Digital Voice and Picture Communication

Prof. S. Sengupta

Department of Electronics and Communication Engineering

Indian Institute of Technology, Kharagpur

Lecture - 36

H dot 323: Multiplexing Schemes

After having seen the different aspects of H dot 323 in this lecture we are going to study about the multiplexing schemes in H dot 323.

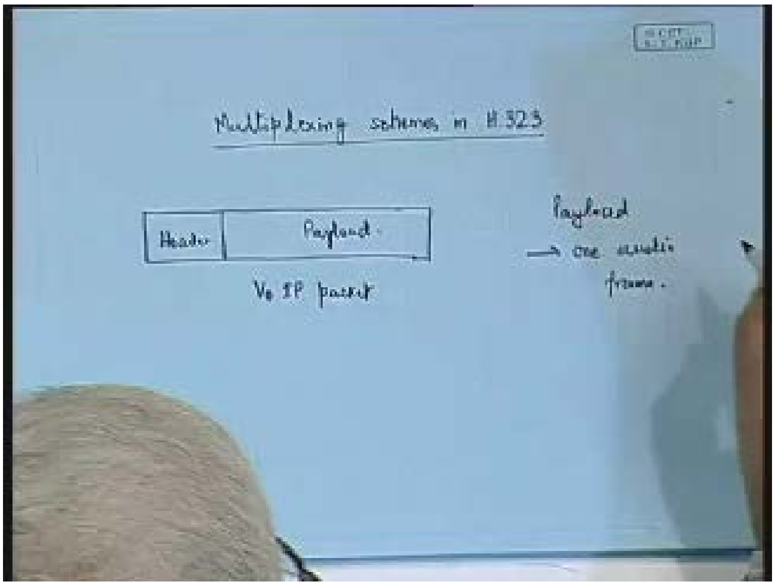
Now one trade-off aspect which we have been discussing I think we discussed over the past few classes sometimes we are referring to this, see there is a trade-off that one has to do when the H dot 323 is being used and that is between the efficiency of compression or rather to say it is a trade-off between the bit rate and the efficiency that we are having in terms of its payload transmission.

You see one of the aspects that we do in the case of H dot 323 is that we normally try to use a codec which is having a very low bandwidth codec so that if you are taking something like a 8 kilobits per second codec in that case in order to accumulate an audio frame you may have to take substantial amount of time. But when an audio frame is accumulated then what you can do is to transmit that audio frame over the IP or rather to say you have to packetize that audio frame and what you do is that by introducing some headers you have a scheme so that the header is followed by the payload.

And normally for VoIP applications what we encounter is that the payload size becomes often quite small because if we try to make the payload on the higher side in that case the difficulty that we face is the larger accumulation delay in order to form that particular audio frame. So it is something like this that supposing this is the header information which is more or less a fixed information for a VoIP packet; say this is the composition of a VoIP packet we are talking of (Refer Slide Time: 3:46) and this is the header and this is the payload, now normally as a payload we consider one audio frame and obviously a small bit rate would mean a large accumulation time.

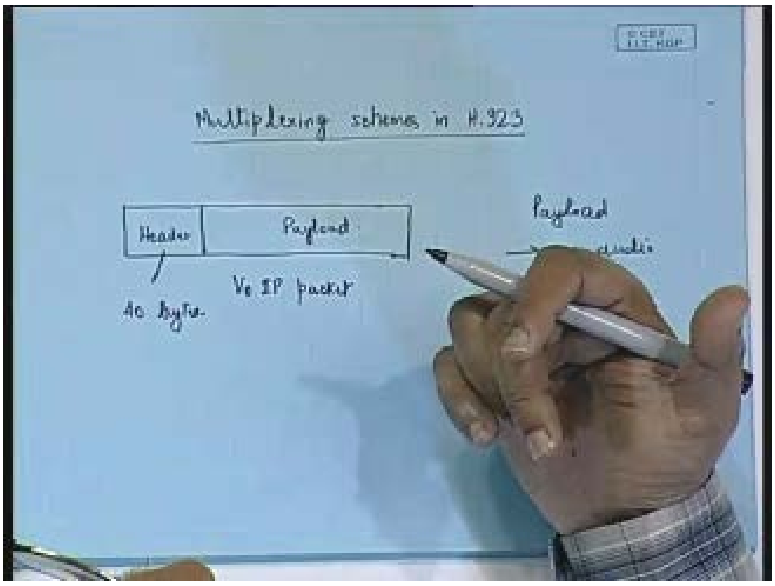
So what is the solution?

(Refer Slide Time: 3:57)



Either we have to go in for a larger bit rate or otherwise we have to accept this accumulation time which will which will lead to delay. Because this, as you know this buffering delay is also forming a very considerable part. Because as we discussed that in the standards of H dot 323 it is specified that the maximum end-to-end delay barring the very long distant calls like what happens over the continents; I mean, all the overseas communication there we may have to tolerate end-to-end delays which is higher than 150 milliseconds. But normally in 150 milliseconds of total end-to-end delay time if we take in that case this payload formation itself will take a very significant part of it. And not only that............ now in order to reduce this problem what we said is that we reduce the payload size. But of course the fragmentation we do not do in less than one audio frame; say one audio frame is the unit which we consider and even then it is seen that................ whereas one audio frame in a very compressed form could mean that it is 10 to 30 bytes; the payload the header of the VoIP packet typically becomes 40 bytes. So you can see that what is the payload efficiency; the payload efficiency is very small because if it is forming only let us say 10 bytes; take the worst case, if it forms 10 bytes out of total 10 plus 40 that is 50 bytes of information, only 10 bytes is the payload then in that case it is an efficiency of only 0.2 so naturally we have to think about increasing this efficiency and how we can do that.

