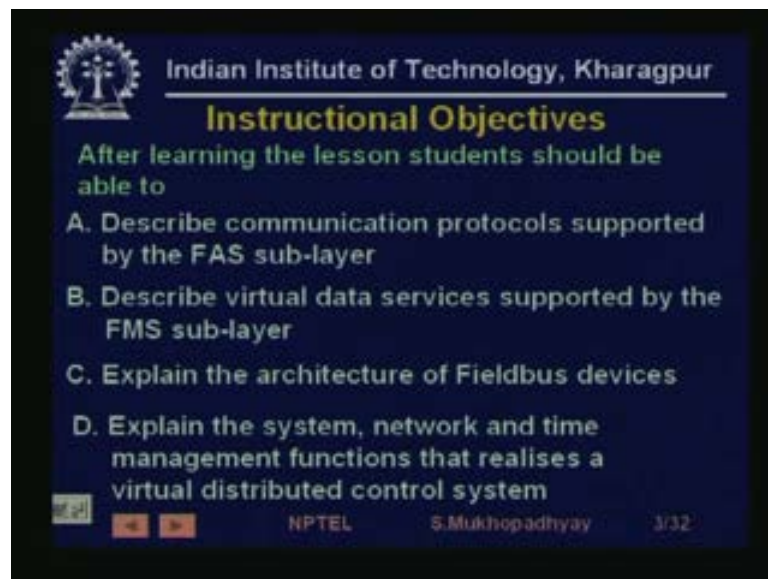


Industrial Automation and Control
Prof. S. Mukhopadhyay
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 38
The Field bus Network – II

Welcome to lesson 38 of the course on Industrial Automation and Control under the NPTEL program. In the last lesson we had been talking about the field bus network and we had seen the basic nature of protocol, we had seen the bottom layers, the connectivity, the data link layer, where basic cyclic and acyclic communication is realized. In this lesson we are going ahead and we shall take a look at the two upper layers of the field bus network, namely the field bus application layer and the user layer.

(Refer Slide Time: 01:38)



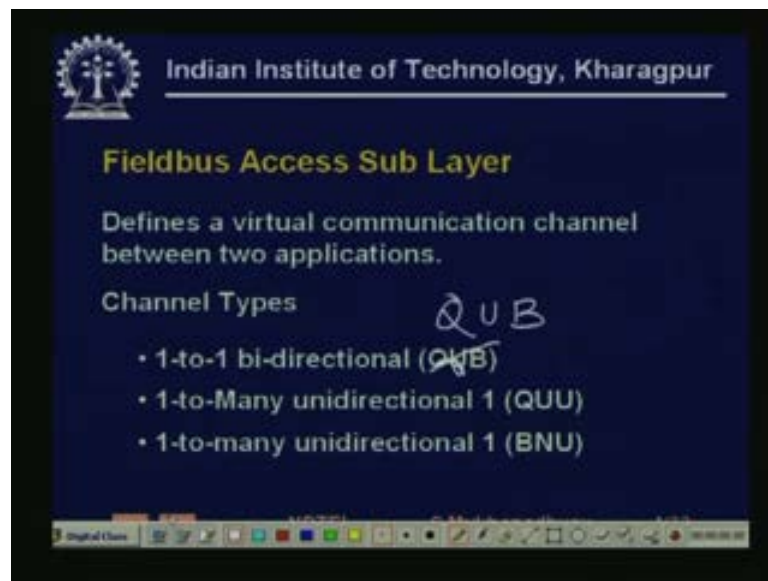
The slide is a dark blue presentation slide with white and yellow text. It features the IIT Kharagpur logo in the top left corner. The text is centered and reads: 'Indian Institute of Technology, Kharagpur' followed by 'Instructional Objectives' in yellow. Below this, it says 'After learning the lesson students should be able to' and lists four objectives: A. Describe communication protocols supported by the FAS sub-layer; B. Describe virtual data services supported by the FMS sub-layer; C. Explain the architecture of Fieldbus devices; D. Explain the system, network and time management functions that realises a virtual distributed control system. At the bottom, there are navigation icons and the text 'NPTEL S. Mukhopadhyay 3/32'.

So, looking at the instruction objectives, describe communication protocol supported by the field bus access sub layer, as we have mention before, the field bus application layer is actually consisting of two sub layers, one is the field bus message specification or field bus message sub layer and the other is typically referred to as FMS and the other is the Field bus Access Sub layer or FAS. So, here we will describe communication protocol supported by basic features of field bus access sub layer, then the virtual data services which are support by the field bus message sub layer.

So, we will basically take a look at access sub layer functionality and FMS functionality, so that you understand the basic purpose of it. Then, we will go over to the user layer and after this you should be able to understand the basic architecture of field bus device, how user computation is done. And finally we will take a look at the various system network and time management functions, that are essential to realize a virtual distributed control system.

So, going before we start ahead let me mention that this networks are extremely complicated and it is very difficult, apart from my own limitations, it is very difficult to give a detail idea of the field bus network in one or two lessons. So, the basic objective of the lesson is that, you form an overall idea of how a network works and it is in particular how a network for control will work. And what kind of facilities are available for programming and in terms of programming and functionality, this is what we aiming at.

(Refer Slide Time: 03:59)



So, with this introduction let us going into the lesson coming on to field bus access sub layer, as we have mentioned that the network protocols organized hierarchically, which means that each layer provides a certain kind of service and by providing that service, it abstracts certain details of the lower layers from the user at of given layer. So, essentially the view at a particular layer need not bothered about certain details which the lower

layers take care of. Like for example, the data link layer takes care of error control coding, so is that is one bit error it will be corrected.

So, how that is done when you are working at the field bus application layer, you actually need not be bothered. So, this similarly when you are at the user layer, actually you need not be bothered as we shall see that how exactly the communication is realized, you have a virtual communication channel. So, you just assume that one particular user process simply talks to another user process, which may be anywhere on the network, how exactly that dialogue is achieved is actually a service which is provided by the lower layers, so therefore, the upper layers did not bother about it.

So, this basically takes a lot of hassle away from an application program, so this is the basic idea and the field bus access sub layer actually you know defines a virtual communication channel. So, it gives you a service by which you can take a message and you can either send it to a particular node in the network or you can send it to a number of nodes or you can actually broadcast it.

So, there are this how exactly it is done is handled by the field bus access sub layer and the lower layers. And therefore, the upper layers need not be concerned about it, so it basically establishes a virtual communication channel between two applications. It gives you the channel types that is this a communication channel can be various types. So, it can give a various kind of communication service for example, it can give you as we shall see very soon that it can give you cyclic or acyclic communications service, it can give you confirmed or unconfirmed communications service.

That is when you have confirmed service then you send a data and the receiver after he receives it gives you back an acknowledgement. So, that is kinds of confirmed service, on the other hand you may have unconfirmed service, in which the user process does not expect any acknowledgement. So, in short the type of the channel and the type of communication it offers a certain number of services to the upper layer in terms of that.

So, for example, it can establish a 1 to 1 bidirectional channel or a 1 to many unidirectional channel of type 1 and or 1 to many unidirectional channel of type 2 we should see these things in the greater details. So, I am not explaining what this by the way should be QUB not this should be I would write it like this QUB not OUB.

(Refer Slide Time: 08:11)



So, moving on, so let us look at the one to one bidirectional or QUB communication, here Q means Queued, let me write it QUB is Queued User triggered and Bidirectional. So, this is first of all this is a connection oriented protocol, what do you mean by a connection oriented protocol, what we mean by a connection oriented protocol is that, it is a one to one, that is one particular node when sends a message the messages meant for particular node.

So, it is connection oriented, it is acyclic typically you know these are used for operator commands, given when a particular let us say a host a wants to configure a device, so it uses this QUB kind of communication. So, it is a queued storage type of communication in the sense that, a queued storage is that you see for purpose of efficiency the communications between the see any communication a message sent by a sender and it is received by a receiver.

Now, both of this processes are actually programs, so they are doing their own jobs, so now, if we require that for example, it may happen that for better efficiency that is sender will actually send the data. And after it sends the data after it reaches a particular place in the receiver, sender will simply send the data and it will get stored, now when the receiver will actually use it is the receivers business. So, here you have a sender, here you have a some sort of a memory and here you have the receiver.

So, the sender is actually writing to the memory and then it is computing it is writing to the memory and then it is going on computing again. Now, when the receiver will read this memory and take this value is actually the receivers prerogative, so it may happen that before the receiver takes it the sender sends another value. So, in each case now there is a question, that whether if the receiver sends a second value, then is it going to erase the first value or is it going to be stored.

