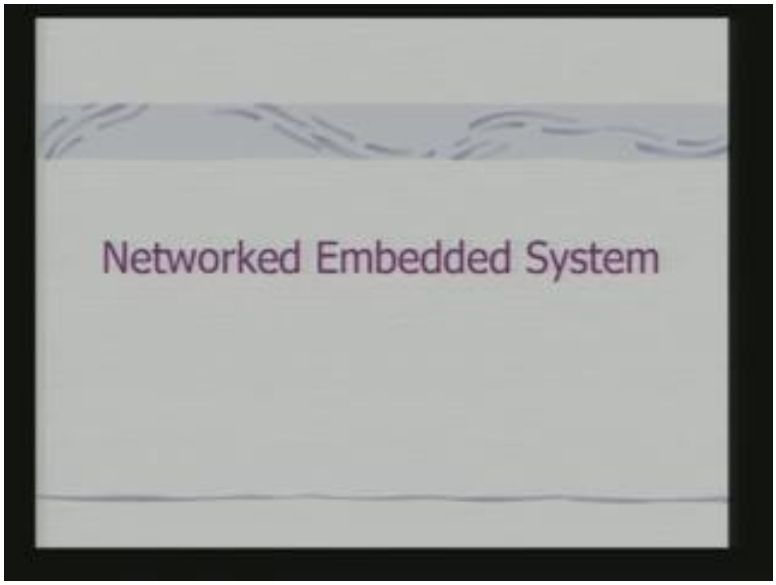
Embedded Systems
Dr. Santanu Chaudhury
Department of Electrical Engineering
Indian Institute of Technology, Delhi

Lecture – 24
Networked Embedded System

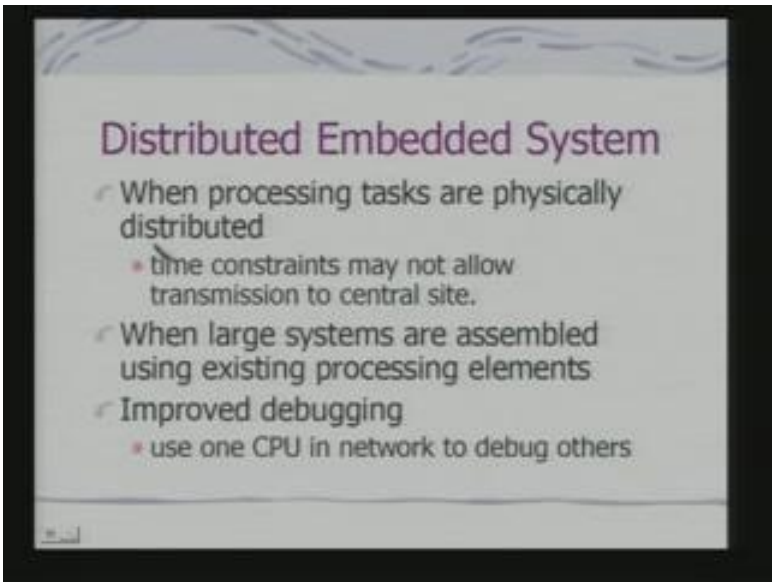
Today, we shall discuss networked embedded system.

(Refer Slide Time: 01:18)



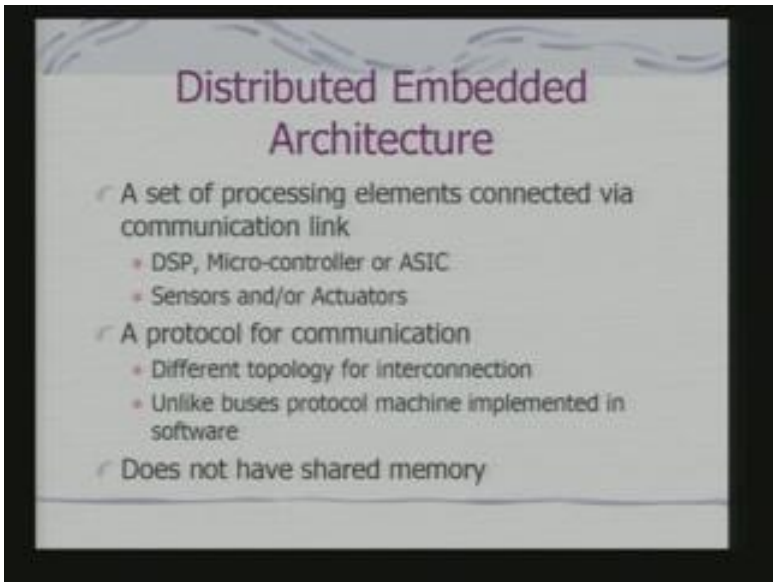
The embedded systems when they are networked they actually provide us a distributing computing environment. Now, how this embedded systems can be networked? And secondly, what kind of applications we should look at for designing this kind of a networked embedded system based design? That will be the concern of our next few lectures. So, why shall we go for distributed embedded system?

(Refer Slide Time: 01:57)



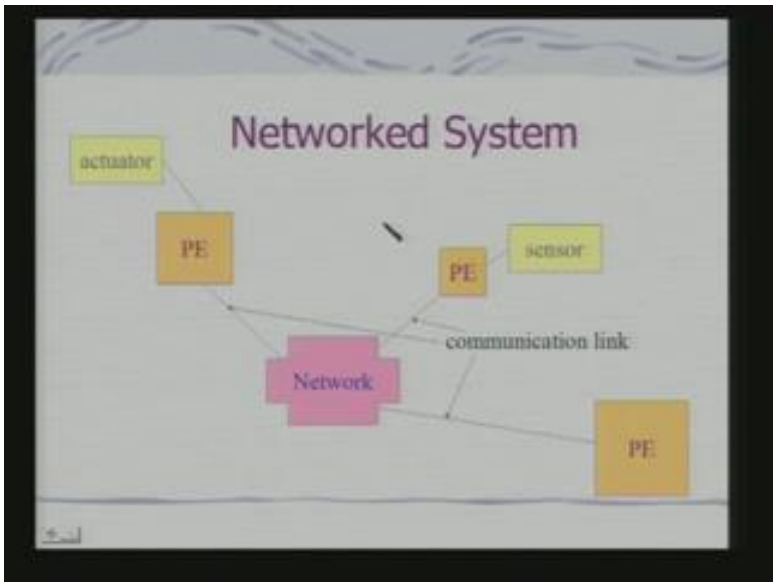
When processing tasks are physically distributed this a obvious choice, because the time constraints may not allow us the transmission of the data from the senses to the central site. And therefore, we would like to have the processing at the note itself sensor note itself. And then communicate may be the process data to a central node are may communicate the process data among this nodes which are physically distributed. Therefore, depending on the requirements of the problem we may need to have distributed embedded system. The other interesting thing is a when large systems are assembled using existing processing element. So, there are individual multiple processes have been used and even in associates today the train list to set up an network among the multiple processes which have been used on the same associate server. So, you have what is called network of process. So, that another reason for going for network embedded system 1 obvious consequence of networking is improved debugging, because you can use 1 CPU or the system in a network to debug others

(Refer Slide Time: 03:20)



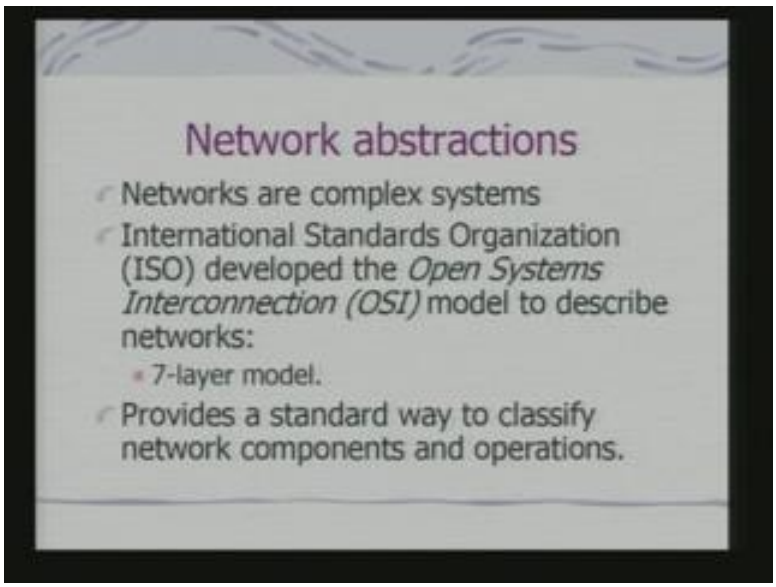
So, what kind of architectures you expect for distributed embedded system? Obviously, a set of processing element should be connected via communication link. This processing elements may be DSP may be a micro controller or an ASIC even it can be an ASCI also there would be sensors and or actuators. All these things would communicate based on a protocol for communication and there would be a different topology with which this elements can be connected and unlike buses we have already discuss buses. Some of the buses also have various kinds of protocol based features unlike buses the protocol machine is typically implemented in software in theses network systems. Although when we are talking about implementation of the software we are not eliminating the possibility of having hardware support to implement such software. And the most importantly such network processes will not have shared memory. This is the most important distinguishing feature when we say that a set of processors that connected via bus are there connected via network. So, typically a networked system would like something like this.

(Refer Slide Time: 04:42)



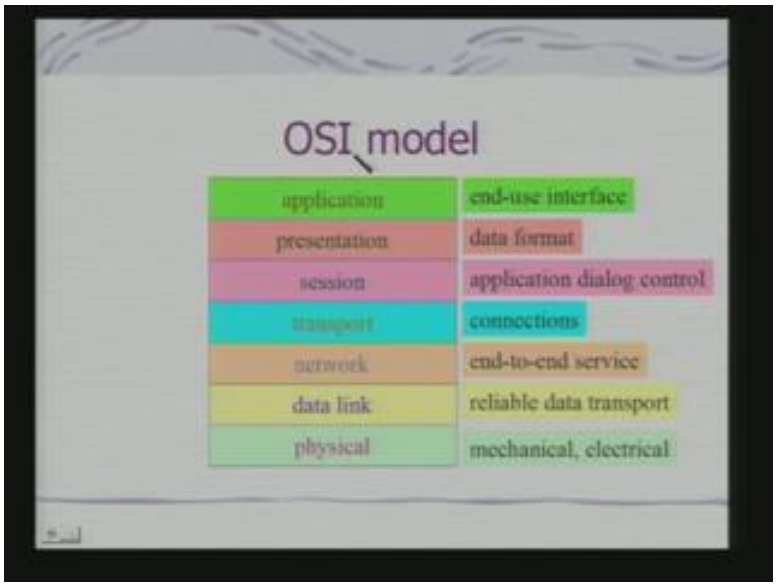
These is are all processing element this is a communication link this is providing the basic network backbone. There are this actuators which are connected to the processing elements they are the sensors connected to the processing elements and there could be a standard processing element as well. And when we talking of the network it is a network base is actually may consist of hardware elements as well and those hardware elements by the way are also embedded systems. For example, if you consider a outer in a network that itself is an embedded system. So, PEs may be CPUs or ASICs as the case may be.

(Refer Slide Time: 05:25)



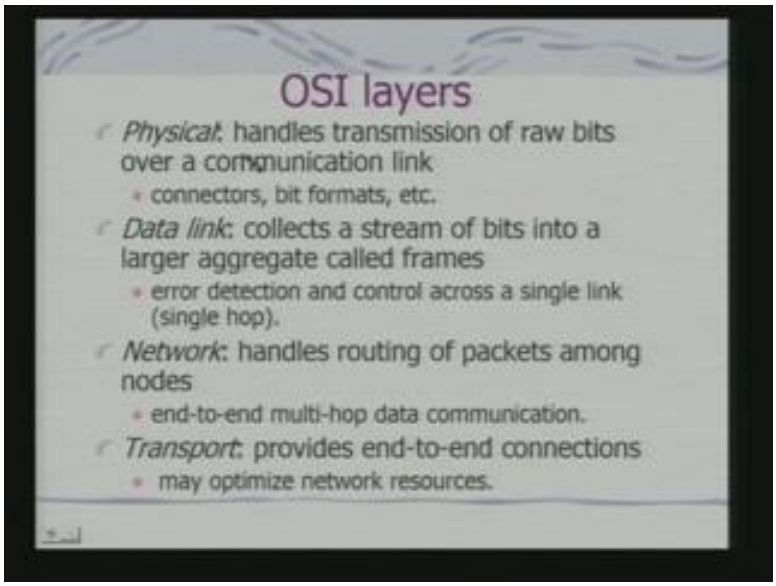
Now, networks; obviously, of therefore, complex systems. So, we need some mechanism or abstraction the most common abstraction is the 1 which was propose by ISO or known as open source interconnections model or OSI model and that is the seven layer model. And it provides a standard way to classify network components and operations.