(Refer Slide Time: 5:55)

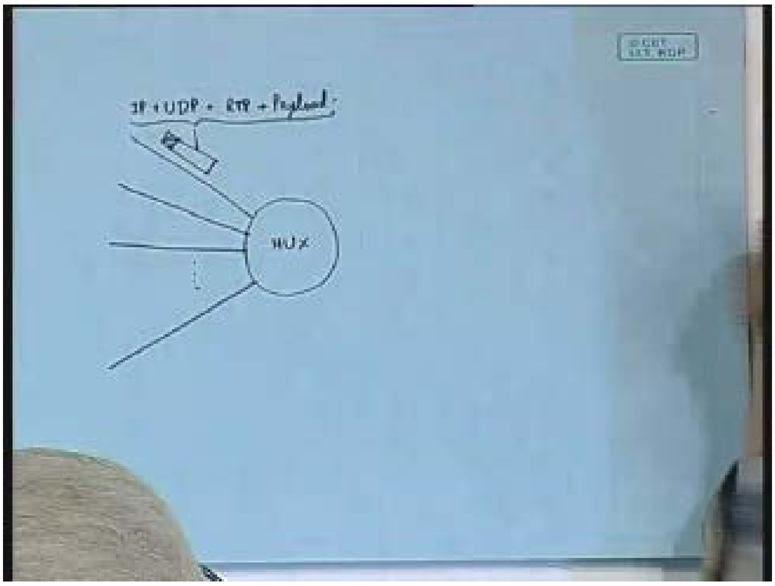


One of the schemes whereby we can do it is to introduce a kind of multiplexing scheme between various VoIP sources who would like to send their packets to the destination. Normally in a large scale VoIP application it is not just one user who would like to send the VoIP packets but there would be several users. So, if we can introduce a multiplexing scheme whereby we can transmit more than several audio frames, I mean, if we transmit several audio frames which are coming from different sources and multiplex them together perhaps in that case it should be possible to send large information and we have to see that in such a kind of scheme is it possible to really have some amount of compression in the header information. This is what is being done as a multiplexing scheme.

So typically what we are going to have is something like this that at the source end we are going to have a multiplexer and supposing the multiplexer is connected to different sources; so let us say that we draw some sources, now each source is lying on its own local area network. Supposing we have a local area network through which we are sending one audio packet, so when we are sending the audio packet then that audio packet will contain a header information and then there will be a payload.

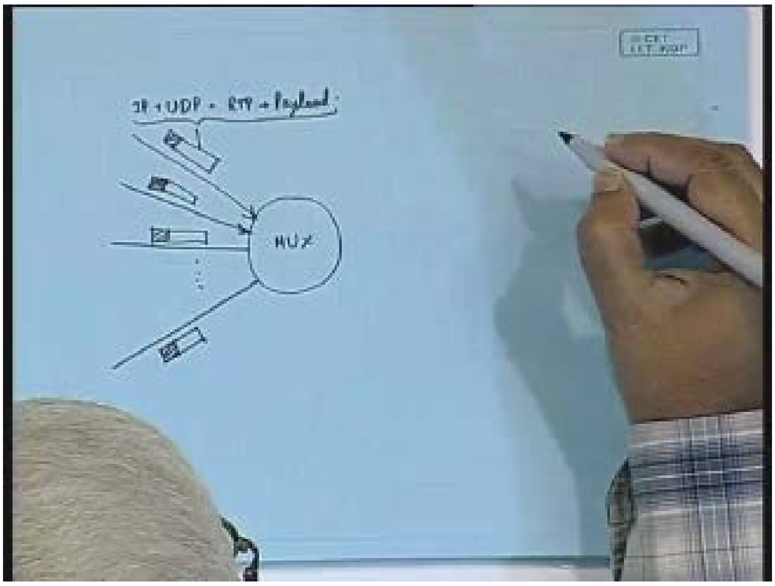
When we are sending the audio packet essentially what we are doing is that, I mean this is itself a LAN that is why we have to send it not only over the RTP, so typically the essential ingredients that would compose this bit-stream is that there will be UDP header or rather to say there will be................to start with there will be IP header, there will be UDP header, there will be RTP header and then there will be payload information. So this whole thing IP/UDP/RTP headers plus payload this would be the composition of this entire packet. Now out of this say header is this much and the payload is this.

(Refer Slide Time: 9:08)



In fact, here we can we may have a bad efficiency in the sense that as I was telling you that since the unit here since the payload is only one audio frame that is why we can say that one audio frame will be a fraction of what the total bit stream is so here the efficiency is low. But nevertheless, in this case we may not mind having a poorer efficiency because the bandwidth what we are having here, I mean, for the local area network normally there is no bandwidth constraint because local area network will be having limited number of users so there the bandwidth is not exactly a restriction that is why the deteriorated payload efficiency will not matter to us much and individually all these individual LAN segments would be having like this means there will be header information, IP component, UDP component, RTP component and then the payload which will go into the RTP like that for every channel.