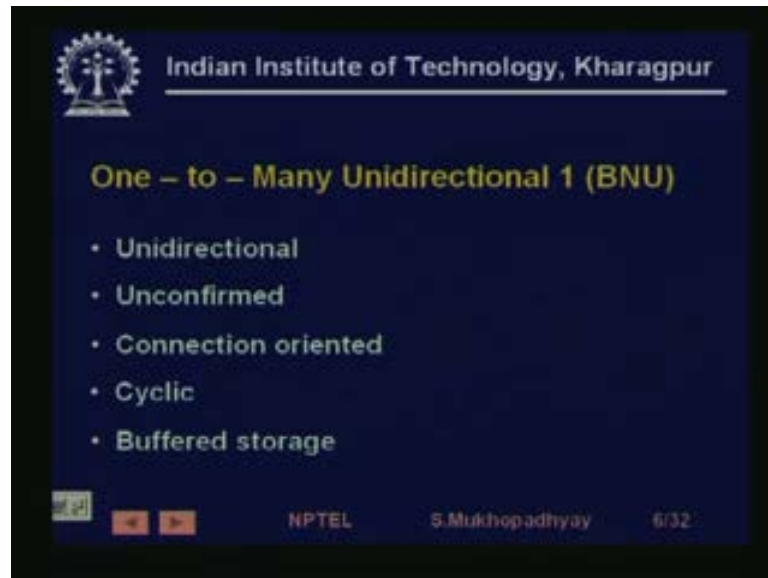
So, if it turns out that it will be stored or it may be stored in queue in a message queue, so that it will not be lost, unless the message queue overflows. So, at least for you know normally sender and receiver are generating and receiving data at the same rate on an average, because otherwise gradually messages will accumulate, especially if the sender rate is higher than the receiver rate.

So, but temporarily it may happen that the sender will send more data, than the receiver will receive. So, in such a case, so that the data is not lost, so therefore, you arrange for a queue, you can arrange for a queue, so that if a short intervals more messages are sent, than they are used by receiver, then they can be temporarily stored in a queue. So, that they are not lost, so this a queued storage kind of communication channel, that is offered it has confirmed service.

So, the sender gets a confirmation that the receiver has received it and it is bi directional, so you see infect these are for very important kind of for example, process device configuration is a very important function. Because, if the device has not be configured properly, then there will be very severe consequences, this is because that data is going to be used for a long, long time it is not like, you know you are continuously sending thousands of packets of data and then just one of them you got lost and it will be quickly over return by writing it another time.

So, it is not like that, here it is one piece data which is going to be use for long, long time. So, good amount of time when you are configuring device, so therefore, a confirmed service is actually needed, so and it is bidirectional, so this is a one to one bidirectional channel.

(Refer Slide Time: 13:51)



Then, you have one too many unidirectional channel of type one, so BNU let me just give you the B stands for Buffered, N stands for Network triggered and U stands for Unidirectional. So, this is a unidirectional service for example, typically an analog input sending data to a PID controller would use these, because it is never expected that the PID controller will send data to the analog input.

So, in this case unidirectional communication is sufficient, it is unconfirmed because, these communications are very generally consume the maximum bandwidth they are cyclic constantly repeated all the time at a certain sampling rate. So, the overhead on these communications have to be reduced, so if you use the confirmed service, typically confirmed service is going to take is going use more network bandwidth. Because, a confirmation is also has to be transmitted, so for communication which are very frequent we want to reduce that overhead.

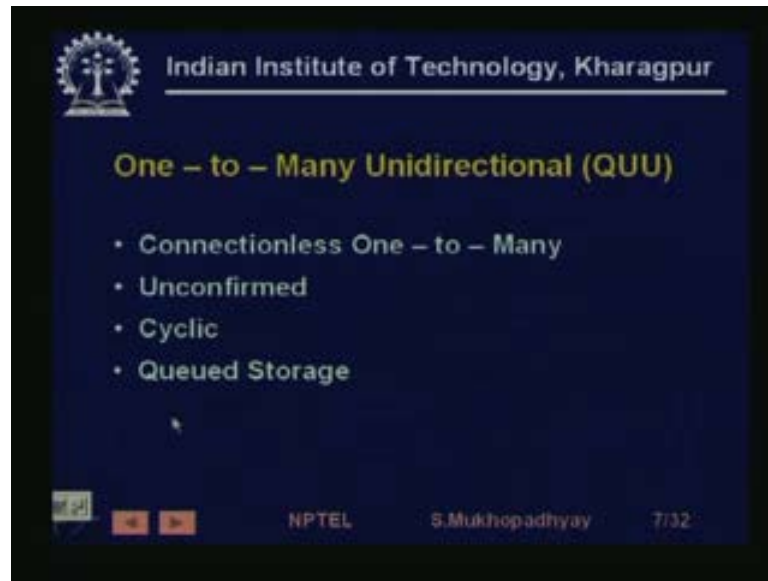
So, that they can be done that is finish quickly and more over even if one data is lost very soon it is going to be provided with another data. And thirdly, the chances of data lost have already been minimized by other engineering techniques which are used in the physical and the data link layers. So, therefore, chances are data getting lost are anyway low, so therefore, there is no point incurring that particular overhead of service confirmation.

So, this is an unconfirmed service, it is also connection oriented although you can send the data to more than one places. So, for example, an analog input can send the data to a PID controller for control, it can also send a data to some other course host device for monitor. So, it is cyclic communication, as I was saying that this is the communication which is actually used for transmitting the data in the feedback loops, so this one of the communication service which is very widely used and it is buffered storage.

So, you know when you can even imagine that when you have control computation, then if you have if by any chance one particular data, you always actually work with the current data. So, if by some chance you missed a sample there is little point in storing, there is actually little point in storing this data and another thing is that typically this communications are short. So, it is not the sometimes it happens that you are sending one communication, but that communication can be actually broken down into several packets for again for reasons of efficiency.

Now, if that happens and if a part of packet gets lost, then if the whole packet is useless in such a case you typically would require a queue. But, here you we are having short message packets which are being refreshed constantly and were generally only the current packet is of use. So, therefore, there is no need to have a large message queue, so therefore, we have you know just a simple network buffer, and where the sender process, simply hands that network packet to the network and the network. It is the responsibility of the network to put it in the buffer and the sender process implicitly assumes that the that data has been return. If there is no queue, so therefore, if the sender process sends two consecutive, say two consecutive values or two consecutive packets, then one will over write the other.

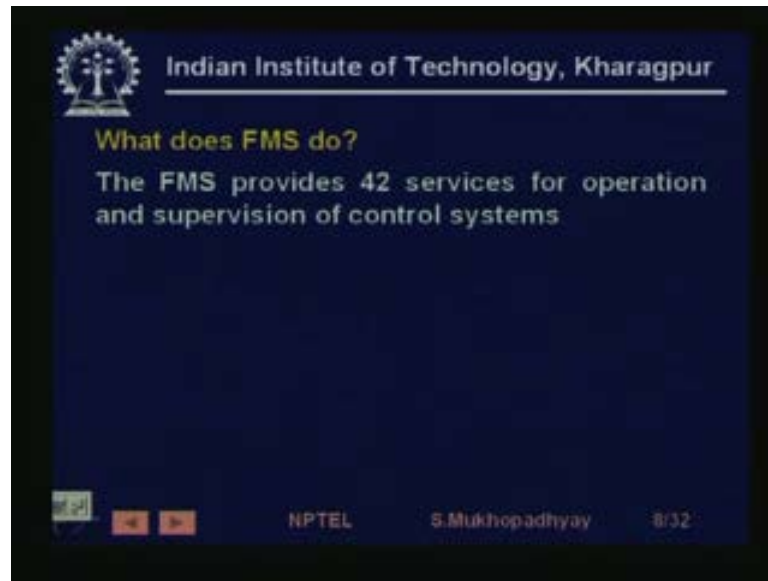
(Refer Slide Time: 18:44)



This is one to many unidirectional QUU, so this is queued this is user triggered and this is unidirectional it is also unconfirmed, it is connectionless in the sense that it will be broadcast and whoever needs it can pick it up, it is not cyclic, it is acyclic as far as I remember it is acyclic gone, it has queued storage. So, you see this is what the field bus access sub layer basically provides, it provides you with three types of channels.