(Refer Slide Time: 05:57)



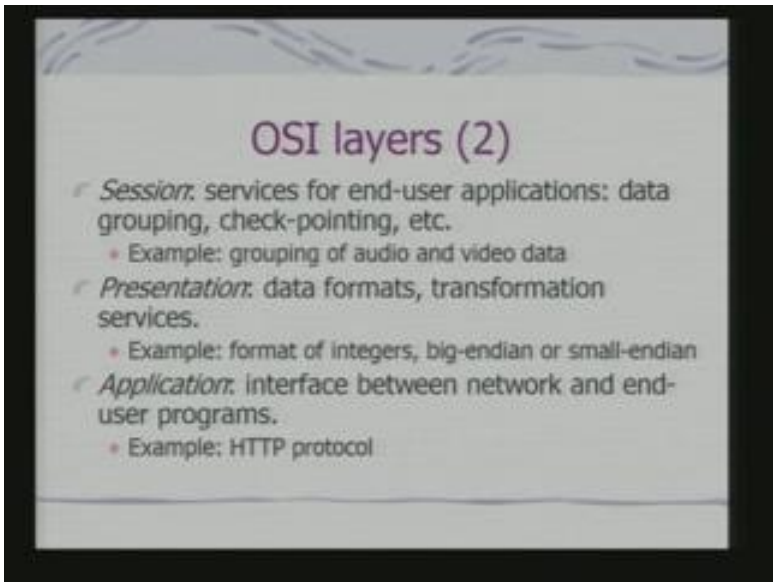
So, the basic model consist of what you say the physical layer which talks about the mechanical and the electrical signals that would basically become unique among the process elements. Data link layer which enables reliable data transport error correction can be part of data link layer. Network provides end to end service your transport layers actually provide from end to end connections that is from source to destination is provided through transport layer. You have the session layer which is on top of the transport connections managing the applications dialog. You have got presentation layer which is more concern with data formats being it changed and top you got an application which enable communication between end user applications. So, these are the 7 layers as part of OSI model. So, several networks in a network protocols that we shall look at they will be implementing some parts of the layer it is not necessary. They will implement all of these layers and depending on the layers that be implement the applications to be built on top of them.

(Refer Slide Time: 07:11)



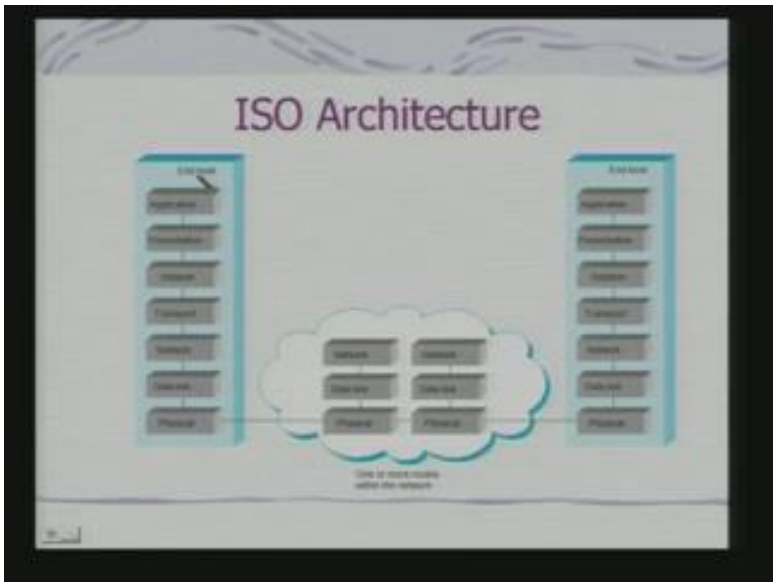
So, going back to the OSI layer definitions what we say the physical layer handles transmission of raw bits over a communication link. Data links layer collects a stream of bits in to a larger aggregate called frames. And error detection and control across a single link is typically the management of data link. So, it is what we say a single hop because the data can go from 1 point to another point that is source to destination via multiple hops. Network layer handles routing of packets among nodes and it therefore, manages end to end multi hop data communication. Transport layer provides end to end connections, that is, it maintains connections between your source and destination. Independent of the fact how the network layer routes a data from 1 point to another point and it would look also optimizing the network resources.

(Refer Slide Time: 08:12)



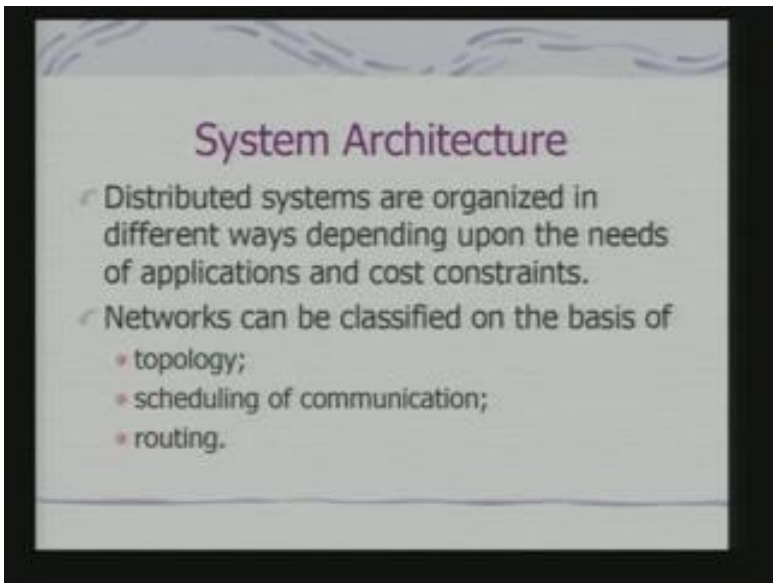
Session layers provides services for end user applications data grouping check pointing if there is an error it can identify that error through some synchronization points. And so that the there can be re synchronization of the data. Say for example, when we talk about grouping of an audio and video data audio and video data in a teleconferencing. Or the video conferencing application may be transported over 2 distinct transport links. And the session layers responsibilities to group the data and present it to the user appropriate layer. The presentation layer is concern with data formats and transformation services say for example, formats of integers can be different for different machines 1 machine can be big Indian other machine can be small Indian. How to such machines communicate? So, the interpretation of data is consistent that is taken care of other presentation layer. And application is interface between network and end user programmers. The most common application layer the protocol to use even in the context of embedded systems is your http protocol hyper text transfer protocol.

(Refer Slide Time: 09:28)



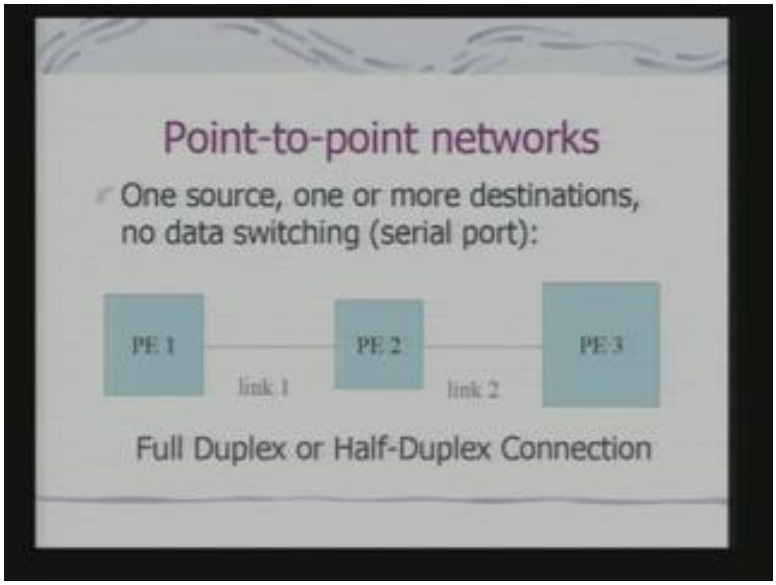
And the architecture if you look in to for communication this may be an enhost this 2 are the enhost and it goes through a network. So, this network layer this layers that network data link and physical link layers can provide the communication path. And this transport layer maintains actually this end to end connection and the session. And session presentation application provides a top layer services for this kind of network communication. Obviously, this is a complete conceptual model actual implementations would have specification of definite protocols.

(Refer Slide Time: 10:05)



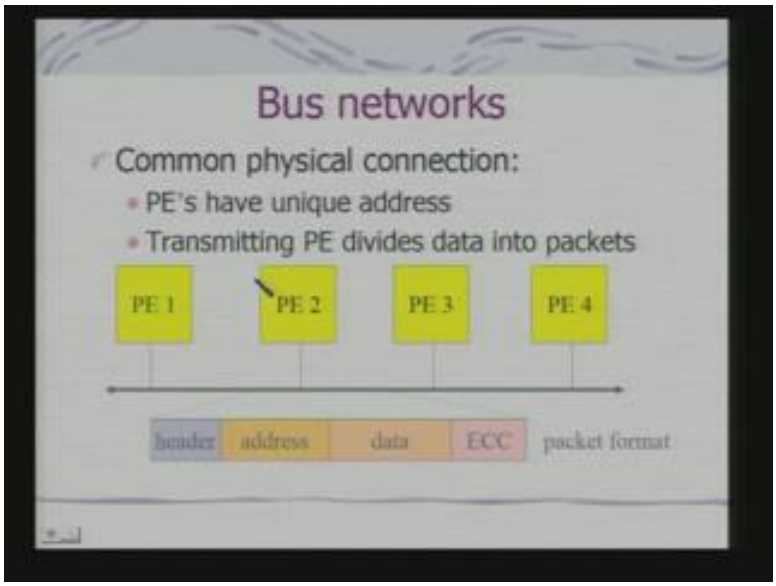
So, what is the basic system architecture distributed systems? And therefore, organized in different ways depending up on needs and the applications as well as cost constraints and this networks. Although we talk about this is a basic protocol model the basic protocol abstraction model network can be classified scheduling of communication topology that they follow and the routing strategy that are adopted.

(Refer Slide Time: 10:35)



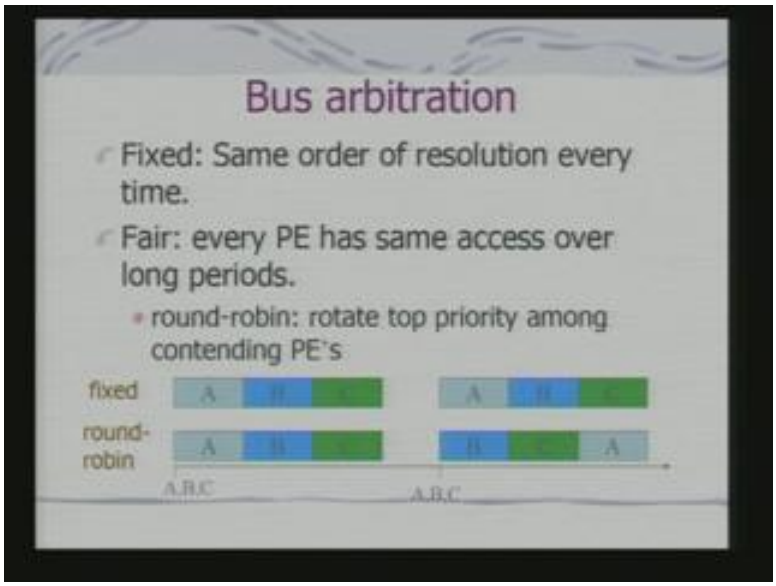
So, the simplest network that we can have is a point to point networks there is 1 source and 1 destination and there is no data switching. So, if you a built a building a network over the serial port between the PEs we shall have this a point to point network connection it can be a full duplex or half duplex communication.

(Refer Slide Time: 11:00)



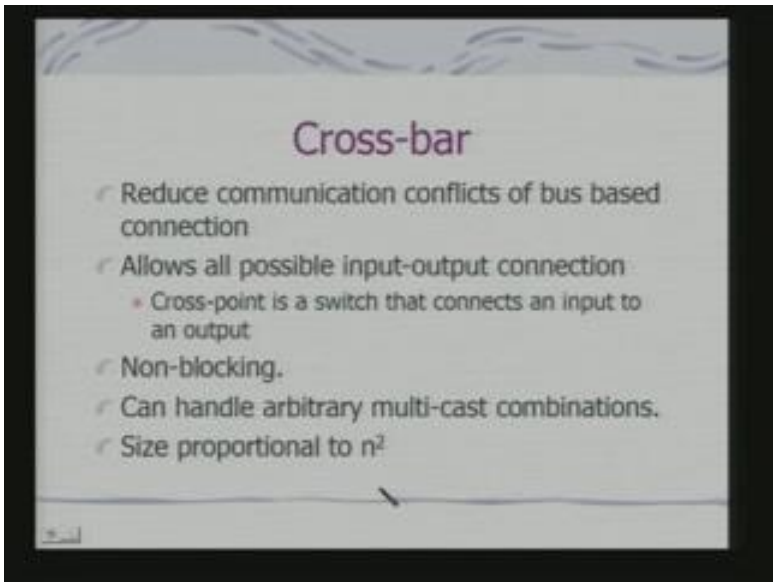
The other thing is that bus we have already studied bus base communication, but here what we are talking about? A bus PEs network so; obviously, the basic distinction with the other buses that we have considered so far would be that you we shall not be expecting in a shared memory lying on the bus or connected to the bus to which the PEs is communicated. So, there are multiple processing elements which are connected on the bus and the data which is being transmitted a typically divided in to packets. So, we will have a typical packet format this packet format will have the header address data and basically a error correction code. Now, this address would tell possibly PE whether this data is meant for it or not. So, effectively the data would flow on the bus the bus is common among the processing element. And depending on the address a processor would know whether it is a attended for it or not. Obviously, in such cases arbitration just like in any other bus like a typical processor bus the arbitration here also is of importance.

(Refer Slide Time: 12:18)



So, you can have a similarly fixed arbitration same order of resolution every time or you may have a round robin if it is a round robin if the fair because every PE has same access over long periods. So, here is an example if I have a fix priority a 2 points always you will have A B C if the requesting the priority will be assign ABC. Now, if is a round robin since B has been in the sense the second priority the second time when there is this request of data transmission B would get the priority and A s priority would be last. So, here the basic point is at this point the both ABC that is all 3 processing elements are trying to transmit the data. There has to be an arbitrations came and that arbitrations came would resolve in favor of the priority. If it is a fix priority the highest priority systems always gets preference if it is a round robin the priority changes dynamically.

