory address register, fourth one is your memory buffer register. So, I am citing four registers over here, and these are bare essential register inside the processor to control the operation inside the processor. So, in subsequent lectures, we are going to discuss in details about the use of those particular registers thus here I am introducing that these are the four status registers or control register we have inside our processor.

(Refer Slide Time: 17:57)

Condition Code Registers

- Sets of individual bits
 - e.g. result of last operation was zero
- Can be read (implicitly) by programs
 - e.g. Jump if zero
- Can not (usually) be set by programs
- Needs for conditional instructions

Handwritten notes:

if (a == 0)
 ↓
 3
 Set: 1
 Reset: 0

Sign bit

0: positive
 1: negative

There is another one register called condition code register, this is a register, but here we are going to work with individual bit. So, say that if we are working with an 8 bit processor then what will happen the size of the registers are 8 bit. So, we can consider an 8 bit register where we can store 8 different information. Ok so, in case of your condition code register, we can work with individual bit ok either we can set or reset those particular individual bit or we can

use those particular individual bit. So, this is one of the issues. And now what we can store or what we are going to store over here. So, this bit will be set or reset, already I have mentioned that set means we are storing information 1, and reset means we are resetting it to 0.

So, these are basically made of flip flop. So, flip flop can be set or reset. And when we are going to set and reset, it depends on some operation inside the processor. So, generally it cannot be set by the programmer, but some of the bits we may have which can be set by the programmer in general we can say that it is not set by the programmer. But those bit will be set by the operation of a processor, we are performing some operation and on depending on the result of those operation, we are going to either set or reset those particular bits.

And when we are going to use those particular condition bits to make some decision. So, when we are going to design some conditional instruction say in high level language generally you write something like no, if $a = 0$ then do something else do some other thing. So, now, here we are comparing. So, depending on this comparison if it is 0 then we are going to perform this code segment; and if it is false then we are going to perform the other code segment.

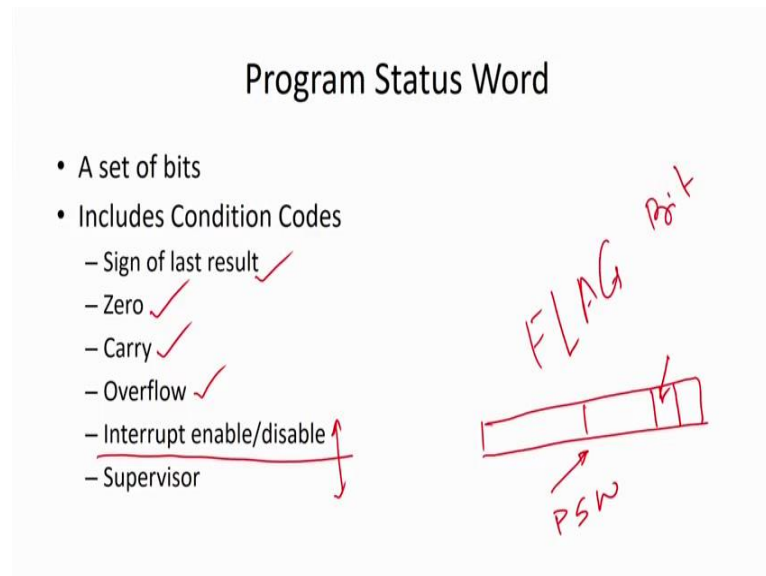
To design such type of instruction in programming languages - conditional instruction, we must have some instruction inside the processor in the instruction set of the processor we have to provide some conditional instruction. So, while going to design those particular conditional instruction then we are going to use those particular condition code bits. So, one of the code bits I can give here like that sign bit. So, when we discussed about the representation of integers or representation of your real number we have seen that we can work with positive number and negative numbers. So, for that we are having a sign bit.

So, when we perform an operation say like that $A + B$ then depending on the result that sign bit will be positive or negative. So, depending on that say if I am going to have a sign bit over here that sign bit will be set or reset depending on the result of my ALU operation. So, if it is a positive number, if the result is positive, then we say that 0 will be stored. So, 0 is going to indicate the positive; and if the result is negative then we are going to store 1, so it is going to indicate that the result coming out from the ALU is a negative 1. So, this is one of the condition code.

Like that another condition code I can say carry flag; already I have said that we are having a carry out of an adder circuit similarly we can have overflow bit condition code because we have seen that the range of numbers is restricted by the design of the processor, we cannot go

from $-\infty$ to $+\infty$. So, depending on the early result sometimes it goes beyond those particular events and in that case we are going to say it is an overflow. How we are going to identify the overflow? So, in that particular case what we are going to do we are going to keep this information in one of the condition flag, and to take the decision we are going to see what is stored in this particular position.