And you can chose any one of this services from the user layer, you can request the field bus access sub layer to send your packets, which have been formed by the field bus message sub layer to send the formed packet to a particular node in the field bus network using a particular virtual communication channel and the field bus access sub layer will actually take care of the rest. So, you do not, so have to just pick the service and you have to pick the destination nodes, that is all you have to do and your packets is will reach.

(Refer Slide Time: 20:26)



Now, what does field bus message specification layer do it actually you know it actually the main thing that the field bus message specification does is that, it provides you know suppose finally, when a packet will actually reach the receiver it is nothing but a number of bits. So, the user process or the consuming process or whichever process will actually utilize this data has to know several things about the data, first of all in one packet several values will be send.

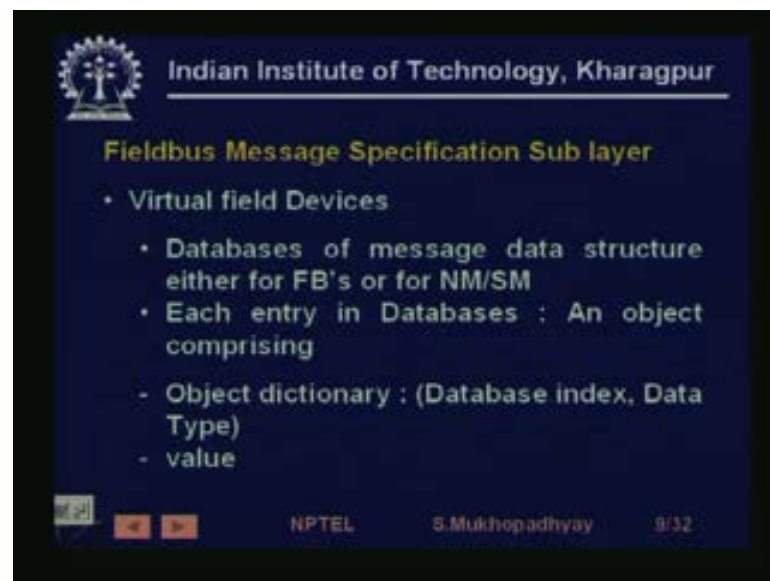
So, first of all it needs know what are these values, second of all it needs to know, so the when you have a long bits stream it needs to know, what are the formats, each length, length of each format, data types, which one is real, which one is character may be units. So, all these things, so basically the context of the data which is needed for interpreting the data, so that is to be provided by the field bus message specification sub layer, so the user layer need not be bothered about it, so the user layer is freed from it.

So, the user layer simply gives the value, saying that I want to transmit this temperature value to that node. And then the appropriate format by which the received process will understand that it is indeed a real number and expressed in which you need, so that it can use that value, that is achieved by transmitting that value in a particular message format. So, that message format is actually formed by the field bus message specification layer and so just like you know there are number of services, which the field bus message specification layer provides.

So, depending on the you actually transmit data in various contexts, so you know sometimes you transmit data for invoking a program. So, you want to start a process in another function block in another node, sometimes you want to access some variables, sometimes you want to manage some take action related to some alarms or events, time stamped events, sometimes you want to enquire the object dictionary structure of another process. So, that you know that you want to talk to that process and as we shall see what is object dictionary.

So, we will know that to able to talk to that process, you need to know what are the services that it supports and what format the it uses etcetera. So, you can actually enquire that know that and then write in that format, so that is kind of plug and play facility, so the processes or nodes which are interested in talking to another node can actually enquire the various facilities, which are supported by the receiver node. So, you see that there are various these are some of the context in which data is transmitted, so now, for each of the these context there are certain formats in which the message has to be send.

(Refer Slide Time: 23:50)



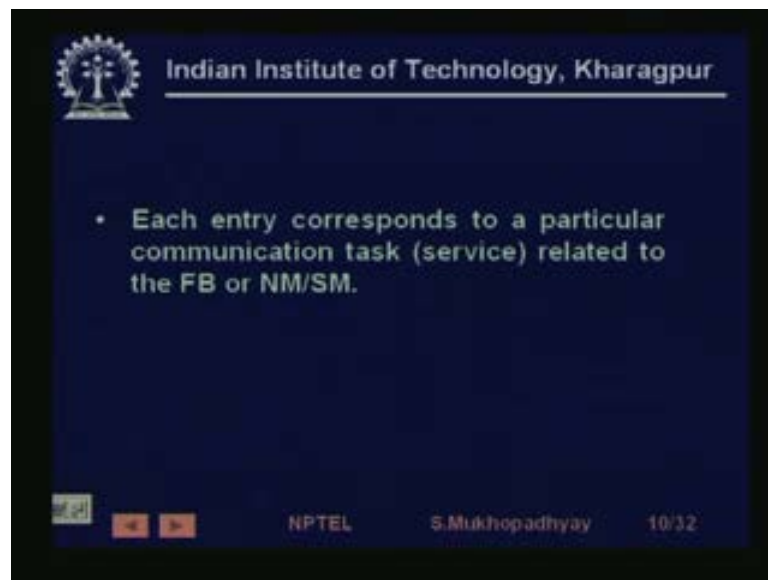
So, the field bus message specification sub layer actually is contains a data base of this format. So, that way it gives a view to the sending process let us say, that the or the process which it generates data or sometime even process which sends the data, that the other hand what is the view of the other end process, what are the services that it supports, what are the formats, what are the various parameters, configuration. So, it

creates an abstract view which we are referring to as a virtual field device of the process at the other end of the communication.

So, that is basically created by the field bus message sub specification sub layer, so it basically contains databases of message data structures, either for function blocks or for network management, system management functions. So, function blocks and network management system management are the user layer, so for various user layer processes it is creates an abstract view. So, that the other part is can either send data for the receiver to use it or it can interpret the data, that is coming from the sender.

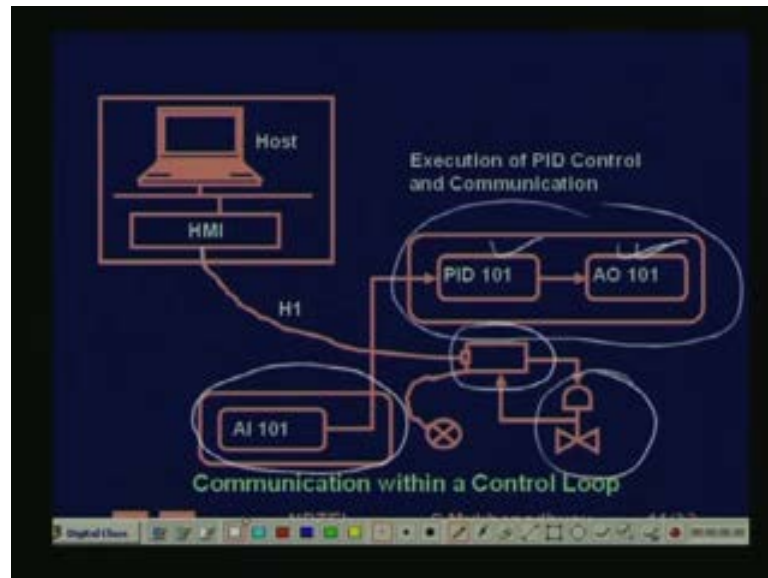
At each entry of this data structures are actually configured as objects, so they contains, first an object dictionary which will contained the database indexes for each item, their name tags, their indexes corresponding to the message data structures, etcetera data types. So, the description of each kind of data that virtual field device can support. And then finally, there will be a values, so each message will finally, it will have for each type there will be a value. So, for example, there will be a for database index, data type, pointers there will be all values.

(Refer Slide Time: 25:59)



Now, each entry corresponds to a particular communication task or service as we have mentioned, there are various kinds of services related to either the actual computation of the function block or the network management, system management of a particular field bus device node.

(Refer Slide Time: 26:16)



So, for example, this is our old example, where we have an this is the field bus device analog input, this is another field bus device which contains two function computational processes, one is a PID, another is an analog output and there is also a communication with the host. So, now, this is a typical structure of a simple control system and then finally, you have the for example, this is a valve, so the AO block will actually be talking to this valve and this is the hardware which actually talks to this valve devices.

(Refer Slide Time: 27:20)

Indian Institute of Technology, Kharagpur

Architecture of a Field Bus Device

- **Transducer Block**
Converts sensor valve to digital representation for function block.
Vender dependent.
- **Function Block**
software module that provides abstract functional representation

NPTEL 5.Mukhopadhyay 12/32

So, now this is the architecture of a field bus loop, now let us look at the architecture of a field bus device. So, now, we are coming to the user layer, so you see that in the user layer whenever you have a field bus device, typically field bus is used for talking to real world devices. So, the first thing, so it actually does lot of I/O and inputs outputs of actual signals, so therefore, naturally this real device is actually piece of hardware, which is existing in the field.