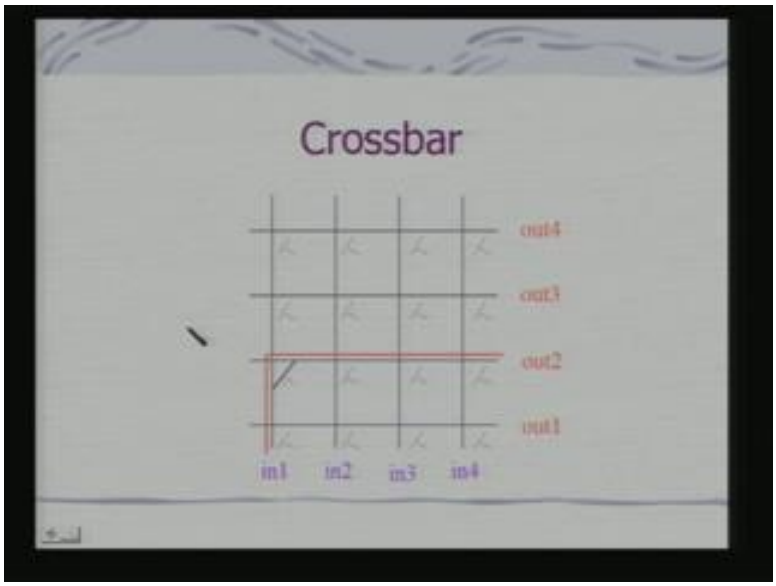
(Refer Slide Time: 13:19)



Now, how this; this is a bar a basically the bus connection. The other possibilities a cross bar cross bar enables all possible communication paths between in the processing elements. So, it reduces that why we say communication conflicts of bus based connection because in a bus based connection there can be always a conflict. So, we need a priority scheme to resolve this conflicts and cross bar allows all possible input output connections. And typically that network is built around a cross point which is a switch that connects an input to an output. And it is a non-blocking; that means, if the data is requested or the data is being send what happens in a non blocking case the data becomes available.

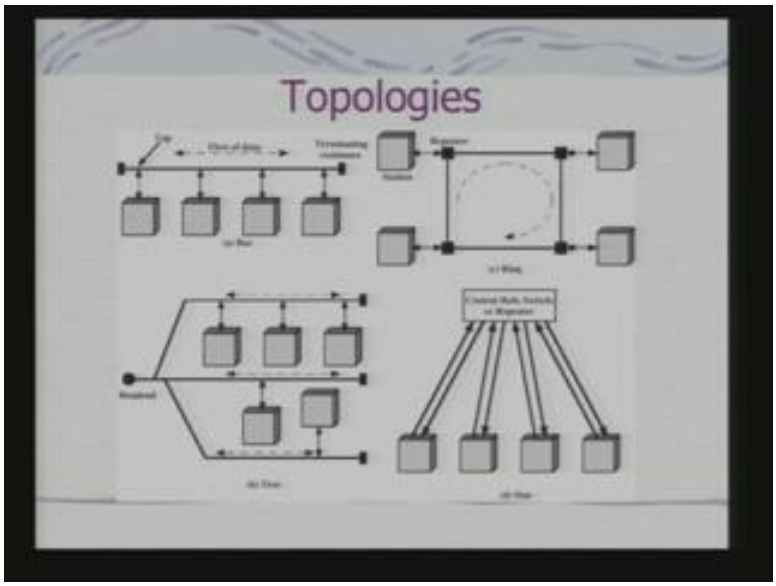
So, the processor is not waiting for getting a chance to transmit are getting a chance to receive and can handle the arbitrary multi cast combinations. If you look at the bus based configuration multi casting require what multi cast means the data meant for multiple processing elements. So, you have to use some special address if it is a broadcast there also you need to use some special address. Otherwise the processing elements would know whether the data is meant for it or not. But here the combination is such you can have depending on the connection pattern we can have multi cast as well as broadcast communication. And the size is proportional to n square why n square it will be clear if will see the basic conceptual nose.

(Refer Slide Time: 14:57)



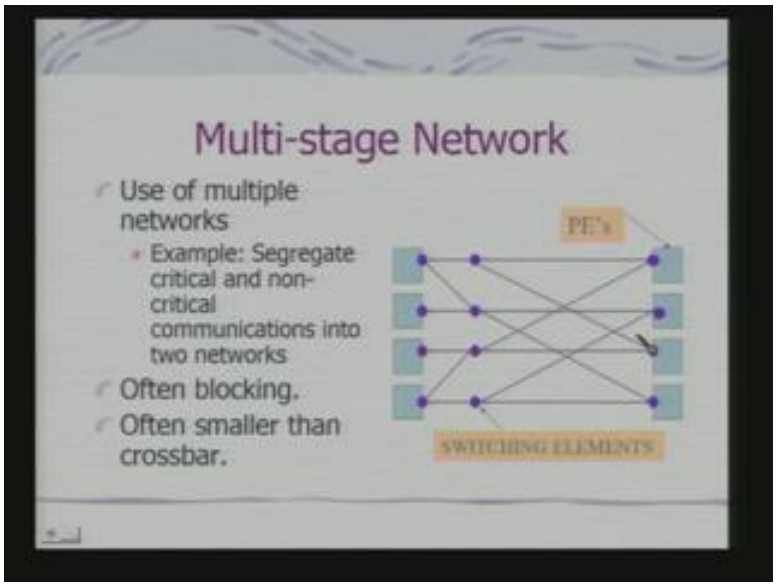
So, here if this are all if you see in 1 in 2 in 3 in 4 and out 1 out 2 out 4 this are input and output lines. And this is the cross switch actually it is a switch and this switch can you can realize that can establish can connection between any input point any output point and there can be multiple connection simultaneously being setup. So, we put this switch on effectively I get this input 1 connected to output 2 and this input and output are what basically the ports the connection ports on the different processing elements. So, you can have any possible combination of input output. So, in fact, data from processing elements processing elements 1 so, data can come to processing element 1 from here is a processing element 2. But at the same time if I enable this switches the data from processing element 2 can go to both processing element 1 may be not 2, 2 is the part of the processing element 2 itself to 3 as well as to 4. Obviously, the complexity here is n square in many cases actually if you are looking at a cross based a implementation with multiple processes. If the demand, if there is a demand of this kind of communication you will find this cross bar switches being implemented in the silicon itself.

(Refer Slide Time: 16:25)



The other standard topologies are 1 is this is a typically a ring. So, the processing elements are connected wiring this we say a repeater; that means, once it gets data from the station the repeater basically reinforces the data. So, that it can be moving around in the network. The basic different in here with respect to the bars is just in terms of the connectivity. Otherwise here also all this station is listing to a account common link this is the tree structure connection where you have the different links in fact multiple buses and ping managed through the head and this is a star link. So, you have a central hub are a switch are a repeater to which individuals stations are connected. Now; obviously, you can understand this hub fails the networks becomes completely disconnected. Now, these hubs themselves to be satisfy the basic conditional switching packet. So, the data from 1 source to another destination in a pretty first way and in fact if you your crossbar switch cross bar switch is a use it here I get therefore, all possible n square connectivity between the nodes.

(Refer Slide Time: 17:48)

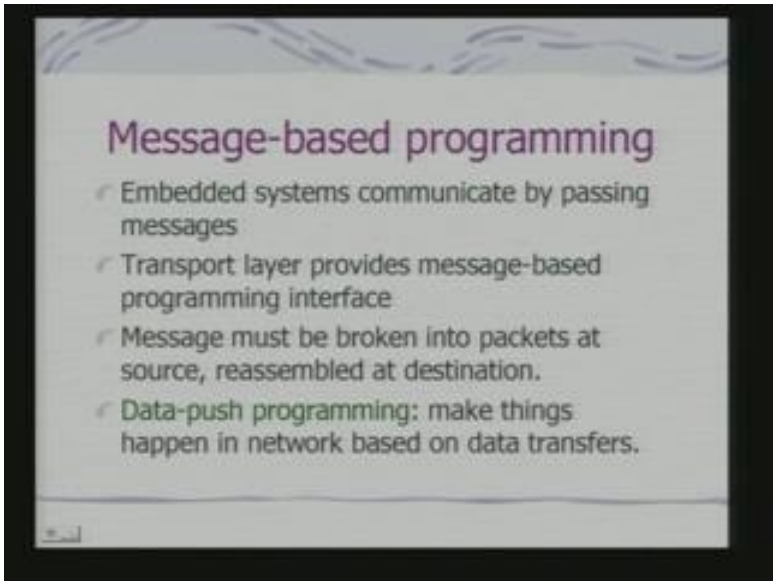


There are also multi stage network because networks can have different capabilities. So, if you have the networks different capabilities and that is also application dependent, because you may like to segregate critical and non critical communications tasks in to 2 distinct network. So, if it is in to a 2 distinct networks you may use different switching element in the network base itself this the basic network connectivity that connectivity matrix. So, you can use different kinds of switching elements and these switching elements actually enable communication if required between the parts of the network in that is between individual processing elements are between the complex networks. So, the basic difference cross bar that you find is; obviously, it is much less because all possible connectivity is not being provided. You have providing selective connectivity using selective use of their switching elements.

So, effectively get among multi stage network and these networks can operate for different requirements operate to meet different requirement. There is a critical communication requirement between this 2 processing elements. So, this is all connected wire is switching element and that is not true with respect to this. So, these processing element is not connected through this. So, if at all the connections has to come from here t may have to take a complex path. And the time required for a data to arrive from this processor to this processor would be more than between this processor. So, effectively I am talking care of the need of the communication tasks and accordingly using the switching elements. Now; obviously, in this kind of networks you do not have a shared

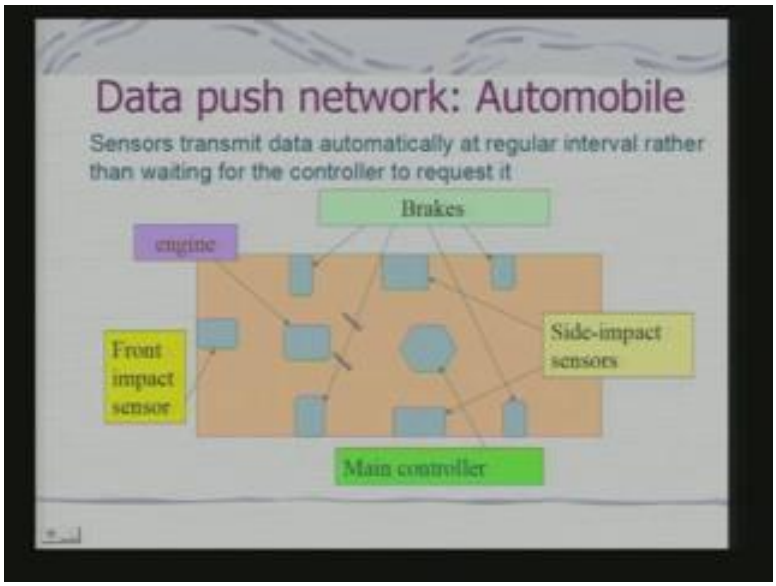
memory how do the tasks communicate with each other the basic way for such communication would be messages.

(Refer Slide Time: 19:54)



So, embedded system in such cases would communicate by passing messages. And the transport layer in the OSI model provides the basic message based programming interface. And typically a message must be broken in to packets at the source and we have assembled at a destination. So, this is the basic feature of a packet network and each packet can follow a different path. And what we say that basic programming model wise that this is data push programming make things happen in network based on data transfers that is when you get a data from a sensor then some action has to be initiated. So, that becomes what we called data push programming style of application development.

(Refer Slide Time: 20:48)



Let us take an example. So, this is example of a car. Now, the car can have a multiple sensors car can have multiple actuators. So, if you see here what we have shown there is a front impact sensor there is an accident front impact sensor. You have got the breaks which are actually you remember you had actually discussed a break controls system. So, this breaks there can be controlled with regard to this controller. Then you have got the side impact sensors you have the main controller and you have got also the engine which has been also controlled by micro controller. So, you will find that so many sensors and so many processing elements existing in this car and they need to be connected together.

So, what is the base to yet connecting together could be a network and in that such a situation you will find that these sensors would transmitted automatically at regular interval rather than the controller requesting for it. Because if I have to take some action it would be on the basis of what happens? So, if you look at a take a look at extreme case. So, if there is an impact. So, front impact sensor would send, because there is a p attach to the sensor as well that will send the data to the controller. Controller in turn will trigger the engine micro controller to stop. So, it is not a request transfer, but it is an example of a data push programming that is whenever something is happening the message is being passed and message is triggering the controller to take action.