(Refer Slide Time: 10:19)

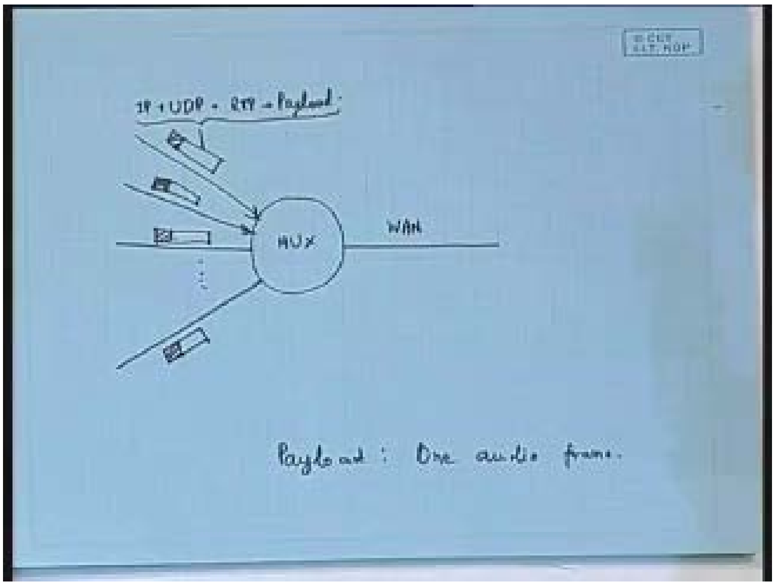


Yes please,

Can we have a larger payload size in this case or do we have a larger payload size in this case?

No, individually when we are talking of the LAN segments there the payload efficiency is poor. That means to say that the payload could be still one frame, the payload could be still one audio frame which means to say that you may have efficiency in the range of 0.2 to 0.5 so here we are not increasing the efficiency rather we do not want to increase the efficiency because here plenty of bandwidth is available; the availability of bandwidth is abundant in the individual LAN segment so there we are not concerned. But what we are concerned with is that when we go out of this multiplexer and then we are in the WAN segment wide area network, there in the wide area network we are going to face some bandwidth constraint. So now that we have decided to use a multiplexer over here what we can logically think is that that why do not we take all these individual packets and then embed them together in the form of one bit-stream but then again that combined packet or that mega packet if we call that mega packet has to be transmitted over the WAN and again it is to be transmitted over the IP and UDP.

(Refer Slide Time: 11:38)



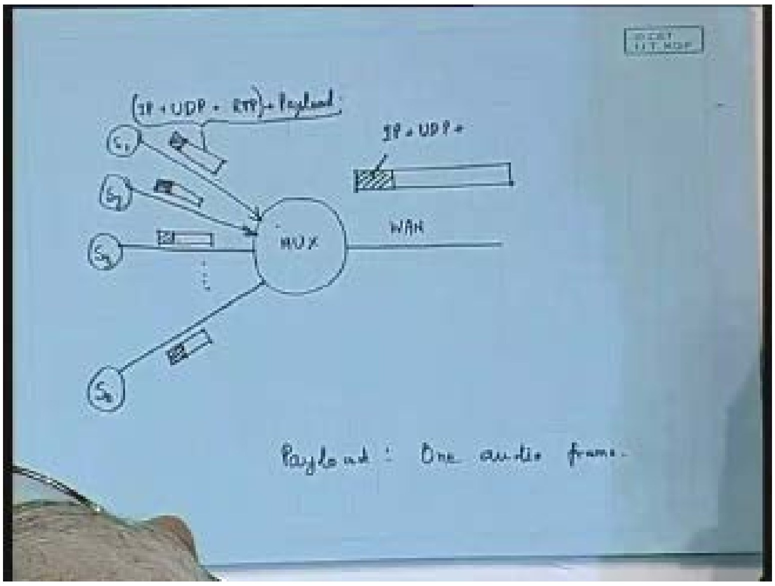
Payload : One audio frame.

That means to say that what we are thinking of now is that there will be IP header, UDP header and then these individual packets but these individual packets coming from the different sources will also be having their own header. So as such if we are talking of the efficiency we are not improving the efficiency yet till the time we achieve something like a compression in the header; now can we achieve that that is what we are going to examine.

So here what we are typically having is that supposing this is the total composition of the WAN packet and here we will be having the header as the IP plus UDP plus we are having the total information here that means to say that the header was already there and then the payload information. Now out of that the IP and the UDP header we may dispense with because what happens is that this IP and UDP header will be pertaining to the source and you can that time use this as the destination so from the individual source supposing this is source S 1 this is S 2 S 3 and up to S n we consider n number of sources.