So, that first block and it is sending signal, so that signal you may be is getting process internally by some other electronics hardware. And then finally, it is coming on to the software process, now you see that we are requiring with for what we need for communication, we need for communication is that each of these devices should have particular standard interface. So, that other devices which want to talk to it, knows in what language to talk, so the front end of each of these devices should be standard.

So, that other devices can talk to this front end and can transmit data, this is the requirement for interoperability. On the other hand, but you can never say that all field devices must be built in a certain way, how they will be built actually depends on manufacturer. So, the manufacturer you can should always be allowed to use a much better signal processing algorithm, which may be invented later on or some he may use of use a particular kind of sensor, which will send data in a certain way he should be able change it whenever I mean is you can change is product line.

So, there has to be, so while you are trying to have standardization and interoperability you cannot stifle, you know flexibility or freedom of building a device. So, therefore, the field bus devices are actually broken up into two parts, so any device will have two parts, one part will be the vendor specific part, this is it is internal construction the no standardization, the vendor can use whatever he wants. And this part generally interfaces to the external signals, external entities, but once this data are computed there has also has to be a standard interface, standard computing and interface support.

Computing is necessary, because for quick configurability you know what the basic idea is that industrial control computations are not of infinity varieties, they probably 80 or 90 percent of the computation are of a certain types, when you have a controller probably 85 percent of you are controllers can be realized by having within some 10 different algorithms.

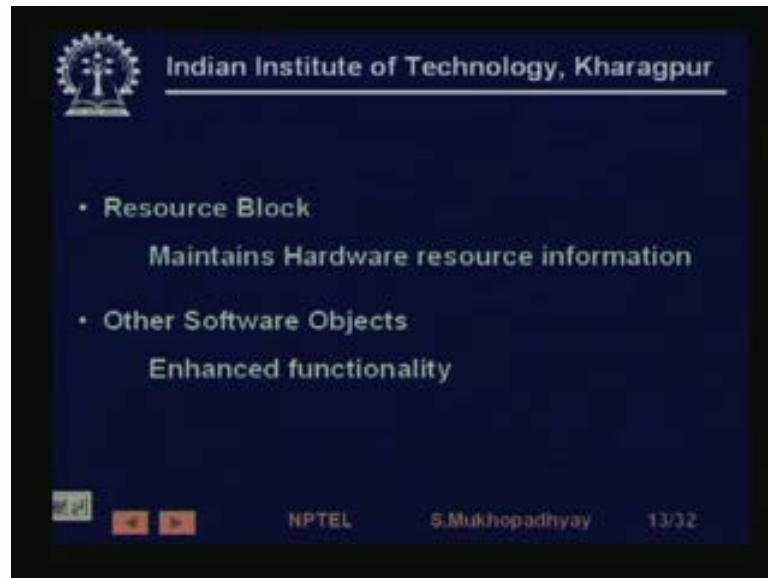
So, if you, so for ease of programming you can always provides some templates, like a like a PID computation template, which we were where the user does not have to write any code, you simply give the values of KP and KIND, rather K_PK_I and K_D values. And then the computation will be done, that is for ease of programming, but ideally speaking the user is still free to put his own very possibly, very great and very advanced algorithm only thing is that, if he has to put that then he has to put it in the vendors specific part or otherwise there is also a standard part this is the standardized part standardized part.

So, this standardized part is used in two ways, either they consist of some typical computational templates, which you can use for easily quickly building up your system and they are used for interfacing with other standardize parts. So, it is this vendor specific part which is called the transducer block, so the transducer block converts sensor values, this is values to digital representation for the function block. So, it can first it does it is own vendor specific computation and then it gives it to a function block and then this function block talks to other function block.

So, as far as interoperability it is concerned as far as field bus communications are concerned it is always between function blocks. While, at the behind the function blocks then there may be proprietary code there may be proprietary code technology, which will simply do the computation and feed them to function blocks and then the function blocks will talk to the other function blocks, this the idea, so therefore, you have transducer blocks.

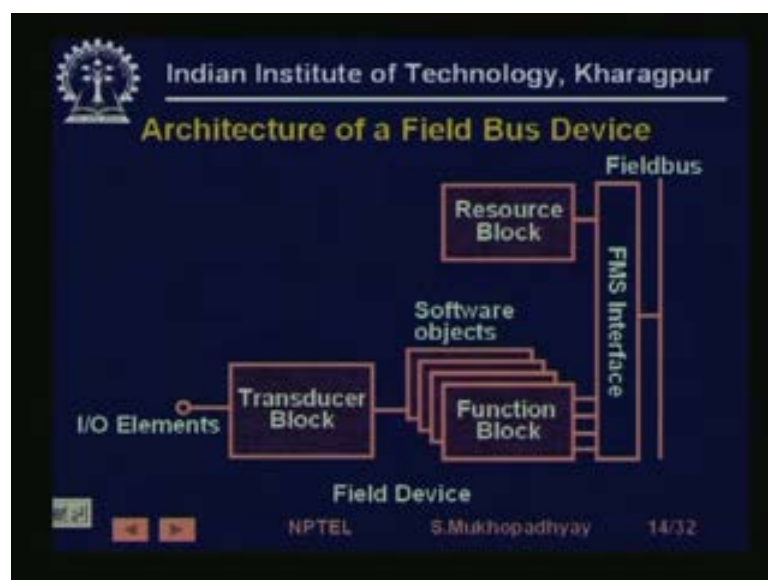
So, function blocks are software modules that provides abstract functional no not only abstract standardize functional representation of the kind of computation and communication that want, not even communication just computation, your communication is taken care of the lower parts.

(Refer Slide Time: 33:10)



There are two other kinds of blocks for example, there are resources blocks which keep track of the hardware resource, the kinds of because these are this can be enquired by other devices. So, this is more a management requirement and there are various other kinds of software objects, which you can use for you know for supporting some typical functions efficiently.

(Refer Slide Time: 33:41)

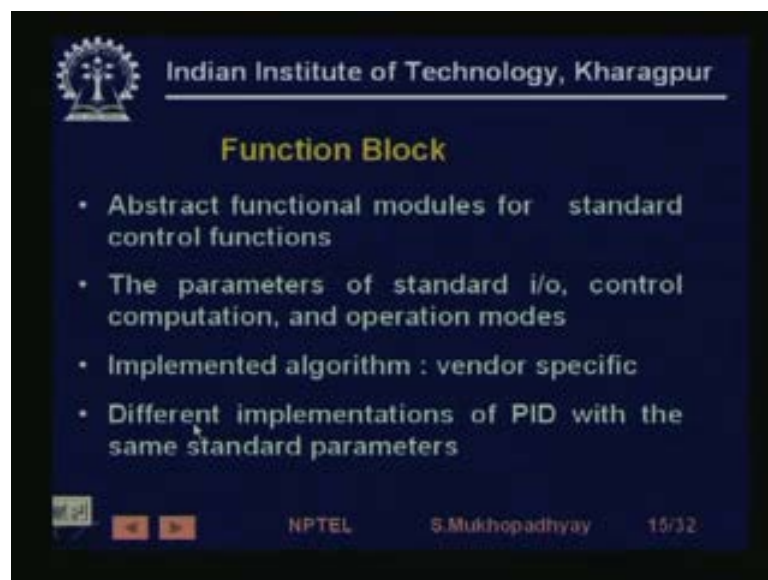


So, because they may not be required, so they may separated out, so typically you know you have this is a very simple device, where you have transducer block, you may have

several function blocks and you have resource block only one resource block is possible. And then finally, all these will interface to the FMS, so all these will finally, tell the FMS that here is the this left and side will actually generate the data and then tell the FMS, that what is the context of the data there is already stored in the FMS interface, that if a particular function blocks is giving a data what is context.

And then and similarly it will be the kind of communication channel that it is communication service that it is asking for that will also expressed. And then the FMS and the FAS will actually send the data through the network, so the user process is not concerned with communication at all. But, just provides the data it is context and the some request for communication, virtual communication.