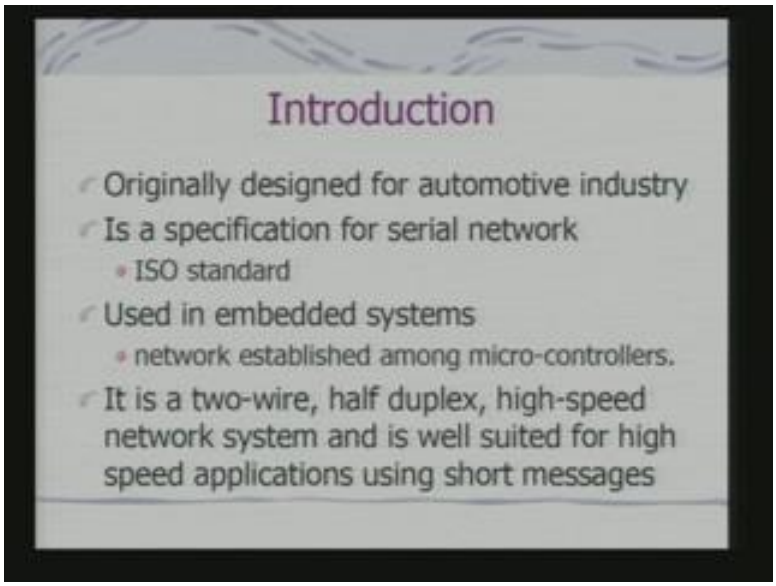
So, the whole style of programming for the application development in such a scenario would be different. Now, what we need to look at this if such a network has to be built

do you need to have some special requirements? Now; obviously, you can understand if you are looking at this kind of appliances say a car where you would like to have such multiple processing elements to be network we may not need truly we do not need all seven layers OSI model to implemented. So, what we shall have? We shall have a subset of such a model with special emphasis on the requirements of such a system. So, such protocols a primarily intended for this kind of embedded applications. So, one such protocol we shall look at now control area network protocol.

(Refer Slide Time: 23:30)

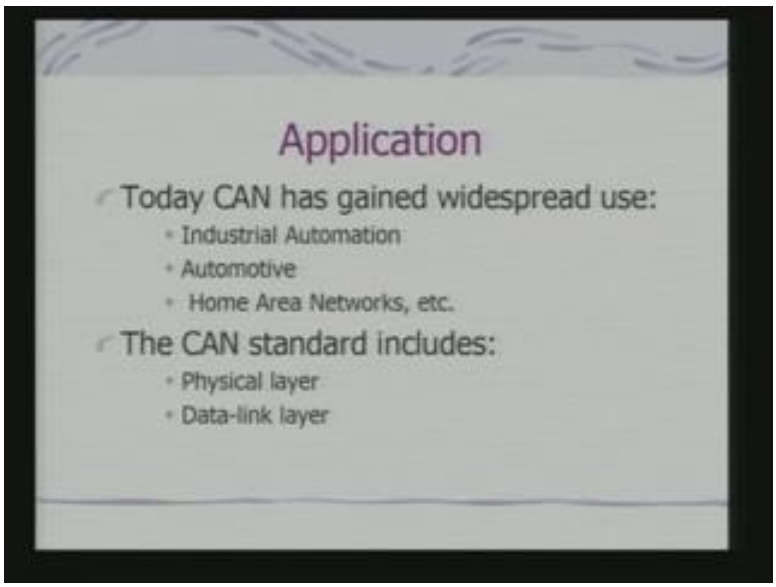
In fact this control area network protocol was developed for catering to requirements of automobiles example that we had just seen.

(Refer Slide Time: 23:41)



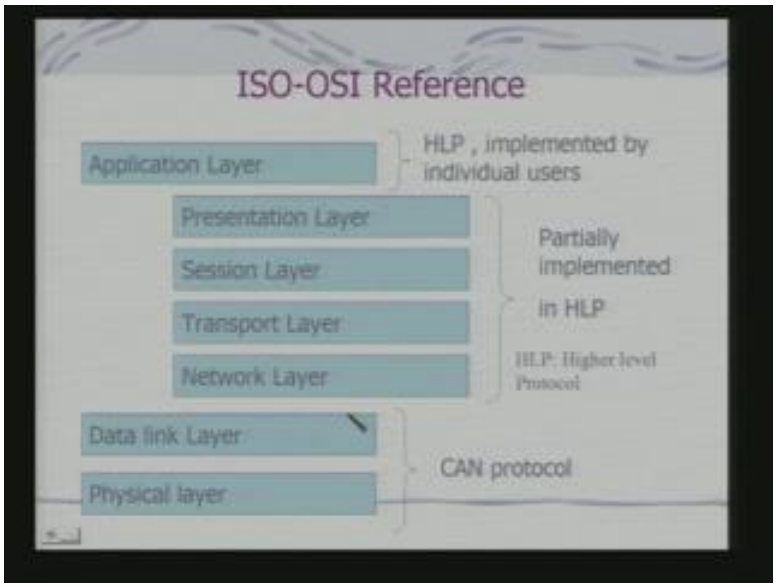
So, it is a specification for serial network; that means, basic communication is a serial link based communication. It is an ISO standard and used an embedded system for the network established among micro controllers then have seen in a car there will be micro controllers associated with the various sensors as well as the main controller. So, how to network such micro controllers? It is a 2 wire half duplex high speed network systems and is well suited for high speed applications using short messages and not really long data flows you are not expecting a continuous long stream of data being exchanged.

(Refer Slide Time: 24:27)



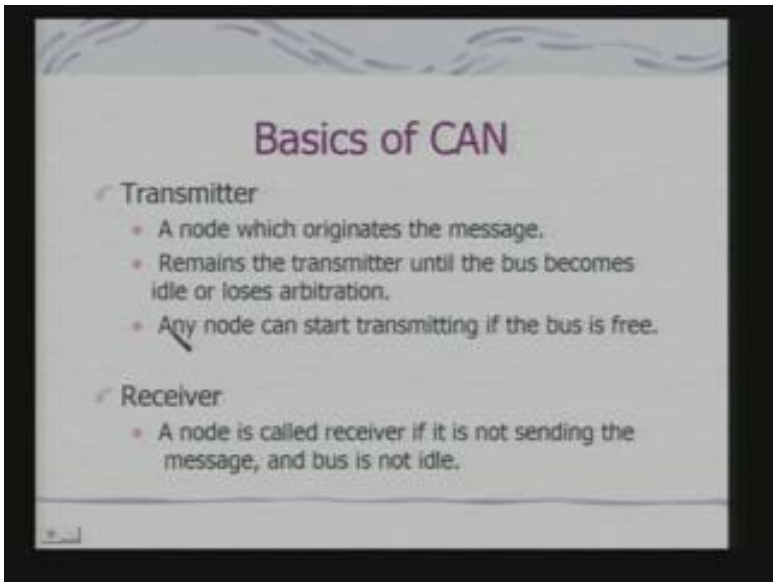
So, CAN is used we shall see electron also in more applications industrial automation automotive that is a car home area networks and CAN standard basically specifies physical layer and data link layer.

(Refer Slide Time: 24:45)



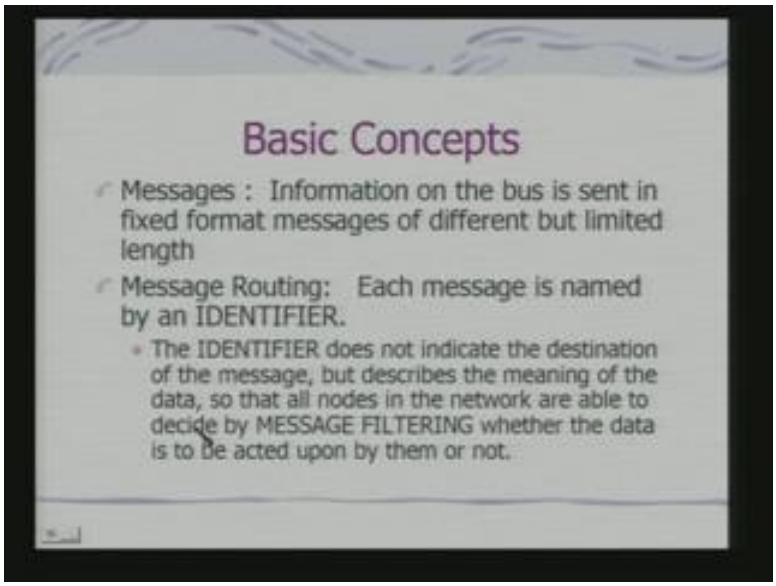
So, if we look in to it. So, these parts are define by the CAB protocol and theses are part of high level protocol which may be implemented by a particular manufactured. Say for example, if I have these 2 specifications available with me. So, what happens? I can use components as well as CAN controllers from variety of sources, because they will be all compactable with these 2 specifications. Today, you can buy a Ethernet card for your PC from a number of vendors, because all of them are consistent with respect to that standard. Similarly, you can have therefore, CAN protocol consistent components for these kind of applications and say a particular car manufacturer. They can have a kind of higher level protocol is implemented on top of this 2 layers and then it not implement in all this layers. They would implement some of the functionality that are required further. Now, these model is something which is important for you to understand when you are talking about networking embedded systems you really do not need always the full flagged functionality of always earlier. Wherever the functionality is needed you need to exploit them this functionality may be applications specific and some part of the functionality can be standard spaced. So, here for CAN data link and physical layer are the specification of the standard remaining can be purely application list.

(Refer Slide Time: 26:27)



So, basically; obviously, if we are looking at this kind of the network there would be transmitter and there would be a receiver. Transmitter is a node which originates the message because this primarily sending messages and that exactly the model that we have looked at for implementation of such a system. And the message remains the any node can start transmitting the message if the bus is free. So, the transmitter rates until bus becomes idle or loses the arbitration a receiver a node is called receiver if it is not sending the message and the bus is not idle. So, in fact in a way you can receive that each node can take up the functional role for transmitter or the receiver depending on the need.

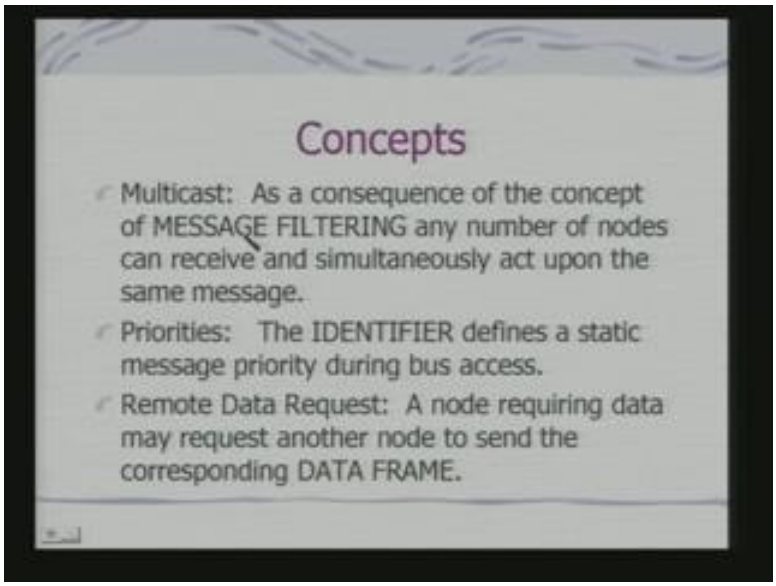
(Refer Slide Time: 27:18)



The messages are nothing, but information on the bus that is sent and they are typically fixed format messages are different, but limited length. Now; obviously, if you remember when you say data link layer protocol the data link; the basic task of data link is to define this kind of frames. So, that data link layer functionality is coming in when we are defining a structure of such messages and there would be a message routing because messages are meant for a different processing elements. So, each message is named by an identifier the identifier does not indicate the destination of the message, but describes the mine of the data.

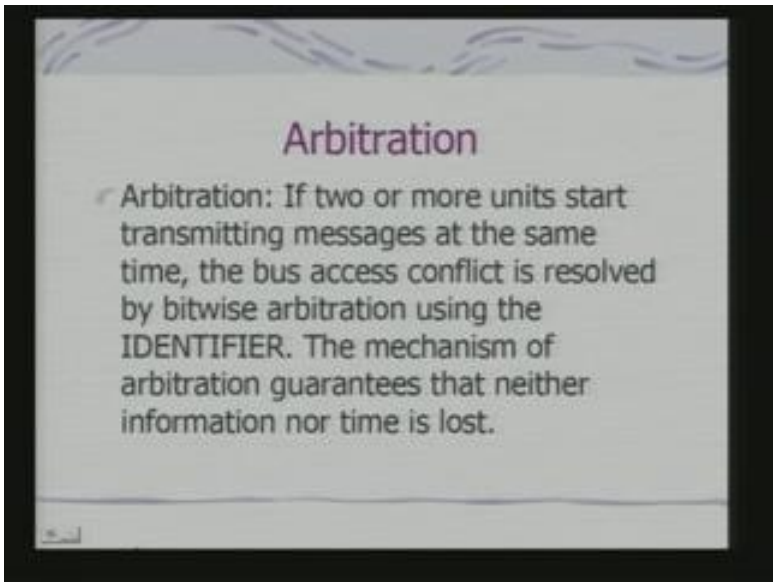
So, in the all nodes in the network is able to decide by message filtering whether the data is to be acted upon by them or not say effectively if you see what we are talking about is that a connectivity is structured where all nodes are listening to the messages being transmitted on the link. So, it is similar to your typical bus base network all of them are listening to data which is being transmitted to the link and depending on the identifier a node decides whether some action has to be taken or not. So, automatically your multicasting broad casting is built in to this kind of scheme. Because it is not just based on the address it is based on the identifier which actually indicates the minimum of the data based on which and node decides to act.