Now whenever we are considering S 1 the destination is this multiplexer. So the source information, destination information and the stream ID information whatever information goes into the IP header everything has been taken care by this multiplexer. So the multiplexer has rather identified that stream and now the multiplexer is putting into the combined bit stream. But even in the combined bit stream also all though the IP and UDP headers for individual streams will no longer be needed. But still whenever it is multiplexing the different streams in that case corresponding to the individual streams there must be some header because when it is multiplexing it must identify that whether it is coming from source S 1 or whether it is coming from source S 2 so at least some amount of information should be there and not only that the time stamp which will correspond to this. Because if it is an audio packet that is coming from the stream S 1 in that case that is associated with some time stamp information. It is also containing the destination information so the source information, the final end destination information and the time stamps so these things will still be needed. So there will be some header which will be anyway required.

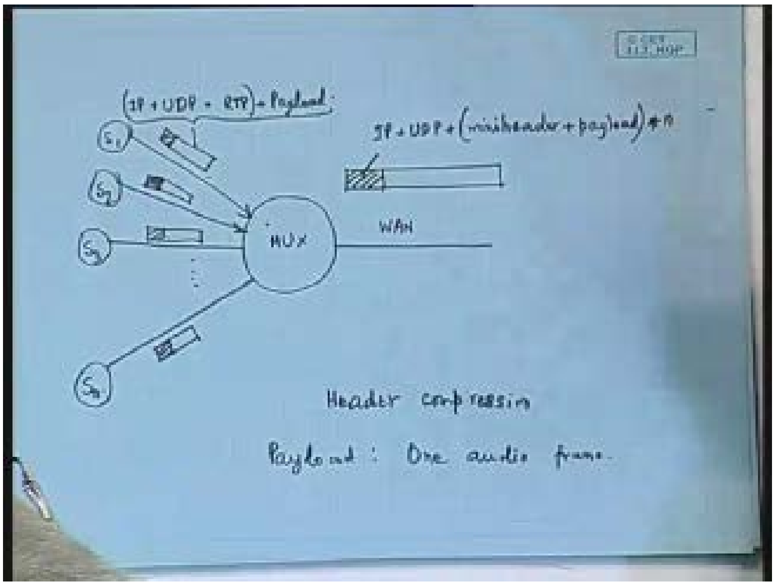
(Refer Slide Time: 15:32)



So what we can see is that, that header information may be less as compared to the total header information what we have over here. So here the IP plus UDP plus RTP these header components if it takes 40 bytes may be that now after multiplexing the individual streams what we are sending in the form of multiplexed packet that may not contain 40 bytes there we may be able to reduce the header size very significantly.

In fact, by research it was shown that it may be reduced the header size could be reduced to as low as 2 bytes even. So from 40 bytes........... so there is something called as the header compression. So we are going to study that what kind of header compression scheme can we adopt but definitely if the header compressions are done in that case what we are considering is that for the individual streams we will consider we will have a miniheader and that miniheader may be miniheader may be as low as 2 bytes; miniheader plus the payload the individual stream payload, and since we are having n number of such sources it will be miniheader plus payload multiplied by n. So this will be (Refer Slide Time: 16:44) the thing that follows the IP and UDP. So the data sequence will be first the IP header for WAN, UDP header for WAN, then miniheader along with the payload for the first stream, miniheader plus payload for the second stream and so on up to n number of streams it will be there.

(Refer Slide Time: 17:03)

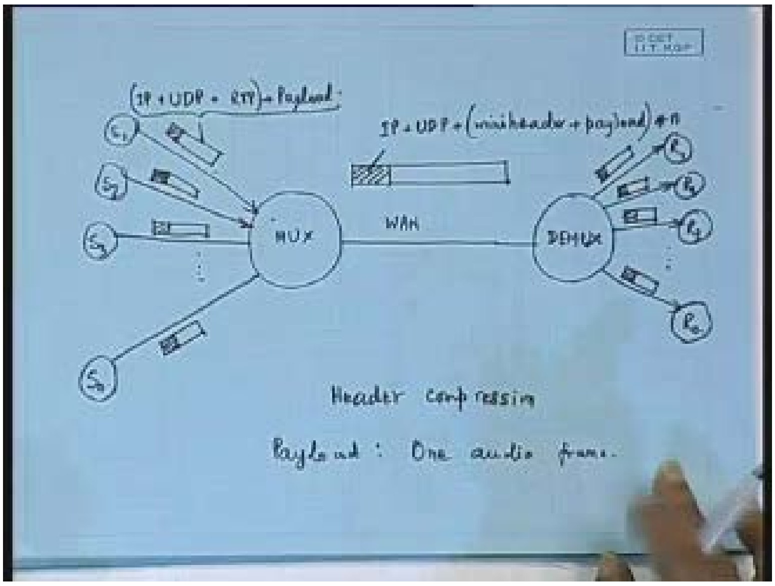

(IP + UDP + RTP) + Payload
IP + UDP + (miniheader + payload) * n
S1
S2
S3
Sn
MUX
WAN
Header compression
Payload : One audio frame.

Now, at the destination end just the reversal will be done that means to say that here we have to have a demultiplexer (Refer Slide Time: 17:14). Now what the demultiplexer will do is that the demultiplexer will identify; see, from the header information it will identify that this is the demultiplexer that we are looking for and then after this the IP and the and after this the individual streams will be obtained and there from the miniheader information we should be able to again compose that what the desired information will be because miniheader may contain some information in the compressed form so there we have to again decompress it and then send it to the different receivers so we will be having likewise n number of receivers.