(Refer Slide Time: 34:58)



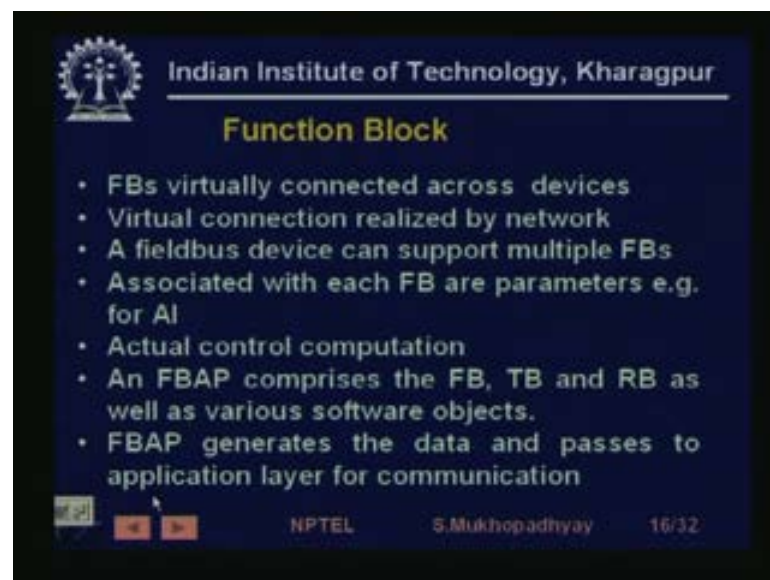
So, abstract functional modules for standard you know control and automation functions, the parameters of standard I/O control computation and operational modes. So, the function block can contain a large number of parameters, some of them are input and output parameters which are signals, some of them are called contained parameters or you know these internal values which are used in the computation of the function block.

You can configure, which one of them can be written by various other devices, which one them cannot be written over written, some of them can be read. So, various parameters and their types read, write access etcetera; obviously, it contains an algorithm which is the algorithm is not if you are putting the algorithm in the function block, then it

has to be it actually you must build it by choosing some a list of it is not vendor specific and, but although field bus it is vendor specific, vendor specific is the transducer block.

So, but you have a large library of common function blocks define, which is continuously increasing and you can implement your algorithm using that library. For example, you can have different implementations of PID with the same standard parameters, so you see that what we are saying is that you are only defining a PID, but may be sometimes you can implement an algorithm which has the same interface, but different algorithm that is possible.

(Refer Slide Time: 37:03)

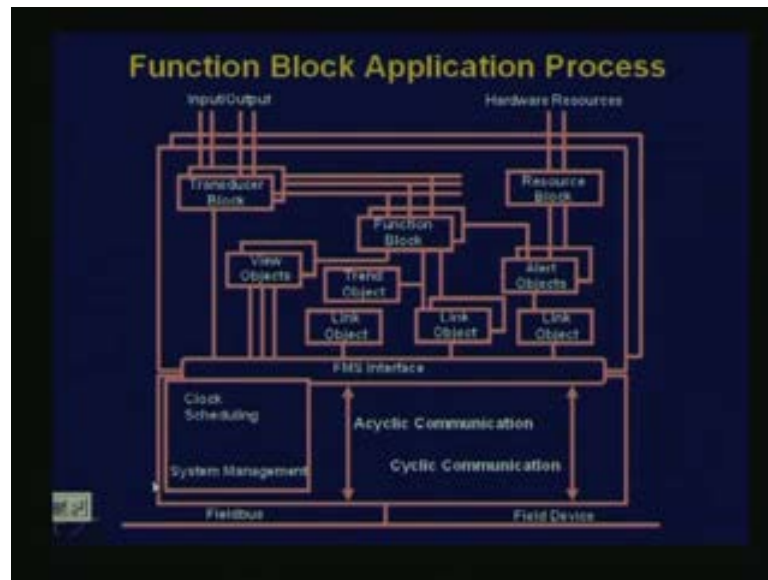


And this function blocks are virtually connected across the networks, so when a AIE tracks to a PID it is immaterial or it is not known to it, where in the network it actually missiles. So, it just say there I need to send this value to PID and then the rest of the network achieves that, so the virtual connection is actually realized by the network, field bus device can support multiple function blocks. So, there can be multiple function blocks in one field bus device, in the typically are an associated with each function block are various parameters, there is some computation.

And apart from these they are some other types of computation and together, so all these computations actually finally, provide the functionality of the device, that what these device can do. So, what monitoring functions it have, what self diagnostics it has, what calibration facilities it has, what control facilities, etcetera, so all these software together

constitutes what is known as a function block application process. So, this contains comprises the function blocks, the transducer block and resource block, as well as various software objects. So, what are those software objects will see and it generates the data and passes to the application layers for communication as we have seen.

(Refer Slide Time: 38:38)



So, here is a view which you of the function block application process, so you see that there are transducer blocks, which actually interface with the I/O electronics and there are the function blocks, which actually compute the core functionality of the device. Apart from and there is a resources block, which keeps track of the resources in the system, apart from these you have a number of objects like for example, you have view objects, you have trend objects, you have alert objects, you have etcetera.

So, these are typically objects which actually talk to the function blocks, they are inside the function block application process. And they provides certain specific kind of services for example, an alert object may continuously monitor these function block data and then if some configured event, some event condition that you have configured, the some temperature going higher than other value, etcetera.

Then, it can actually generate an event, times stamp it there are such time and event was generated and then communicated. So, this communication functions have been separated and they are independently scheduled units in the software all these software, similarly when you have trend or view objects these actually you know trend data is need

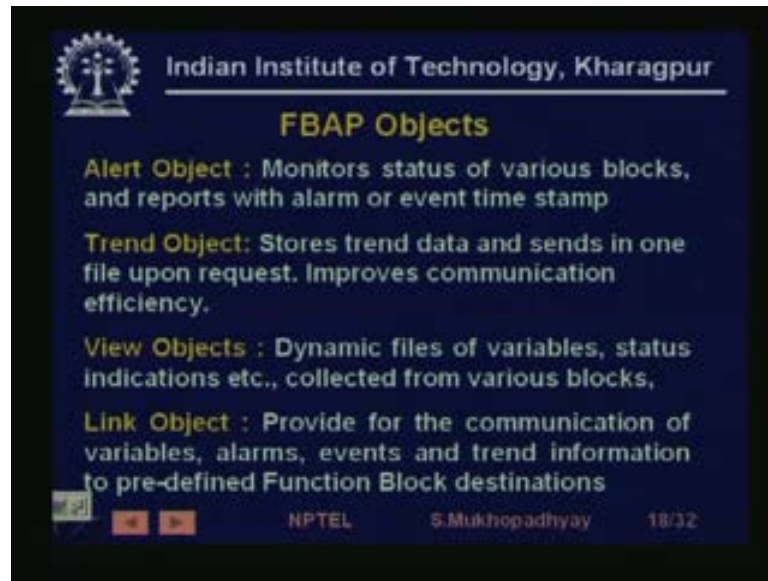
not be send, so frequently. So, you know you can accumulate the data and probably average it over some interval and then send an average data and that will be adequate for doing trending.

So, such averaging computation and then putting a lot of data in within one packet and then sending it just for trending, such functionality are typically supported by the trend object. Similarly, you may have view object which can create you know I mean dynamic files of various variables for giving at some host or at some operator station. So, these are separate, separate objects which view which free the main function block and from executing these tasks which are typically infrequent and they separate these tasks as independently schedulable units.

Then, you have some link objects, so the link objects are after all these some of these objects, actually communicate with the FMS through a link object. So, the link objects jobs is actually you know interpreting the various address tag, assignments and then asking for particular service I mean mentioning which service is to be used. And then sending that request, after adding all those things sending that request to the FMS and then the FMS works further below and does either cyclic or acyclic communication as required.

Then, there is always system management function, so various kinds of tasks are needed as we shall see regarding related to time, related to network address management etcetera. So, those are done by these system management and network management functionality.

(Refer Slide Time: 41:58)



Indian Institute of Technology, Kharagpur

FBAP Objects

- Alert Object** : Monitors status of various blocks, and reports with alarm or event time stamp
- Trend Object**: Stores trend data and sends in one file upon request. Improves communication efficiency.
- View Objects** : Dynamic files of variables, status indications etc., collected from various blocks.
- Link Object** : Provide for the communication of variables, alarms, events and trend information to pre-defined Function Block destinations

NPTEL S.Mukhopadhyay 18/32

So, here we have the alert objects, monitors status of various blocks and reports with alarm or event time stamp as I said, trend objects stores trend data and sends in one file. So, basically you know when you are sending trend data every communication has an overhead, so if trend data is required infrequently at one shot it can update some pass data.