(Refer Slide Time: 29:01)



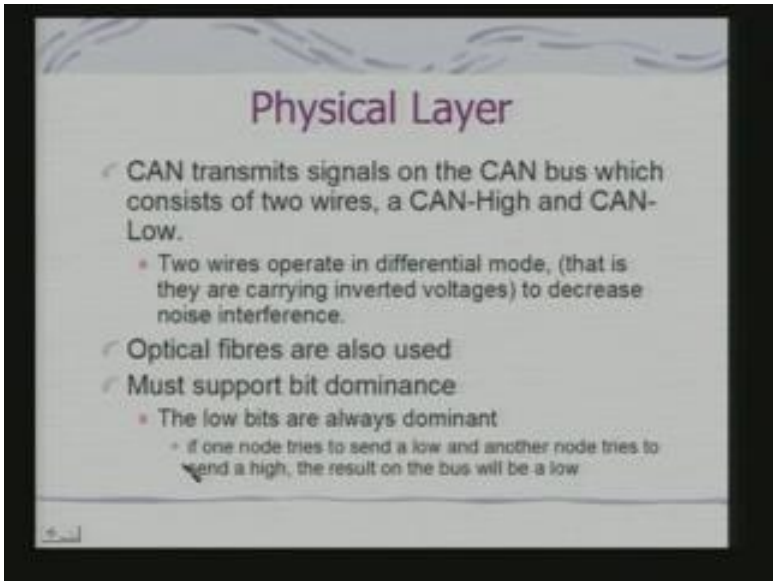
So, multicast is a consequence of message filtering any number of nodes can receive and simultaneously act upon the same message this is the most important feature. So, multicasting becomes automatic and; obviously, I need a priority because if all of them are listening there would be always conflict if more than 1 node is trying to transmit data. So, I need a mechanism to reserve this conflict and here again the priorities are used the identify defines the static message priority during bus access there are also mechanism for requesting for data. A node requiring data may request another node to send the corresponding data frame. So, if you see that there are 2 kinds of messages actually you are talking about 1 message is that is sending the data another message is that of requesting the data. So, a transmitter can actually send data are it can also send a request for data arbitration occurs.

(Refer Slide Time: 30:08)



Two or more units start transmitting message of the same time and it is a resolved by using the identifier priority the mechanism of arbitration guarantees that the neither information nor time is strictly lost. So, that arbitration mechanism is an interesting part of CAN and we shall discuss it later on first let us start looking at the physical layer.

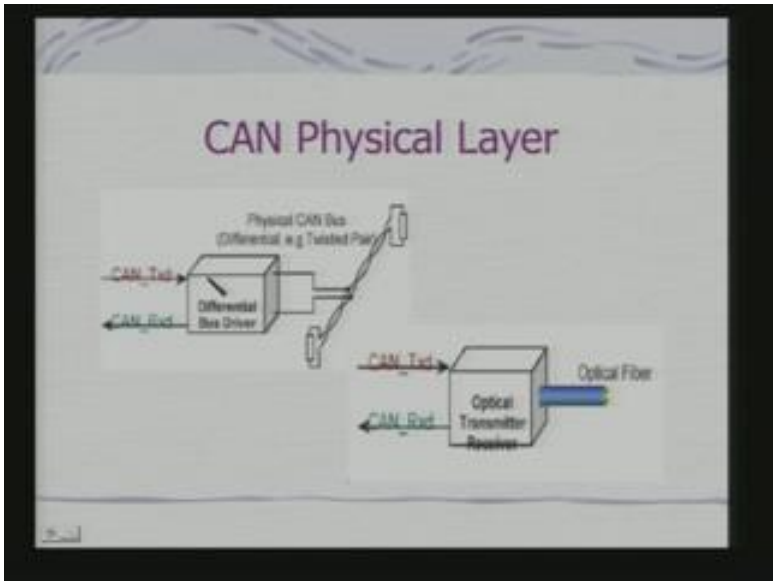
(Refer Slide Time: 30:29)



Physical layer is it suggest that transmission of the data in a differential mode that is it is a 2 wire CAN bus. So, we have what we called CAN high and CAN low lines and 2 wires operate in differential mode that is they are carrying inverted voltages. So, why

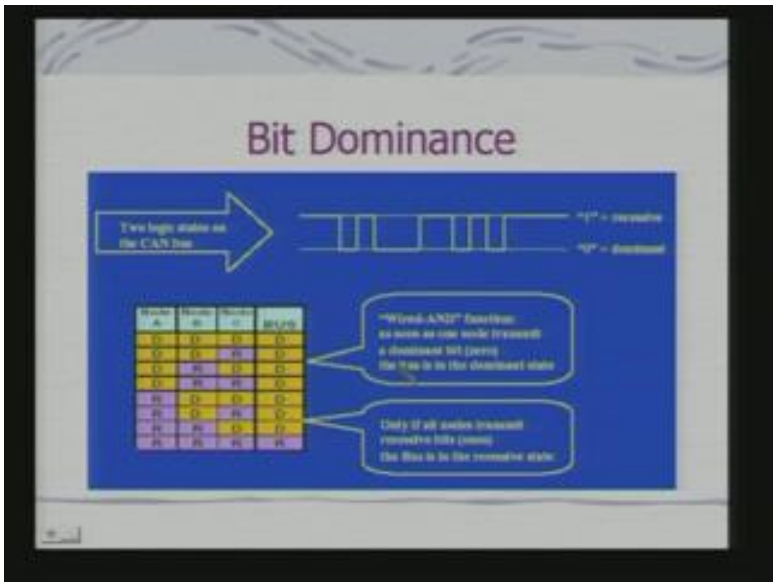
differential mode advantage is there will be a automatic noise cancellation if you are looking at an application scenario of that of the car which is highly noisy than that of the initial standard networking environment. So, if you have a differential mode then the noise cancellation becomes much better also optical fibers are also used which is naturally protected against his noise. The other interesting feature of the layer is much support what is called bit dominants and in this case the low bits are always dominant. That means, if 1 node tries to send a low and another node try to send the high result on the bus will be the low. Now, this is very very similar to your I to c bus configuration. So, CAN physical layer something like this?

(Refer Slide Time: 31:52)



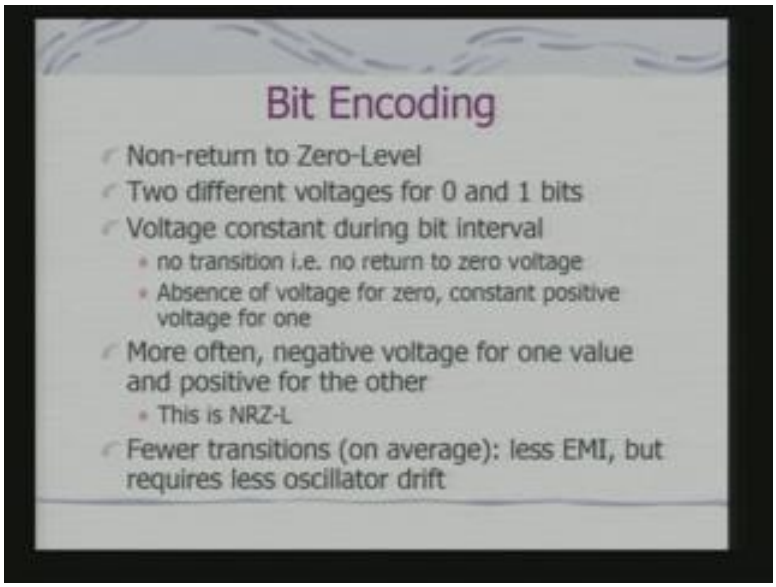
So, you have got the CAN transmitter and CAN receiver data lines is can go come from a CAN controller. This is your differential bus driver and that will be connected to a twisted pair, because I can pear cable for differential mode transmission. So, that will be connected to a twisted pair cable if I using an optical transmitter receiver that will be connected to a optical fiber. So, both these are being used with figure to the CAN and these lines communicate with the CAN controller.

(Refer Slide Time: 32:25)



And bit dominance basically tells you the basic combination. So, if it is effectively what a wired and function. So, the basic connection mechanism again be implemented through a simple wired.

(Refer Slide Time: 32:41)

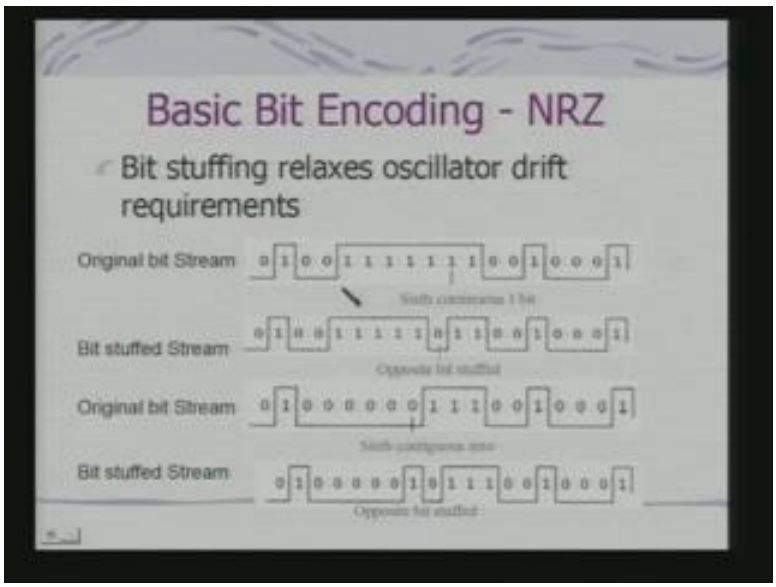


Next comes the bits encoding; at the bit level how the bit information is encoded typically what is used is non return to zero level this is the standard way of sending digital data non return to zero level. There are 2 different voltages for zero and 1 and voltage is constant during bit interval. So, no transition si that is no return to zero voltage

and absence of voltage for zero and constant positive for 1 there could be other way round also more often negative voltage for 1 value and positive for the other. So, they can be any way round, but what is the important is that there are no transition. So, the bit period now what is important to know this voltage constant during bit interval there is no transition involved during bit interval under that why it is non-return to zero level?

So, effectively you got fewer transitions for data transfer. So, less electromagnetic interference issues would coming, because any transition means what you generate basically all possible electromagnetic radiations. There could be interference problems the moment such transition are minimized EMI problems are minimized. But it requires less oscillator drift why because your bit period has to be strictly monitored. Because if the bit period is loss if there is a oscillator drift if the bit period changes the interpretation of data can become erroneous. So, if you look in to the structure what we are talking about is say if this is your original bit stream.

(Refer Slide Time: 34:30)

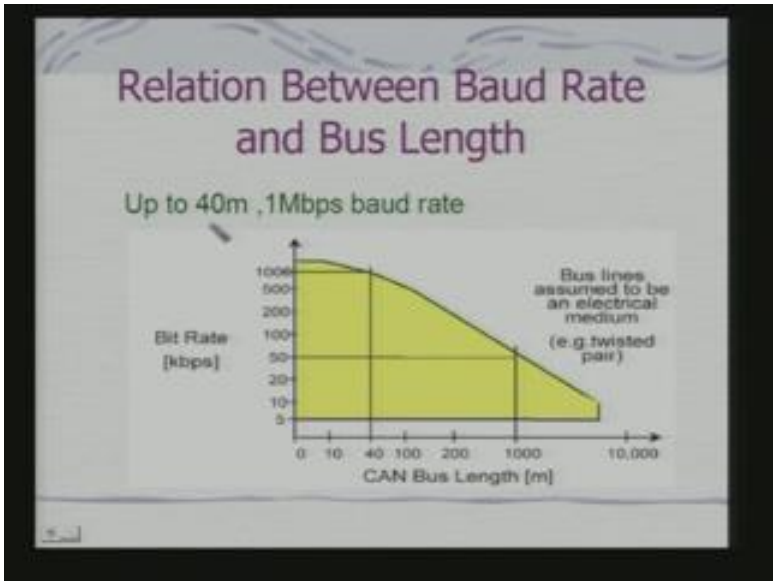


So, effectively if there are 2 zeros there are no transitions in between. So, there are constant voltage level looking at zero bits low voltage and once is the continuous 6 continuous 6 once it is a continuous voltage level. So, since there are no transitions during bit interval; this is non-return to zero. Now; obviously, what it requires they cannot be a clock drift from processing element to processing element if there is a clock drift then interpretation of this intervals bit intervals can become erroneous. So, instead

of counting six ones I may count five ones. So, then there will be wrong data. So, how to minimize this problem of clock drift?

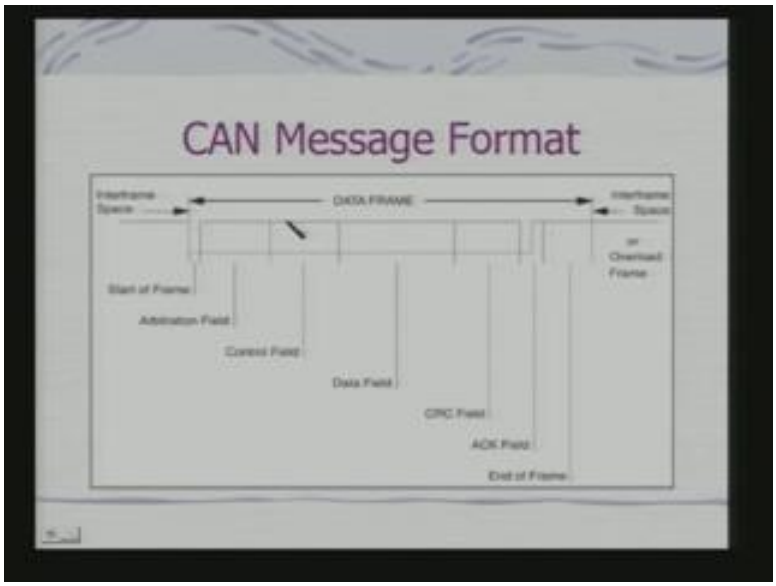
So, what you do says is bit stuffed stream what it does is reduces the convention and at regular intervals it puts in a opposite bit which is stuffed in it is put in between. So, what will see here is that here instead of continuous 6 bits what you have got this five ones and there is 1 0 bit has been stuffed in. So, that the clock can be synchronize if required because this transition has to be noted similarly there would be opposite bit stuffed in a case of continuous zeros. So, effectively what your CAN users it is non- return to zero and bit stuff in to prevent the problem of clock drift and wrong interpretation of bit interval. Typically the feature is that if you look at the baud rate.