(Refer Slide Time: 22:23)



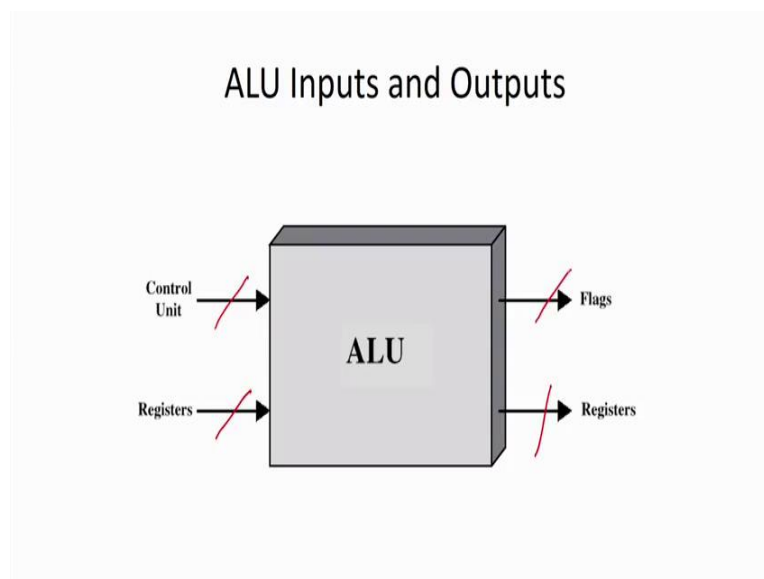
So, this is a condition codes. And condition codes are basically known as your flags also flag bits, we can say these are flag bits or condition bits. And already I said that we having a condition code register, and what about value we have in this particular condition code register this is also known as my programs status word PSW - program status word. Basically it says that what is the status of my program at that particular point because at some point of time my result may be your negative or at some other point whatever value I have getting it may be an overflow condition, sometimes my result may be 0. So, like that we can have flag bits inside this particular PSW - programs status word or condition code one bit for your sign bit, one bit to indicate the zero bit whether the result of the processor is zero or not. One bit to indicate whether carry out is generated or not, another bit may be saying about what is an overflow occurs in the computation or not.

So, these are the bits which will be set or reset depending on my result of the ALU. Along with that we may have some other bits also condition code which may be set by your programmer while writing his program. So, one of these things is maybe it is talking about the interrupt

enable and disable. So, just to give an idea I am just giving you one example the interrupt means like that we can interrupt or we can disturb the working of the processors. A processor is executing a particular program, but at some point of time I want to give some input from the keyboard or maybe I want to control some other devices to the computer; in that particular case that program will be interrupted and the processor service will be given to that interrupted request. So, that interrupt whether we are always going to allow it or for some cases we are going to disallow it. So, again to have these things we may have one bit which is going to indicate about that when interrupt is allowed or it is disabled.

So, basically sometimes it may happen that when we are performing some high priority job we feel that or we expect that no one should disturb me at the particular point; so in that particular case what will happen we can be disable this particular interrupt after completion of the task we can enable it. So, we are having some more flag bit which can be set or reset by the programmer.

(Refer Slide Time: 25:02)



So, these are the registers. So, what resistance we are getting basically storage element. One we are having the general purpose registers, secondly, we may have some dedicated register to store only data and store only address. We are having some special purpose register here I have mentioned four registers called program counter, instruction register, memory address register and memory buffer register. And along with that we are having a special register call condition

code register, where we are going to store the status of the processor. So, these are the some storage unit or element we have inside the processor and these are known as our registers.

And what is the other component inside the processor, this is the ALU - arithmetic and logic unit. So, ALU is going to perform the basic operation inside the processor. So, it is nothing but a combination of processing element. So, in a nutshell we can simply say that it is like that inside that ALU we are not going to look into it will just see it is a black box. So, what we are having we are going to give input from the registers, it will perform the operation depending on our request. So, control unit some signals will come from the control unit if we identify which operation we are going to perform. After performing the operation it is going to give us the result whatever result we are getting we can again store it into the registers. And so, I have mentioned that depending on the result it is going to set some of the flag bits like that after performing some operation if overflow occurs then what will happen that overflow bit will be set by those particular signals.

So, some signals will also come from the ALU and according to the status of those particular signal we are going to set or reset those particular flag bit. So, in a higher level, you can see that ALU in this particular way we are having input, we are having output, we are having some control signal and its input and some of the data or bits flag bits as an output. So, this is another component inside the processor which is ALU, and this is the basic processing element inside the processor.

(Refer Slide Time: 27:10)

Function of Control Unit

- For each operation a unique code is provided
– e.g. ADD, MOVE
- A hardware segment accepts the code and issues the control signals



Now another component is your control unit. So, what is the function unit or sorry what is a function of this particular control unit? So, what will happen inside a processor we are having several operation. And we say this is the instruction of the particular processor and we are having an instruction set. So, for every instruction, we are going to provide a unique code like that here we are talking about that I can perform an add operation. So, for that particular add operation, we are going to give a unique code to it, because we are going to work with the numbers or which can work with the binary digit only zeros and ones. So, one unique code will be given say I can say that this is 001 is going to give for add operation.

So, similarly if I am going to perform a subtract operation then I am going to give another unique code. So, if I want to move the information from say register to ALU then what will happen we must have an instruction, we must have an operation for that and this operation is also going to get an unique code. So, I am going to say that 001 is the code for that if we are going to work with 4 bit. So, this is the way we can say that we are going to give an unique code to each and every operation. And when we get this particular unique code, we know what operation we need to perform that control unit is going to generate the appropriate signal to carry out those particular operations or carry out those particular task.

So, when we talk about that ADD instruction going to perform the add operation, it may have several sub steps. So, in every sub step that control unit is going to generate the appropriate signal and that will go to the appropriate component inside the processor to enable that particular component. So, this is the function of our control unit.