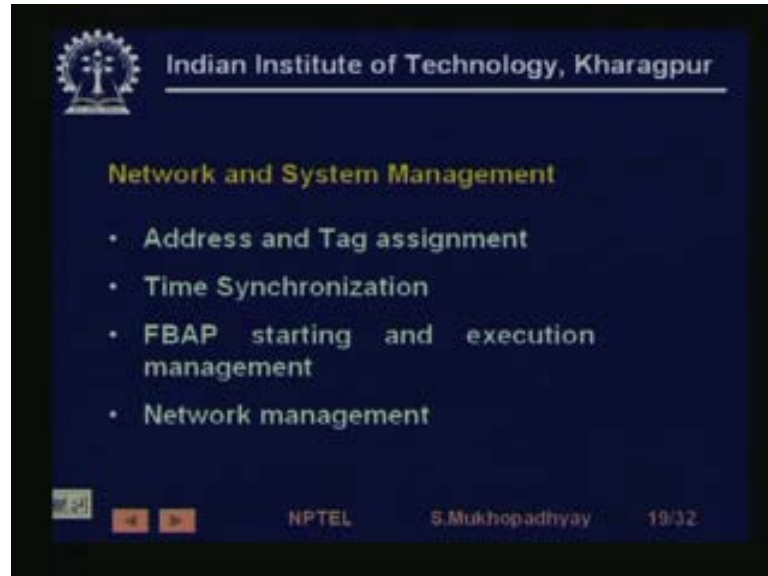
So, therefore, it will have much less overhead if you can you know club rather than sending trend data very frequently, store the data probably compress it in some way, probably average it out and then send it in one file. So, that the trend will be updated for a, so trend updating has to be for long interval, so these will improve will reduce communication overhead and will improve communication efficiency.

Similarly, view objects are dynamic files of variables, status indications, etcetera which you use on typically use for monitoring and supervision and operator station. So, they actually you know put together probably a view of the process by taking values from different function blocks and then putting them in a certain defined view. So, a certain defined view of the process for example, when a process is working in a certain mode, let us say in a maintenance mode or in a some test mode or some manual mode.

So, in each mode you may require a different view configuration, so in that mode you would like to see those quantities related to the process. So, such view objects can actually provide you those views, similarly link objects provide for the communication

of variables, alarms, events all these that other objects. So, the function blocks needs to transmit and they define the destination and the required service type to the FMS.

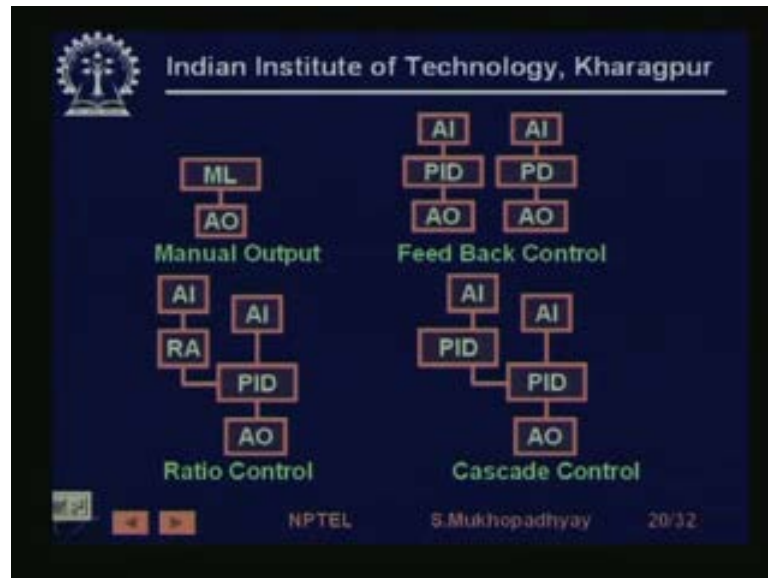
(Refer Slide Time: 44:00)



You have network and system management functions, which are address and tag assignments. So, you see every device is going to have a network address and each device there can be even hundreds of different, different signals each one of them have to be assigned a unique address unique tag. So, it is with this address and tag that when a particular data arrives first of all we using address it will arrive at node and then after it has arrived at that node, what data it is and how it is to be used that will one of thing major things which will determined that is the tag.

That is which particular signal are you getting, then there is an absolute need for time synchronization. As we have seen that it requires a very because, it is real time communication of hundreds of function block processes, so requires absolutely properly scheduled communication. So, therefore, time synchronization that is every on the bus segment everybody has a same sense of time is actually very important and therefore, various time synchronization techniques are there, so such time synchronization has to be done. Then, certain time, certain function block application process has to be started and their execution has to be managed. And finally, there are you know other network management functions, which are carried by this network and system management.

(Refer Slide Time: 45:28)

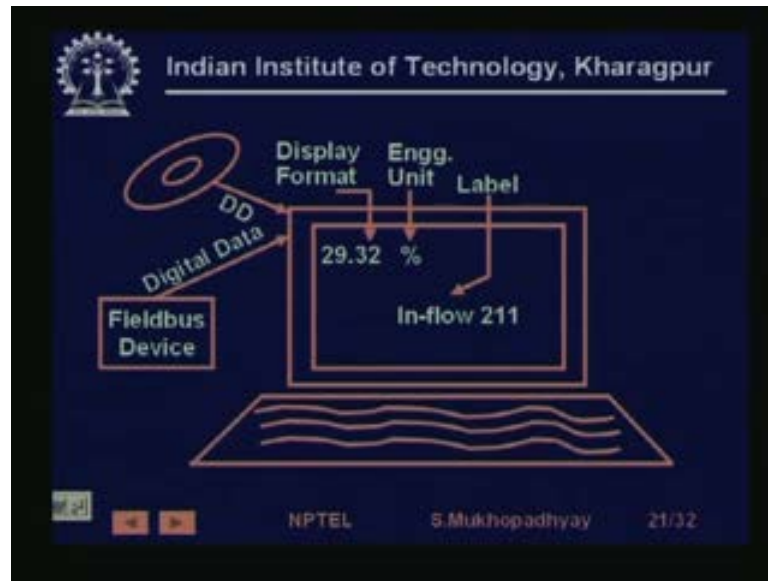


So, you see here we have there are as I said that field bus standard provides a large number of function blocks, which is ever growing. So, some of the for example, you can have a manual output, so you take up manual input, so from manual device and you directly send it analog output, so this is manual outputting. Similarly, you can have a feedback control you can either have a PID algorithms and AI PID and AO or you can have PD control algorithms. So, AI, PD and AO you can several kinds of PID, PD and various other things for example, here you have ratio control.

So, in ratio control a fraction of one particular stream is used as set point for another particular stream. So, you see that this is the used at the set point for the PID and this is the use of the feedback, so finally the PID is doing ratio control, similarly you have cascade control, in cascade control again the output of the outer PID block is used a set point of the inner PID block.

So, this is the outer PID block where you are getting AI as a feedback then the output of that is being used a set points to other PID block and finally, and this is the feedback of the inner loop, so this you get the final out. So, you see several such algorithms can be configured using standard function blocks, libraries available with field bus standard.

(Refer Slide Time: 46:59)



Similarly, there are you know field bus has gives you facilities for doing as they say doing plug and play operation. So, what happens is that there are even although it is configured or lot of it is the function block interface is standardized, but still you need to have you know for each device you need to provide to the host, what are configuration values it actually uses or it should use.

So, that when the device is actually connected to the network using the host or the network management and system management functions, one can configured the various one can configured devices, one can also set the parameters which are to be used while interpreting the data from the device. So, there is some sort of it is like a device driver, so along with the device one needs a device driver, in this case it is called a device description and it is written in a particular standard language called device description language.

And which is actually supplied along with each field bus device, it is supplied by the manufacture it has to be either supplied or if it is very standard device in can sometime we know download it from the net. But, in any case once you have the device description and once and then if you receive data from the field bus device, then you can interpret the data and you can even configure the data.