So let us say that these are R 1 R 2 R 3 up to R n so there are n number of receivers and to each of these receivers the data packets or rather the voice packets will go with its own IP plus UDP plus RTP. Then again to these individual LAN segments we have to form this

IP/UDP/RTP header plus the payload information and this will go to the different receivers at the destination end.

(Refer Slide Time: 18:52)

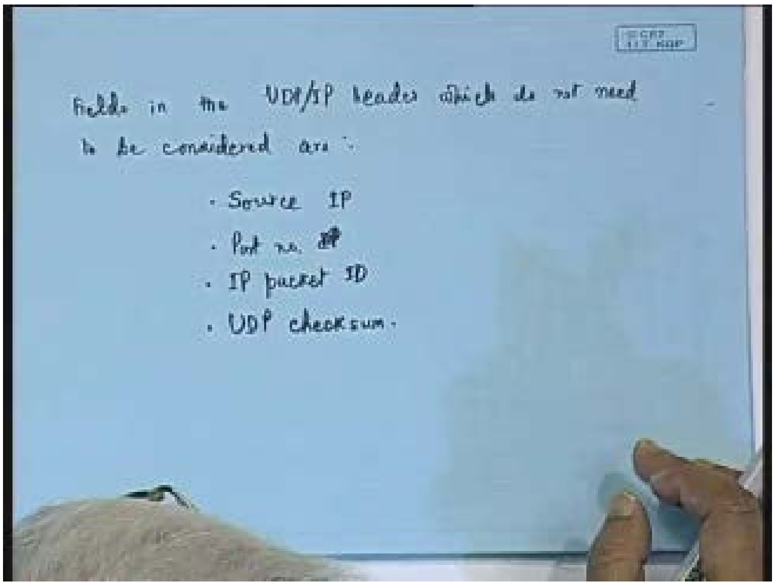

(IP + UDP + RTP) + Payload
IP + UDP + (miniheader + payload) * n
S1
S2
S3
Sn
MUX
WAN
DEMUX
R1
R2
R3
Rn
Header compression
Payload : One audio frame.

So, where we want to have a big saving is that if a significant of a significant amount of header compression could be done while transmitting the information through the wide area network then the overall payload efficiency can be increased. Now what philosophy is actually adopted in order to achieve such kind of a bandwidth compression such such kind of a header compression.

Now what are the header information that will be needed?

When the packets are transmitted from the source to the MUX the MUX acting as the destination as the first destination for the source packets it is going from just from the source to the multiplexer, the fields in the UDP that do not need to be considered are...... fields in the UDP/IP header which do not need to be considered these are the source IP, then the port number IP, then the port number sorry port number then the IP packet ID and then the UDP checksum.

(Refer Slide Time: 21:02)



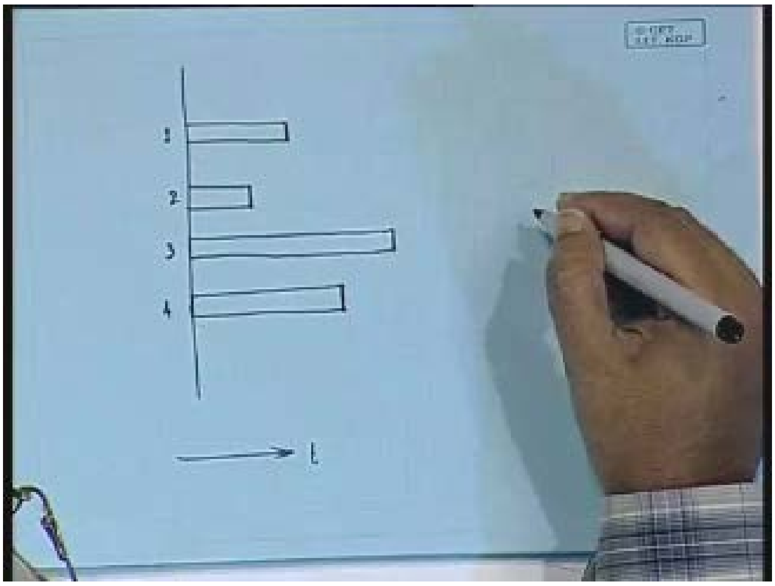
Fields in the UDP/IP header which do not need to be considered are :-

- Source IP
- Port no.
- IP packet ID
- UDP checksum.

But whenever we are recomposing; I mean, at the multiplexer end whenever we are obtaining each of these in that case the question is that what are the other information that we will be able to compress; that part we see. Before that we go in to one important aspect and that is to say that at what rate are we going to deliver the packets.

Let us say that supposing that there are n number of sources and all the sources they start collecting their information from this time so the time axis in this case we take like this (Refer Slide Time: 22:00) and supposing this is where the stream number 1 completes the first audio packet, supposing the stream number 2 completes the first audio packet here, supposing the stream number 3 that takes a longer time to compose an audio packet, supposing stream number 4 takes this much; supposing we just consider four sources 1 2 3 and 4 and this is the time when its individual audio frames are ready.