(Refer Slide Time: 36:18)



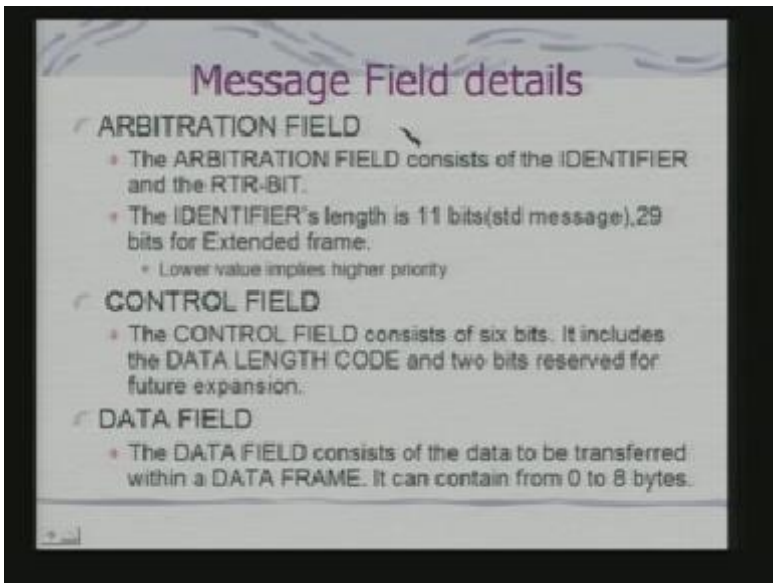
So, up to forty meters it can work at typically 1 Mbps as the baud rate. So, this is typically the characteristics of a CAN. So, 40 meters what you can realize that if you are looking that an appliances like car are any such this 40 meter is good enough. And so, I can have a high speed data transfer inside an application like car.

(Refer Slide Time: 36:44)



This is the basic message format in fact, these format basically is somewhat it gives you the features of data link layer not just a physical layer. So, there is a data frame. So, you have got of typically start up frame is arbitration field control fields, data field, CRC field take care of a error reject acknowledgement and a recent end of frame this a typical CAN message format.

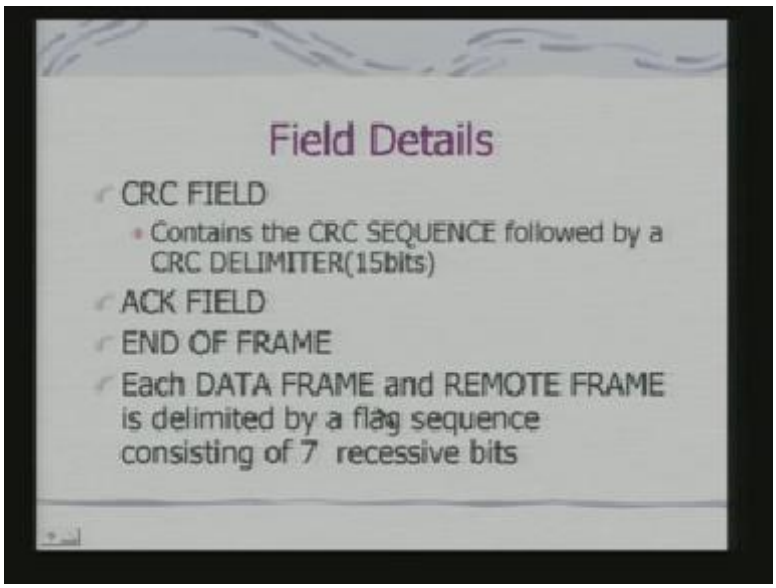
(Refer Slide Time: 37:11)



So, arbitration field actually consist of what is called the identifier and RTR bit. In fact, RTR bit indicate whether it is a data request frame or a actually a data frame the

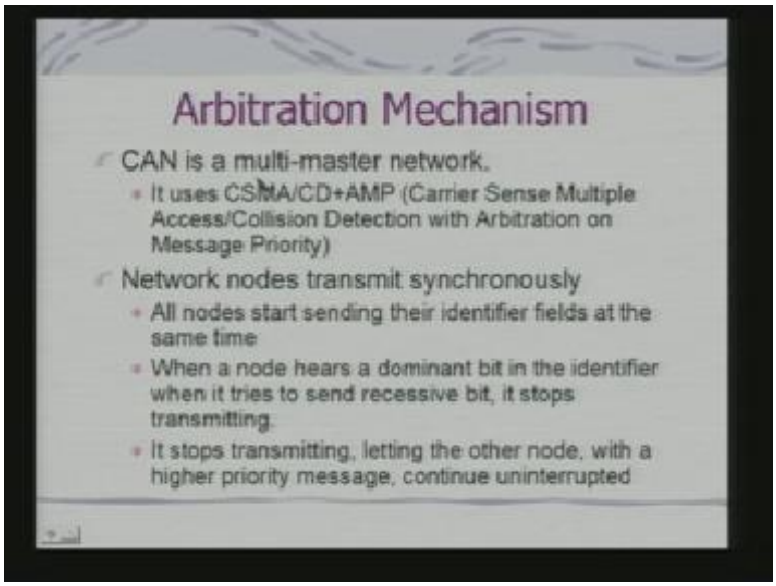
identifiers length is 11 bits. And there is a another version which has got 29 bits for extended frame and in fact these encodes also the priority this 11 bits are encode the priority. So, lower value implies higher priority for this field and this will be very interestingly it for arbitration. The control field consist of 6 bits includes data length code and 2 bits reserved for expansion data length code tells you how many, the how many data bytes are part of this message? And data field contains actually the data bytes and there can be maximum of 8 such bytes.

(Refer Slide Time: 38:07)



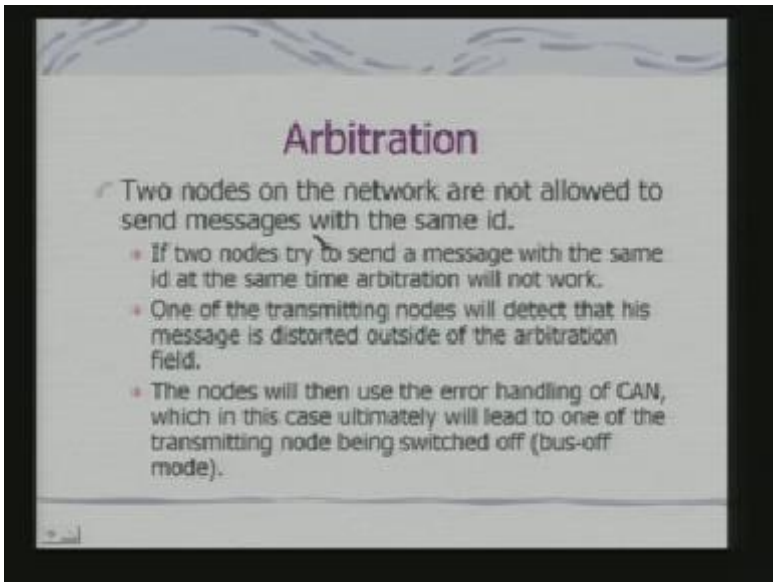
CRC field is there acknowledge field there is end of frame and each data frame and remote frame. Remote frame is actually request for the data frame is delimited for a flag sequence consisting of 7 recessive bits seven bits to which 1 is a recessive bit in this case. In this case high would be the recessive bit because there is active. So, 7 recessive bit would be put in 2 separate out the individual data frames why this is required, because next frame transmission has to start again at the end of this recessive period. So, that is what is first separating the message frames now we come to the arbitration mechanism.

(Refer Slide Time: 38:55)



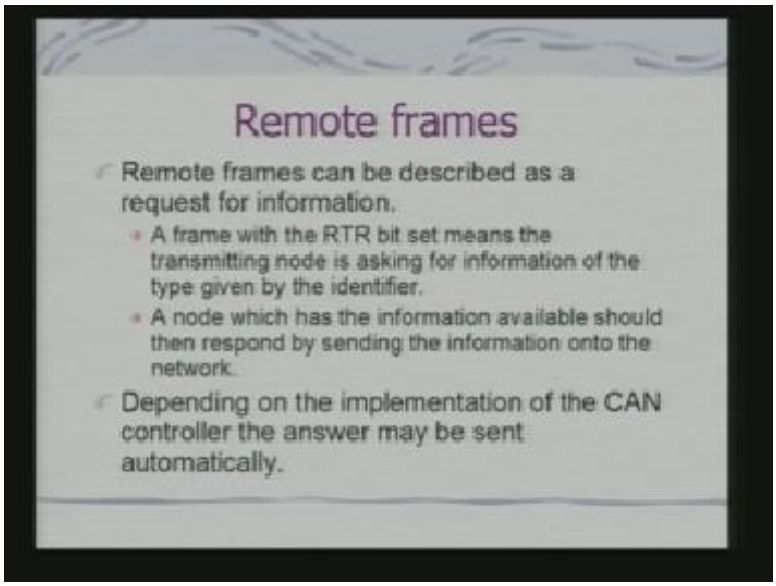
In fact, CAN is a multi master network and what we say it uses carrier sense multiple access, collision detection with arbitration on message priority. This is the basic protocol CSMA CD plus AMP and this AMP basically distinguishes this network from your well known Ethernet. Now, all network nodes transmit synchronously. So, all nodes start sending their identifier fields at the same time it is a synchronous at the same time. So, when a node hears a dominant bit in the identifier when it tries to send recessive bit it stops transmitting. So, it is stops transmitting letting the other node with a higher priority message continues uninterrupted and these becomes automatically resolve with regard to the priority why because the lower value will have higher priority. So, if you have the zero over there that is a dominant bit so; obviously, the dominant bit will be taken precedence over the recessive bit. So, if I have a message with a higher priority that message will be transmitted. So, the collision gets automatically resolved on the basis of message priority this is something which is very very interesting with CAN. So, 2 nodes on the network and therefore, not allowed to send messages with the same id.

(Refer Slide Time: 40:27)



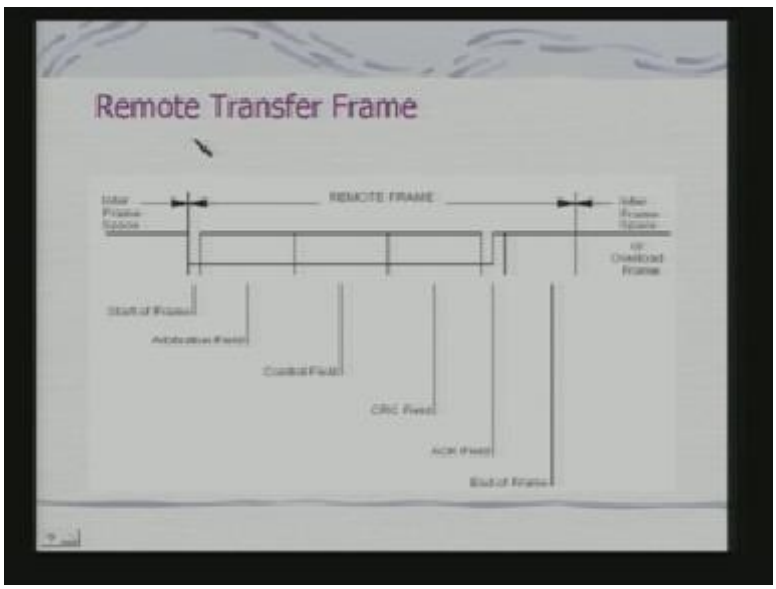
This is the another issue because this identifiers are what identifiers are not link to the node identifiers link to the meaning of the message. So, if the 2 nodes tries to associate the same identifier value they are also the problem. So, the arbitration mechanism is address this problem as well if 2 nodes try to send the message if the same idea the same time; obviously, arbitration will not work. So, one of the transmitting nodes will detect that his message is distracted outside of the arbitration field because since they are sitting on the bus we can listen to the data that it is sending as well .The nodes will then use error handling of CAN which in this case ultimately will lead to one of the transmitting node being switched off. Because this error can only detecting the transmitting node only because since it is a connected on the same bus and each node is listening to whatever data is put on the network. So, transmitting node can find out that the data it is sending is not the data that it is receiving why because they are also the zero bit will be dominant bit. If the other node is sending as zero outside identifier field then the transmitting node will be detect in error and that error has to be handled through CAN error control mechanism.

(Refer Slide Time: 41:59)



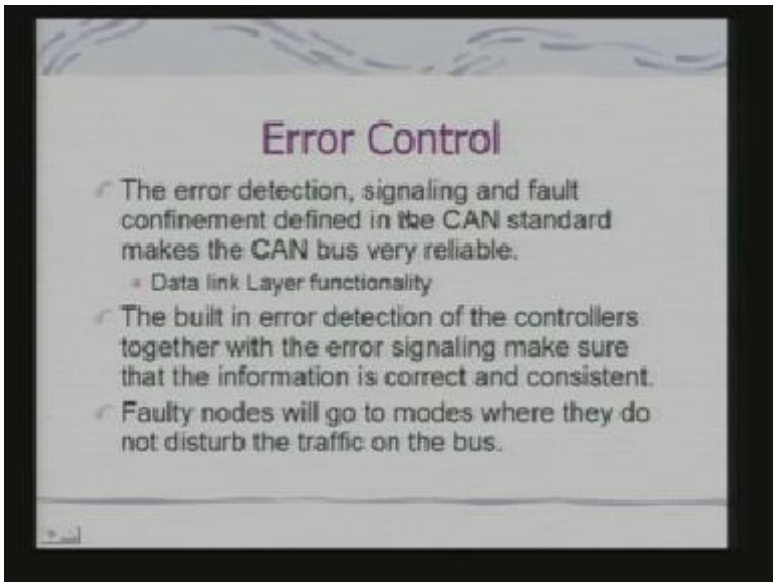
The remote frames are basically used for request of information. So, a frame with the RTR bit set means the transmitting node is asking for information of a particular type given by the identifier. Identifier indicates the data type and node which has the information available should the responding by sending information on to the network. So, this is also very interesting when a data is being asked for it is not being ask for with the node address it is being asked for with the type information. So, the nodes which have the data of the particular type can respond automatically depending on the implementation of the CAN controller the answer may be send automatically.