(Refer Slide Time: 48:47)



Indian Institute of Technology, Kharagpur

Device Descriptions

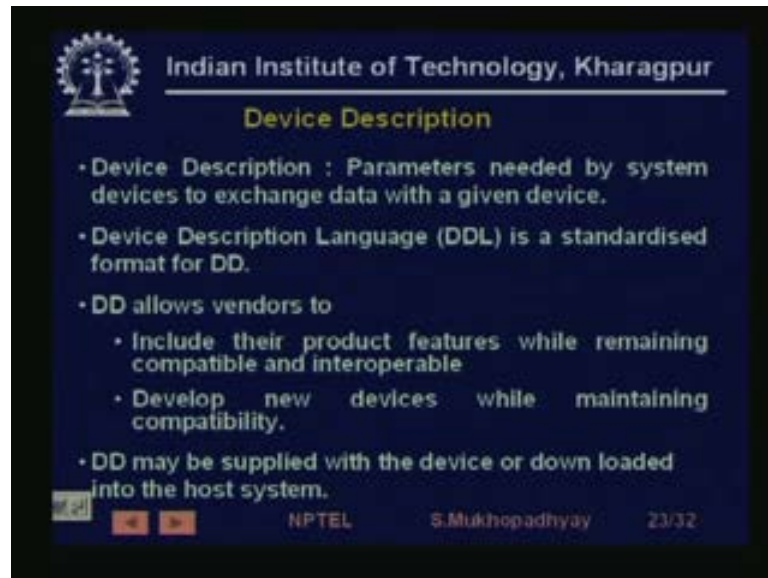
- Analogous to device drivers (Vendor specific)
- Provides info related to
- Communication formats including configuration, calibration, diagnostics.
- DDs are written in DDL
- Describes function blocks
- Plug and Play-like abilities

NPTEL S.Mukhopadhyay 22/32

So, this is a devised description which is used and provides various information related to communication formats, including configuration, calibration. So, you can know what kind of calibration facilities it actually supports and how does it do the calibration, how to interpret the calibration data, various self diagnostic information, etcetera. So, these information's are provided along with the devices in a specific format and they can be used to for to you know incorporate the device into the network and using it effectively very easily DD's are written in DD languages.

So, and it also describes the various function blocks, and it has a plug and play like facility. So, once the device is put on the network using this device description the host will know, how to configure it, how to detect it, how to interpret data, etcetera, so installation becomes very easy.

(Refer Slide Time: 50:02)



Indian Institute of Technology, Kharagpur

Device Description

- Device Description : Parameters needed by system devices to exchange data with a given device.
- Device Description Language (DDL) is a standardised format for DD.
- DD allows vendors to
 - Include their product features while remaining compatible and interoperable
 - Develop new devices while maintaining compatibility.
- DD may be supplied with the device or down loaded into the host system.

NPTEL S.Mukhopadhyay 23/32

Same thing device, description parameters needed by system devices to exchange data with a given device. Device description language is a standardized format for DD, DD allows vendors to. So, you can include your own product features, you can upgrade the products features, while remaining interoperable develop new device while maintaining compatibility and DD may be supplied.

(Refer Slide Time: 50:30)



Indian Institute of Technology, Kharagpur

Interoperability

- Open architecture system
- Full compatibility of Fieldbus Standard compliant devices
- Clear separation between standard and vendor specific parts
 - Allows flexibility in technology
 - Guarantees inter operability
- Promotes Competition

NPTEL S.Mukhopadhyay 24/32

Interoperability as you see is actually a very important concept, that this you know this is they are in is coming in all products, all standardize products. For example, when you

have instrument which is IEEE 488 compatible and it means that it supports a particular communication network and you can, while the instrument part of it can be proprietary, but it possible for other device to actually exchange data with device using a standard interface.

Similarly, when you have object orient programming, you have public part of in you object and private part. So, the private part is where the you can change any implementation at will not affect any other part of the software, so it is the similar concept that you need to be interoperable, you need to have plug and play facilities, but the same time you and your own implementation you can keep it hidden.

So, full compatibility of field bus standard compliant devices and clear separation between standard and vendor specific parts, allows flexibility in technology and, but guarantees inter operability. And obviously, when you have interoperable it promotes competition, because now you can your any it is not company dependant, so all other companies product will also work.

(Refer Slide Time: 51:57)

Indian Institute of Technology, Kharagpur

Time Management

- Link Schedule Time : Common clock for each bus
- The LAS uses this to synchronize devices on the bus
- System management synchronizes function blocks on the bus using LST
 - Triggers FBs at appropriate LST offsets from start of Macro Cycle to ensure synchronised distributed computing of real-time control
- Application time (AP-time) : Real time reference used for event time-stamps.

NPTEL S.Mukhopadhyay 25/32

Now, some of the system and network management functions for example, time management, there is a link scheduled time. So, you know everybody will have to start talking, stop talking and start listening or stop listening according to some time schedules, which are distributed by the link active schedulers. So, unless the device itself has keeps track of time, it will not be able to do that, so there is standard time which is

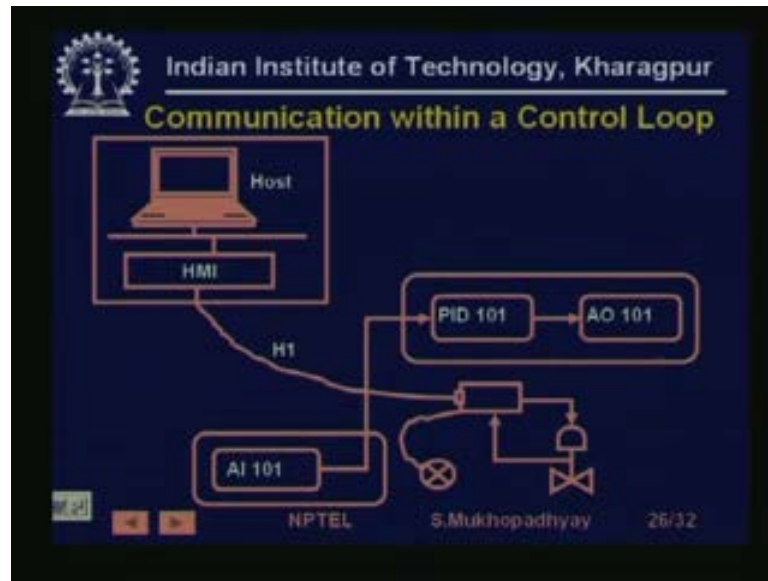
maintained in the link, which is called the link scheduled time it is common clock for each bus.

And the link active scheduler uses this to synchronize devices on the bus and this time has to be synchronized. So, time has to be the system management synchronizes this function blocks on the bus using LST, so it has to the link active scheduler, this time has to be maintained by the link active scheduler. And the system management synchronizes of each device, must synchronize different function block executions and communications with respect to this time, because unless it does that it from the LAS and the using the link scheduled time, it will know that when which function block has to execute. So, after an AI computation only then a PID can execute, so unless from the link active scheduler commands, we have seen that it will distribute tokens from the arrival point of this tokens it will know that when it can scheduled. So, when there is an arrival when it sees a CD for a compiled data meant for an AI it knows that the data is now coming and then after data comes to the data will be coming within this time slot and after the data come, then it can scheduled it is PI.

So, such execution scheduling can be done only you maintain your own common sense of time. So, it triggers function blocks at appropriate link scheduled time offsets from the start of the macro cycle to ensure synchronized distributed computing of real time control. So, it says that he knows that when the AI phase is going to be complete, so it schedules the PID block after that, so that when the PID block picks up the data from the buffer, it will receive the latest data which is sent by the AI, so this kind of scheduling has to be done.

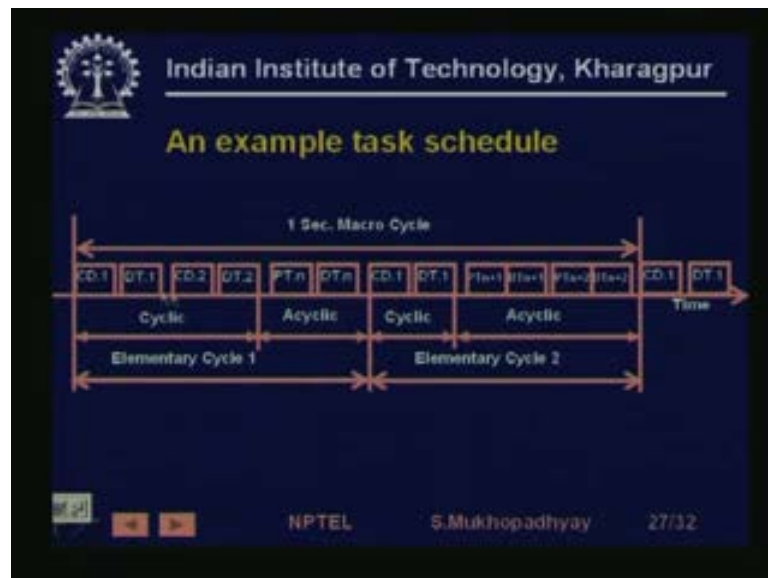
And for application time this is you know for particular ((Refer Time: 54:35)) general you know for general events stamping and later on you know application time is a full system wild time which has to be maintained, because later on all the event times, stamps etcetera have to be interpreted.