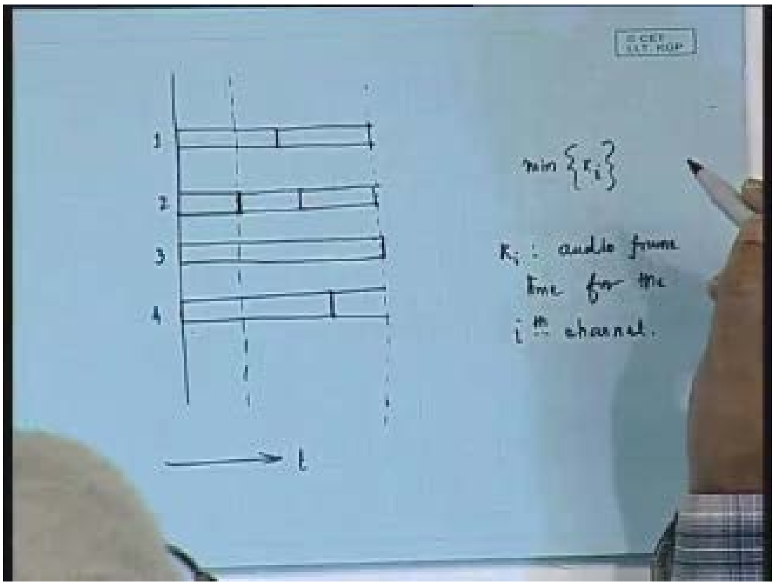
(Refer Slide Time: 22:32)



Now the time at which the audio frames are getting ready they are different and they could be different; why? Because its source could use a different compression scheme; one of them may be using G dot 728, another may be using G dot 729 their bit rates could be different. And not only that, the amount of compression that can vary from one frame to the next and that is why there is no guarantee that all the.............. in fact in general it will never happen that all the channels will complete their collection of one audio frame at the same time simultaneously.

Now the question is that if such is the case in that case what should be the rate at which we compose the packets; do we wait till the maximum time elapses that means to say that if we think that at this time since at this time instant (Refer Slide Time: 23:40) already we will be having the streams from frame number streams from 1 2 3 and 4 one audio frame each has arrived so now it is the time to send the multiplexed packet; is it a good policy? It is definitely not; why? Because if we consider this way then by this time we may be receiving some more packets from this (Refer Slide Time: 24:11) by then may be three packets of two will be over I mean three......... not packets sorry three frames of source two may be available, two audio frames from source one may be available may be one and a half times the source four are available one and a half frame time is available from source 4 that means to say that we should try to make optimal use so the policy should not be to see that which is the maximum time for one audio frame rather to say we consider that amounts to the different channels which we are multiplexing which is the minimum audio frame time. Supposing K i is the audio frame time for the i'th channel then we should see that what is the minimum of this K i.

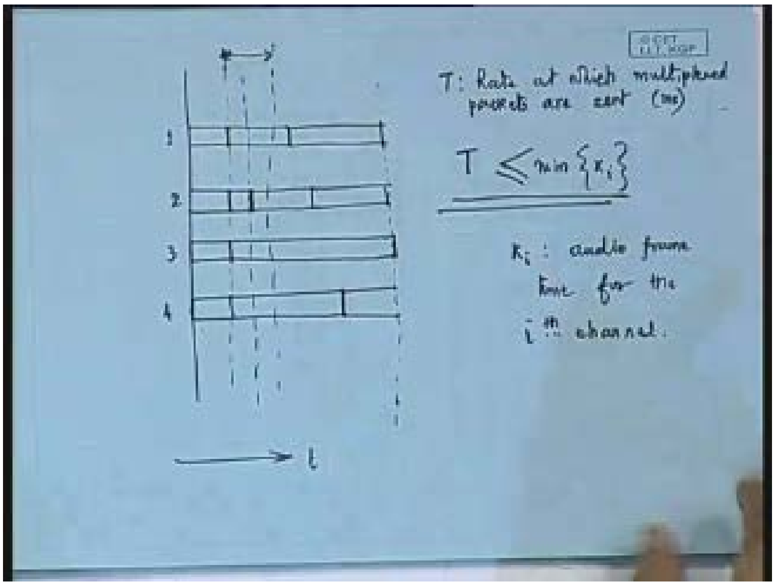
(Refer Slide Time: 25:36)



Now in this case (Refer Slide Time: 25:29) in this particular example when we consider this time instant then audio frame 2 is ready but audio frame 1 is partly ready, audio frame 3 is partly ready, 4 is also partly ready but what we can do is that still whatever we have received from the source 1, source 3 and source 4 these information we can transmit into the packet even though they are not corresponding to one complete audio frame, it is only a part of the audio frame but who says that we have to form one complete audio frame it is not needed.