(Refer Slide Time: 42:47)



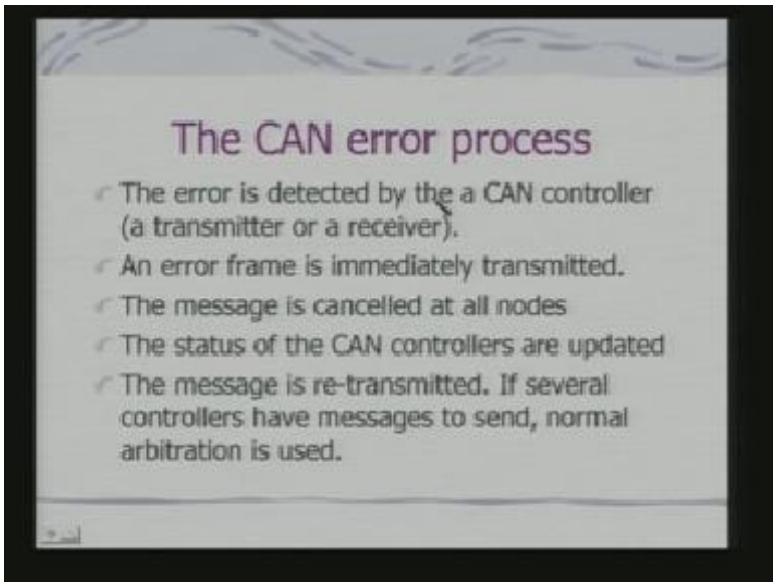
So, typically this would look like this it does not have a data frame it has got a control field it has got a arbitration field which is your identifier field and your acknowledgement field. So, this is your RTR; this RTR the remote transfer frame. He you really you do not have a data this is only the basic difference that is you have got.

(Refer Slide Time: 43:03)



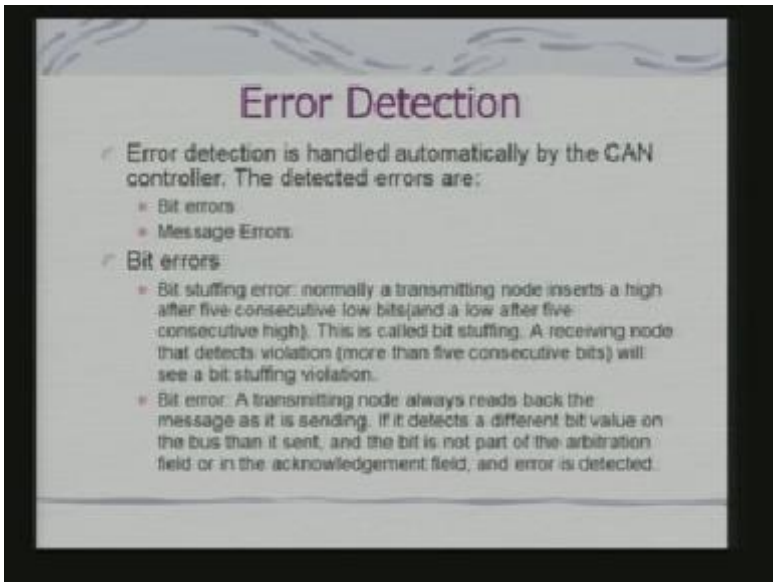
The next interesting feature is error control, because all data link layer protocol are expected to provide some kind of error management. The error detection signaling and fault confinement define in the CAN standard makes a CAN bus very reliable and this is a very typical of very data link layer functionality. The built in error detection of the controller together with error signaling make sure that the information is correct and consistent. And faulty nodes will go to modes where they do not disturb the traffic on the bus this is also very important issue. Because that node because all this sending on to the bus a node become it faulty that needs to be isolated to maintain the use correct use of the remaining part of the network.

(Refer Slide Time: 43:57)



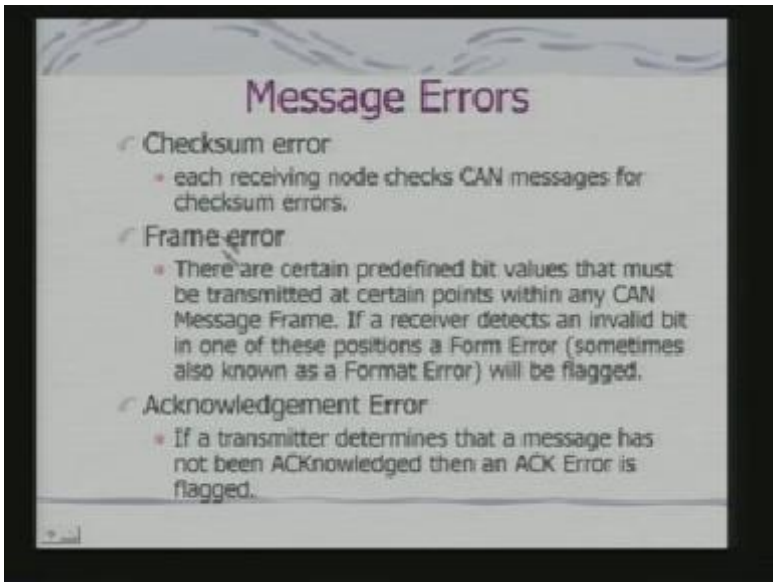
So, what is the error process say error is detected by the CAN controller which has function of the transmitter. Or as a receiver whenever it detects an error an error frame is immediately transmitted and the message is cancelled at all nodes. Any message you see been transmitted and status in updated and then the message is retransmitted if several controllers have messages to send normal arbitration is used. So, try to understand scenario scenario is the movement a CAN controller detection error. It sends an error frame which is a third type of frame we have looked at basically data frame we have looked at remote data request frame now it is an error frame. So, when an error frame goes in none of the message is the transmitter. So, messages all nodes reframe from sending messages after sometime the message is retransmitted and when the message is retransmitted normal arbitration scheme is followed.

(Refer Slide Time: 44:58)



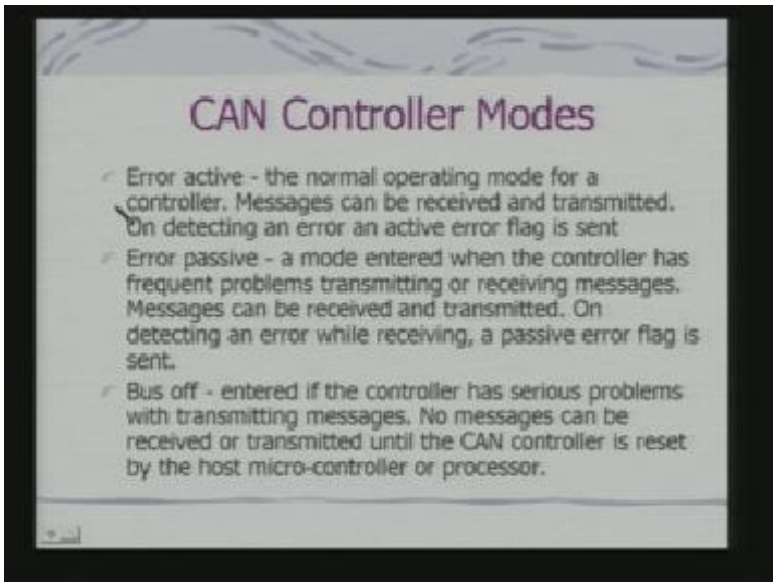
So, how what kind of errors can occur? Because this error detection is by the done by the CAN controller there could be 2 kinds of errors we say the bit errors and the message errors. Bit errors is typically a bit stuffing error I already shown you the bit stuffing format a receiving node that detects violation that is you have got more than 5 consecutive ones are more than 5 consecutive zeros will see a bit stuffing violation and it would flag the error. So, randomly some bit changes has taken place and a bit error is what transmitting node always reads back the messages as it is sending. If it detects the different bit values on the bus than it sent and the bit is not part of the arbitration. And that the bit is not part of the arbitration field all in acknowledgement field then the error is detected. So; that means, the error which is now on the bus has been changed for some reason. So, that error can be detected by the transmitting node only once it detects the error it had flag the error as well. Then there will be a message errors.

(Refer Slide Time: 46:10)



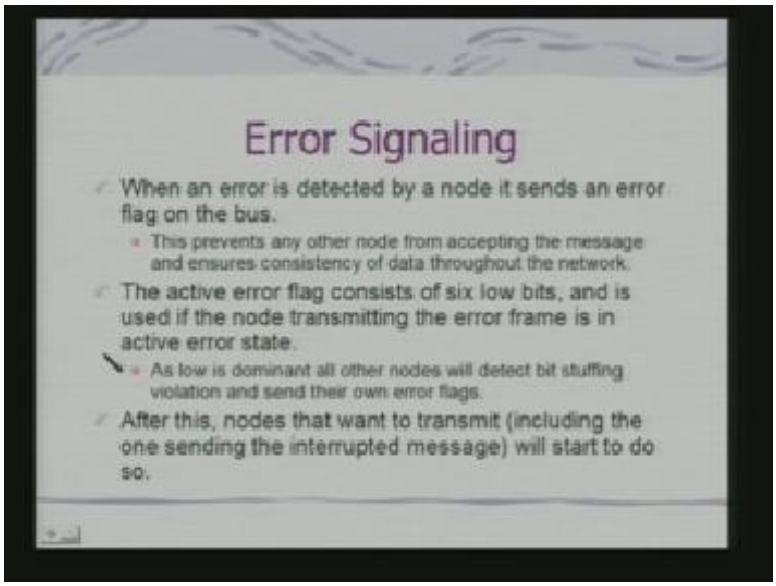
Because you already have the error correction code at part of the message frame. So, using that error can be detected take some error and frame error is with respect to some of the predefined bits. And if receiver detects an invalid bit in one of the positions a form error will be flagged. And a transmitter determines that a message has been acknowledged. Now, this is very very important if a transmitter determines the message has not been acknowledged. Then an ACK error is flagged because there is a provision for if you see a given a provision for an acknowledgement bit that acknowledgement bit has to arrive if it has not arrived in transmitted will flag an error. So, transmitter will decide to retransmit the message because it has not been acknowledged.

(Refer Slide Time: 47:02)



So, are the basis of this errors, the CAN controller can be in variety of nodes error active error passive bus off. Now, this is an active error flag messages can be received and transmitted and on detecting an error an active error flag is sent, because this a normal mode of operation there is an error it will just set an active error flag. What is the passive error? A mode this mode is entered when controller has frequent problems and if there is this problem on detecting an error while receiving a passive error flag is sent. And you will see that how this 2 different this 2 flags are different and how that helps in error management? Bus off is entered in the controller has serious problems with transmitting messages. So, no messages can be received or transmitted until the CAN controller is reset by the host micro controller or processor are if an manually. So, it is completely off the bus.

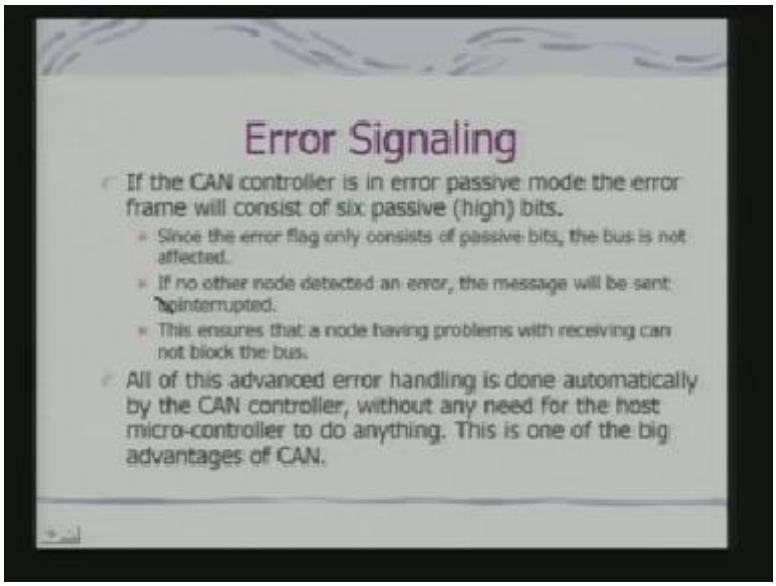
(Refer Slide Time: 47:58)



So, when an error is detected it sends an error flag on the bus that is the error frame. This prevents any other nodes that accepting the message and ensures consistency of data throughout the network. So, if the message is already there if it receives in a flag that message would be actually rejected the message has been transmitted. The active error flag consists of 6 low bits and is used if the nodes transmitting the error frame, is in active error state what is that mean? This low bit should be the dominant base as low is dominant all other nodes will detect bit stuffing violation and send their own error flags, because this is 6 low bits. So, it is valid in the bit stuff requirement you cannot have. So, once it detects the error they will send their won error flags.