(Refer Slide Time: 54:48)



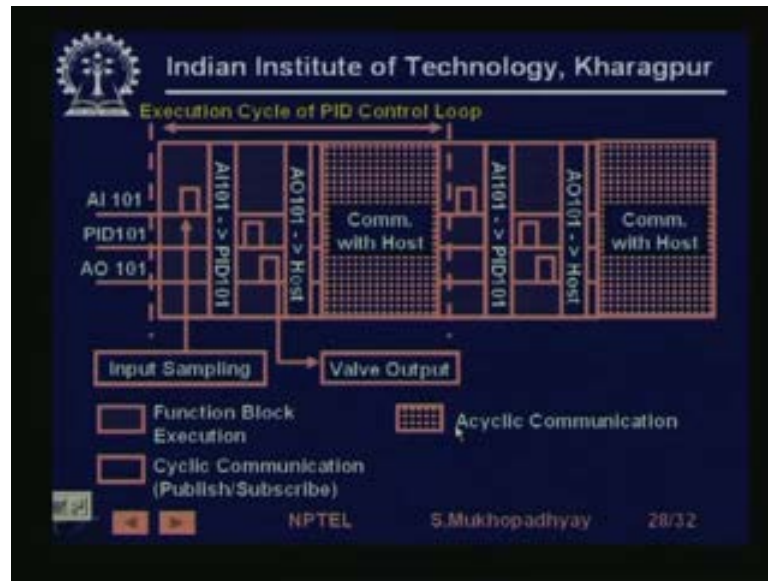
So, here we have communication within a control loop that same diagram, so we will skip it.

(Refer Slide Time: 54:53)



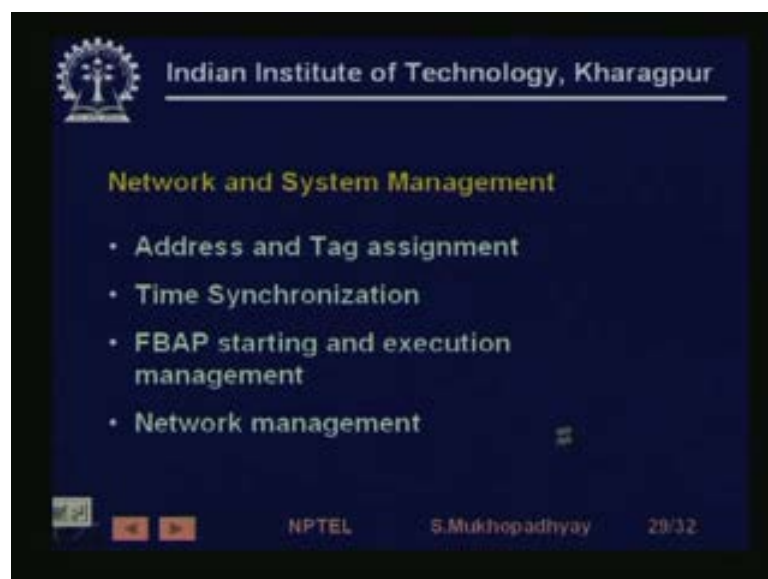
And we have already seen these that how we communicate using cyclic and acyclic communication over macro cycles and using elementary cycles.

(Refer Slide Time: 55:06)



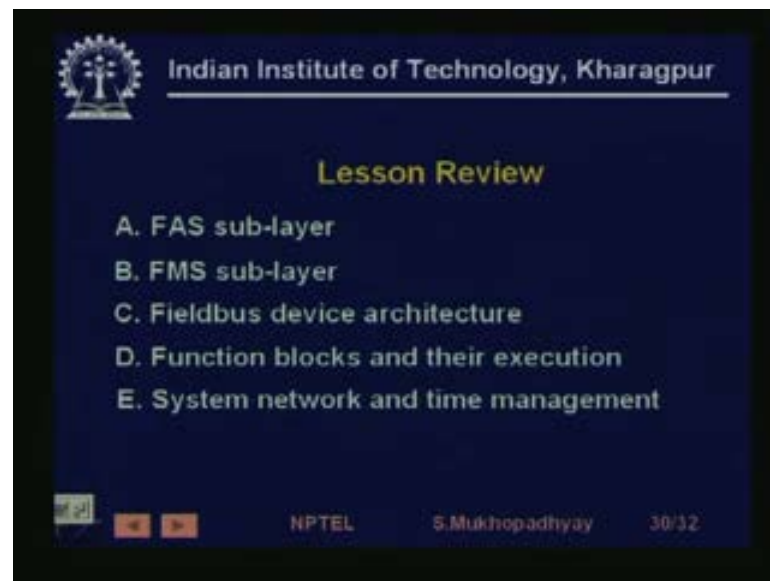
And we have also seen this is, so you see that is what I was talking that the inputs will be sampled, then there will be communication, then the PID block will execute, then the analog output block will execute. Because, there on the same device no communication needed, then the output block will communicate to host, so in this way communication will be scheduled by the system management, the rather the computation will be scheduled by the system management functionality for control.

(Refer Slide Time: 55:37)



So, an again network and system management, so it basically involves address and tag assignments and maintaining them and when a device comes on recognizing that device, when a device goes out of the network, removing it from the scheduling list. So, all these are network address and tag assignment and management, then time synchronization, function blocks starting and execution management and network management.

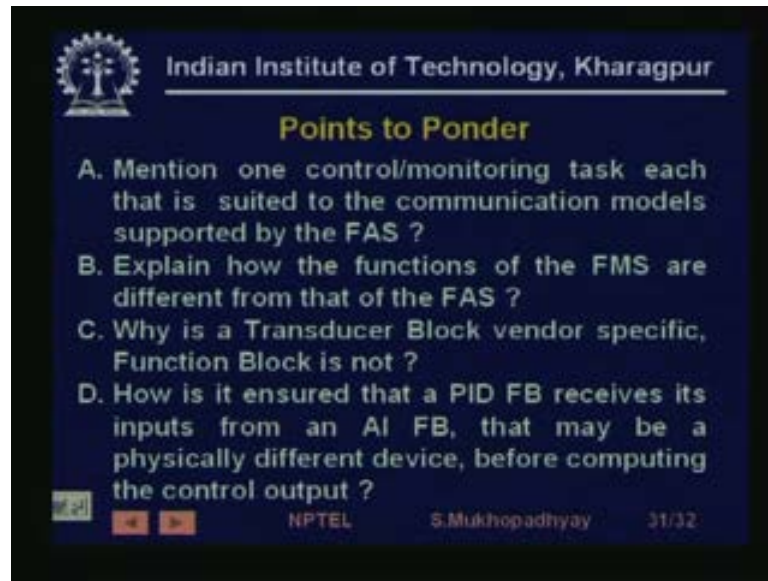
(Refer Slide Time: 56:06)



So, we have come to the end before that let us review the lesson, so we have look at the FAS sub layer, which basically defines virtual communication channel and provides three different kinds of communication channels. Then, we looked at the FMS sub layer, which provides 42 or 42 is number it the keep changing, so different kinds of information services and provides the message data of structures for those service.

And finally, we saw the field bus device architecture, which actually let us the manufacturers provides standard interface to they are computation, while maintaining their own proprietary flexibility. So, we saw the function block and their execution and the system and network and time management function.

(Refer Slide Time: 56:51)



Indian Institute of Technology, Kharagpur

Points to Ponder

- A. Mention one control/monitoring task each that is suited to the communication models supported by the FAS ?
- B. Explain how the functions of the FMS are different from that of the FAS ?
- C. Why is a Transducer Block vendor specific, Function Block is not ?
- D. How is it ensured that a PID FB receives its inputs from an AI FB, that may be a physically different device, before computing the control output ?

NPTEL S.Mukhopadhyay 31/32

Before we end points to ponder, so mention one control monitoring task each, that is suited to the communication models supported by the FAS. So, QUB, BNU, QUU for each one try to find a task, which can use that model. Explain, how the functions of the FMS are different from that of the FAS this is clear, why is a transducer blocks vendor specific function block is not, this is also clear mentioned in next lesson. How is it ensured that a PID, FB receives it is inputs that is by scheduling done by the system management using link scheduled time, that is all.

Thank you very much.