So if we take the frame rather to say that if the multiplexer sends out a multiplex packet every T milliseconds so T is the rate at which T is the rate at which the multiplex packets are sent. So this T we are measuring in milliseconds that means to say that how many milliseconds do we take to send the multiplex packets. So in that case what we should achieve is that T should be less than or equal to minimum of K i. If we take the minimum of K i this is what we have taken as minimum of K i (Refer Slide Time: 27:06) alternatively we could have accumulated the packet here itself because even here also.............. if we take T to be only this much in that case we are not waiting one audio frame to be completed for every channel. But nevertheless we collect this much from source 1, this much from source 2, this much from source 3, this much from source 4 so individual headers we append and we send the information and together then we are multiplexing and then we will be having the accumulated effects of all these bit-streams, accumulated effects of all these bits which will go into the multiplexed bit-stream. Therefore, that is how the rate at which the multiplexed packets are sent is decided that is based on the minimum of K i.

(Refer Slide Time: 28:05)



T: Rate at which multiplexed packets are sent (ms)

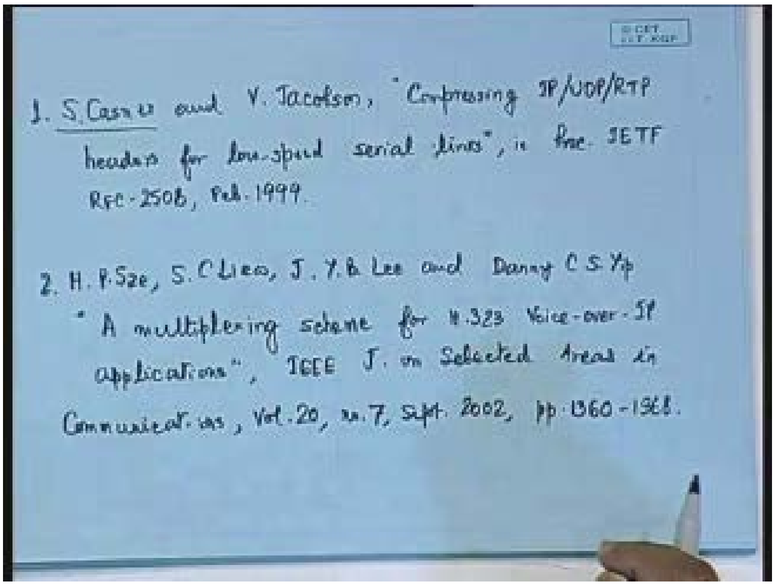
$T \leq \min\{K_i\}$

K_i : audio frame time for the i^{th} channel.

Now we again address the question of this header compression that how the header compression can be actually achieved and there we make use of an algorithm which was suggested in 1998. In fact, the reference to that is......... I am giving you the reference that is by Casner and Jacobson so this is Casner and V. Jacobson; and the title of the paper is "compressing UPC..... "Compressing IP/UDP/RTP Headers for Low Speed Serial Links".

Actually it is available as an IETF document so this appeared in the proceedings of IETF RFC 2508 February 1999. This is the compression scheme and in fact the paper that you can refer to, to learn the details about this details about the multiplexing scheme because the paper which I am referring to they also suggested and improved upon Casner's algorithm and this paper is by H. P. Sze S. C. Liew J. Y. B. Lee and Danny C. S. Yip; the title of the paper is "A Multiplexing Scheme for H dot 323 Voice-over IP Applications" and this paper appeared in IEEE journal on Selected Areas in Communications: Volume 20 number 7 September 2002 and the page numbers are 1360 to 1368.

(Refer Slide Time: 31:51)



1. S. Casner and V. Jacobson, "Compressing IP/UDP/RTP headers for low-speed serial links", in Proc. IETF RFC-2508, Feb. 1999.

2. H. P. Sze, S. C Liew, J. Y. B Lee and Danny C S Yip "A multiplexing scheme for H.323 Voice-over-IP applications", IEEE J. on Selected Areas in Communications, Vol. 20, no. 7, Sept. 2002, pp. 1360-1368.

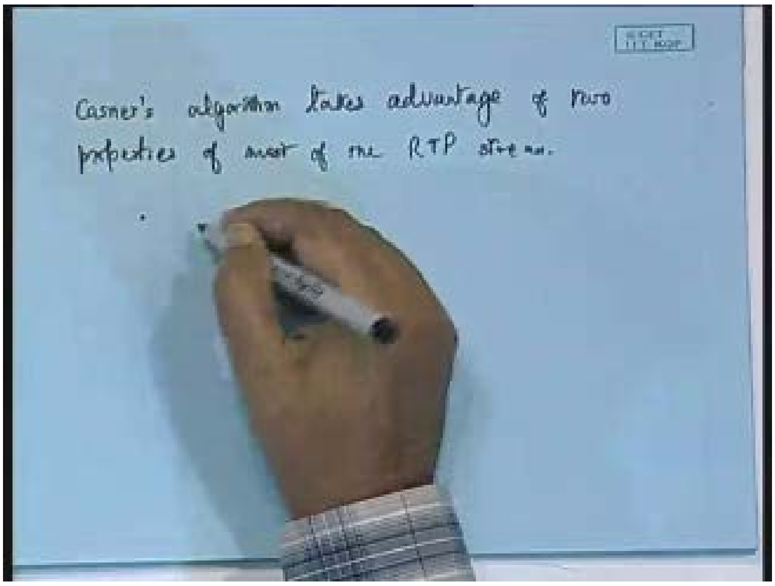
So you can have a reference to both these to understand details about the scheme. Yes, I was trying to tell you about the basic philosophy behind the header compression.