After this the nodes are 1 to transmitting will start to do. So, that means, the data is basically being removed all of these all of this nodes will flag errors. So, just note how the error is being managed whenever an active flag error is transmitted that is will be indicated by what will be indicated by a transmitting nodes. For example, when it detects an error and then this 6 low bits if it is sent what will happen that the dominant bit. So, the receiving nodes will indicate a bit error and that will race again the error flag. So, all the nodes will be now realizing there is error in the bus. And therefore, they could ignore the data at the next instant, because it is a synchronous bus. So, the next instant will arrive for the transmission of the frame than the retransmission can take place here is also a passive error mode.

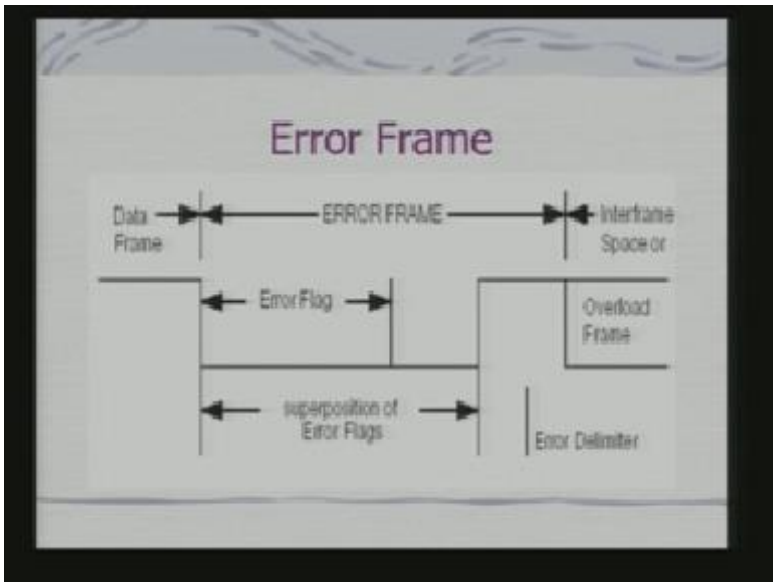
(Refer Slide Time: 49:55)



The CAN controller; so, in this case error flag only consist of passive bits. So, bus is not affected. So, if no other node detected an error then message will be sent uninterrupted this ensures that node having problems with receiving cannot block the bus. This is something which is very very important a node may find it is getting in to repeated error. This is because why if you remember when a node goes in to passive error mode when there are more frequent errors and when it is a normal mode of a operation. So; that means, the error that is detecting may be because of the maul functioning of the controller CAN controller at that node itself. So, effectively what it is pushing now it is pushing 6 passive bits and these bits will not modify the data on the bus, because low bit is dominant bit is it is clear?

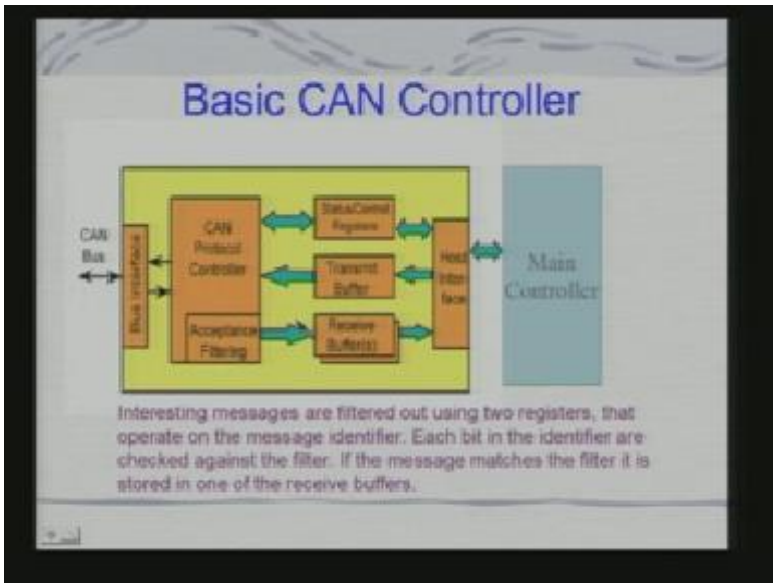
So, data transmission will take place without any interruption only the current node will be unable to receive the data. And since you are sending any kind of requires a data with identifier which is not link to the, but link to the meaning of the data. There may be other node which may satisfy you request for the data that is how the complete operation can on even, because of a failure of a single node is it is clear? So, this advance error handling makes can a very attractive protocol for these kind of embedded network embedded application where you are using it say in a car in industrial situation as well as even in medical equipments.

(Refer Slide Time: 51:49)



So, the error frame look something like this. So, here are the error flag and this is the error flag which is transmitted once the error is detected by the corresponding transmitter are the basic.

(Refer Slide Time: 52:06)

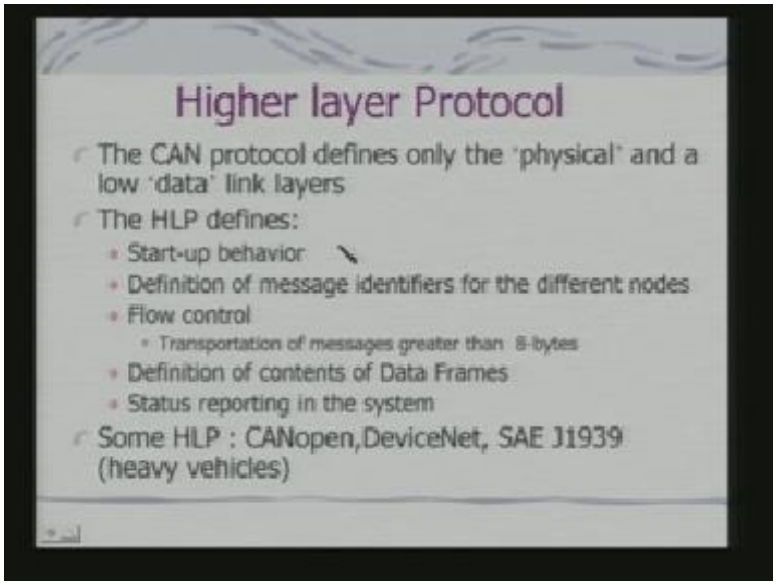


So, basic CAN controller look something like this. So, this is your main controller what is this main controller? And main controller can actually be your micro controller or an ASIC. This is basically at the Can controller additional hardware which also implement this protocol. So, what is interesting here is that you has got the status and control

registers transmit buffer and a receive buffer and the CAN protocol controller which is which will implement what is called an acceptance filtering this protocol controller examines what the identifier field. So, what you say the interesting message is the filtered out by using 2 registers that operate on the message identifier, because that value be loaded on to this register of the protocol CAN controller identifier.

So, simple conceptual model could be from if this CAN a controller is configured with a micro controller for a particular task then this micro controller would load a particular value to that register. So, the protocol controller will do what will check for that particular value in the identifier field of the CAN message if that value message if this 2 values matches. Then what it will do then it is stored in one of the receive buffers they can be multiple such received buffer. If there is a match between the value loaded in that register with that of the identifier field of the frame being received. Then the data part of that frame is stored in the receive buffer and from receive buffer it will get transfer to the main controller for processor. Now, there can be a higher protocols built on top of this basic physical layer and the data link layer

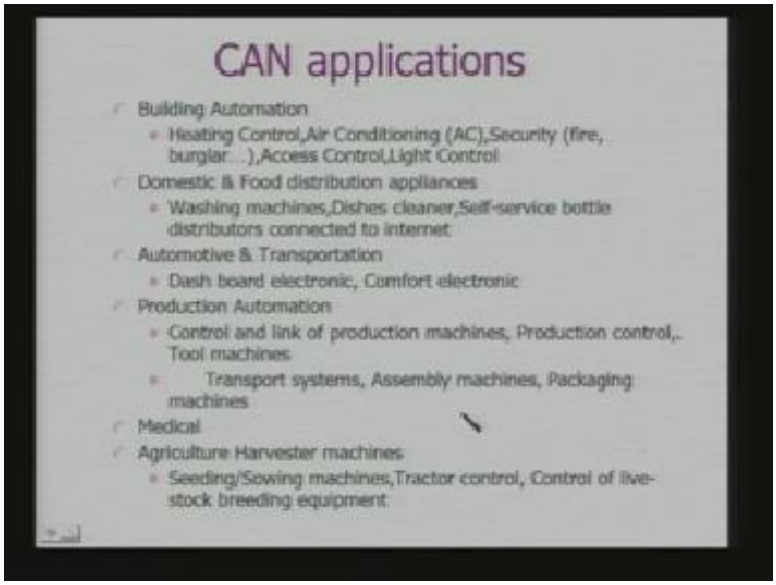
(Refer Slide Time: 54:03)



So, this higher layer protocols can define what a start up behavior definition of message identifiers for different nodes, because who defines this message, identifiers message identifier the protocol says that many bits are reserve for this. But what is the meaning of decide in defers that has to be define through a higher level protocol then definition of

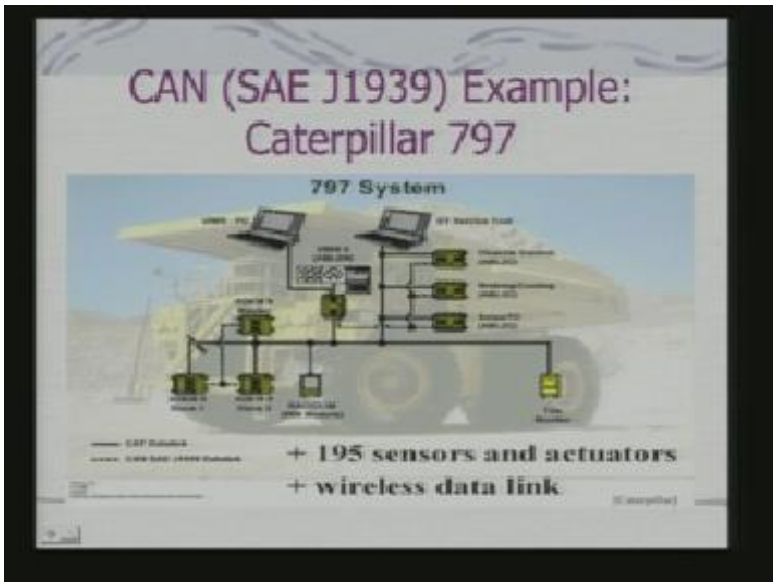
contents of data frames also the flow control. Because you need transportation of messages greater than 8 bytes there has to be a flow control mechanism implemented that comes a part of the higher layer protocols. So, this are some of the features with the higher level protocols to implement when their when an application is being implemented around CAN. And there are some protocol which are available say CAN open device net there are also protocols developed by this kind of vendor associations. This is again a specification from automobile engineer for targeted for heavy vehicular application SAE J1939 this is an automobile association specifications.

(Refer Slide Time: 55:18)



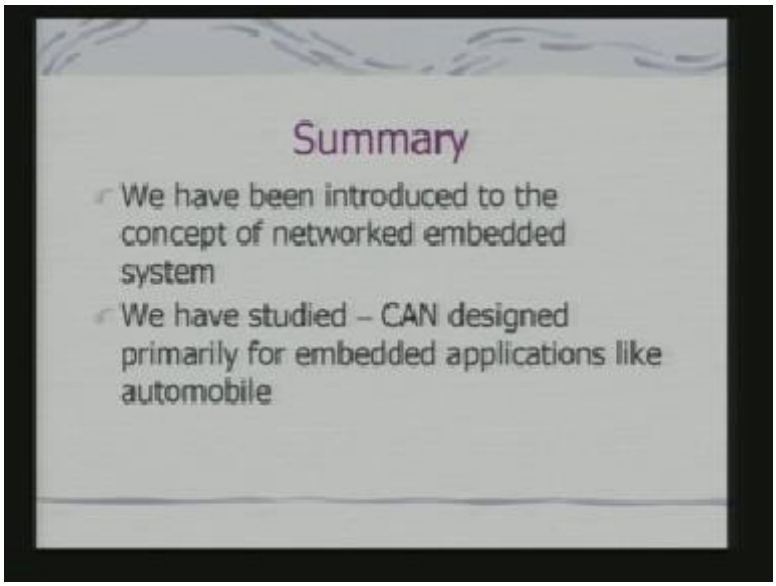
So, the applications many; in fact, building automation, a food distribution appliances, transportation, production, medical, agriculture harvester machines. So, you will find the interesting thing is that all this applications are with regard to the devices which are to be used in a variety of scenario situations which is not necessarily a very kind of a laboratory controlled conditions.

(Refer Slide Time: 55:56)



So, that is why this error control feature of CAN, become something of importance an example is this caterpillar this has got 195 sensors. In fact, an actuators plus wireless data link in such a way it is look at the number of processing elements. And sensors being going in to a heavy vehicle and the need for therefore, networking protocols are sophisticated physical data link. Networking protocol to manage all the sensors and processing elements in such a kind of a vehicle given. If you see that your automated drilling vehicles which are used for highly explosions another's there also the various kinds of components are used with sensors and processing elements. They also needs to be network and scanners a protocol which is used for those purposes as well. So, this brings us to end of today's lecture.

(Refer Slide Time: 56:52)



So, what we have understood; obviously, the concept of networked embedded system through this basic application and you have studied this protocol which is design for applications like automobile. So, it is not a general purpose protocol, but it not that general purpose protocol there also not being used for embedded system. So, we shall look at use of general purpose protocols in embedded appliances in the next class.