Now Casner's algorithm basically takes advantage of two proprieties in most types of the RTP streams. And what are these?

These are that most of the fields in the IP, UDP and RTP headers they do not change over the lifetime of a session; means for a particular session they remain the same. So naturally you are see whenever you are thinking of doing any header compression you are looking for the redundant information that is present in header.

Now if you find that............. supposing there are ten sources and there are ten destinations, now source number 1 is sending some packets to source number 5 so source number 1 and destination number 5 supposing they are involved in a session, now when source 1 is sending to destination 5 and that is declared as a session then whatever packet source 1 is sending for that particular session those packets will be having a set of headers, I mean, those packets will be having some header which will be there for every packet that source 1 sends to the destination. But mind you we cannot dispense with that packet altogether because when source 1 to destination 5 packet is identified after that immediately source 2 may be sending the next packet to destination number 4, source 3 may be sending to destination number 10 so like that different things will be there different packets will be there or different streams will be formed that is why we have to have some mechanism of identifying also. But that identification is it essential for us to have the complete header information, no. So what is being done is that they have suggested that using a context table if it is possible to refer to that table and get only some identifier information or to form only some identification information which can go into a miniheader. So rather than giving the complete header information that means to say the constant fields which are associated with the header that may be dispensed with and instead of sending the constant headers we can try to have some miniheaders which will have only the changed information and through which one can identify that who is the source and who is the destination. So there should be a reference, there should be a table and the minimum information that will be required to refer to that table only that will go into the miniheader. I will show you this with some example so then it will be clear.

(Refer Slide Time: 35:44)



Casner's algorithm takes advantage of two properties of most of the RTP stream.

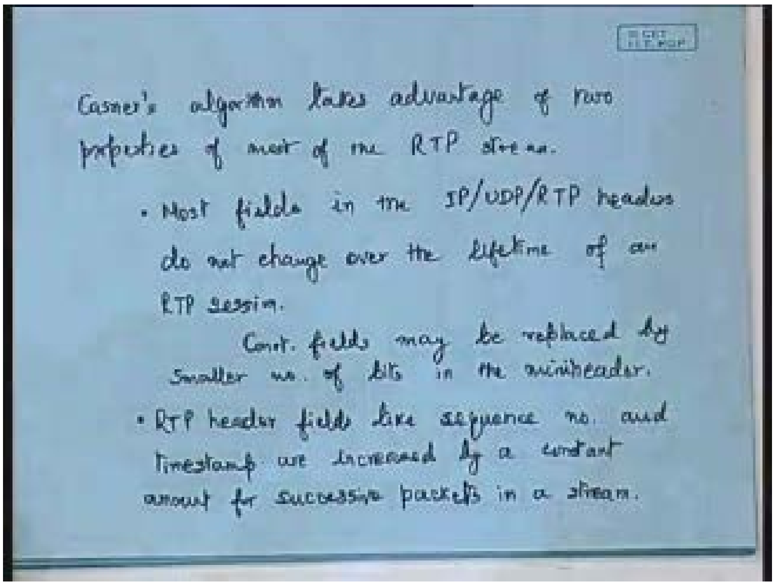
So the first property for most of the RTP stream is that most of the fields in the IP, UDP and RTP headers do not change over the lifetime of an RTP session. So naturally these constant fields may be replaced with a fewer number of bits so the constant fields may be replaced by smaller number of bits in the miniheader.

Now everything is not constant because even though source 1 is sending the packets to destination number 5, the packet is identified with some packet ID. That means to say that some kind of a sequence number will be there because with every packet switching as you know that there has to be some packet ID because there is no guarantee that the packets will be delivered exactly one after the other.

In fact, individually the packets may be differently routed also and it is only at the destination where after identifying the packet ID the destination will put the packets in the sequence and then form....... I mean, the ultimate information can be obtained and that happens with every packet switching scheme.

Thus, some packet ID will be there but when it is being generated from the source the packet IDs which are going or the sequence numbers which are going that should be just incremented by 1. If the previous sequence number was 5 then the next packet will have a sequence number of 6, the next packet will have sequence number of 7 like that. That is why we need not have to send the absolute information or rather only the incremental information that there is an incrementing by 1 or if incrementing by 1 is implied then we can automatically infer that in the next packet that is coming from this source to this particular destination there one can have one number then we can indicate this varying thing by having a kind of an a priori knowledge that the sequence number is going to increase by 1 and likewise for the time stamp also that the time stamp which is associated at this moment, for the next packet with the time stamp because the rate at which the packets are composed that rate is not varying that rate is maintained as the same and the packets are also not arriving in (bur...39:30) in most of the cases we will be finding that the time stamps are also regularly updating that means to say that there is a....... the sequence number and time stamps they are actually increased by some constant amount that is the second property that RTP header fields like *s*equence number and time stamp are increased by a constant amount for successive packets for successive packets in a stream.

(Refer Slide Time: 40:53)



Casner's algorithm takes advantage of two properties of most of the RTP stream.

- Most fields in the IP/UDP/RTP headers do not change over the lifetime of an RTP session.

  Const. fields may be replaced by smaller no. of bits in the miniheader.

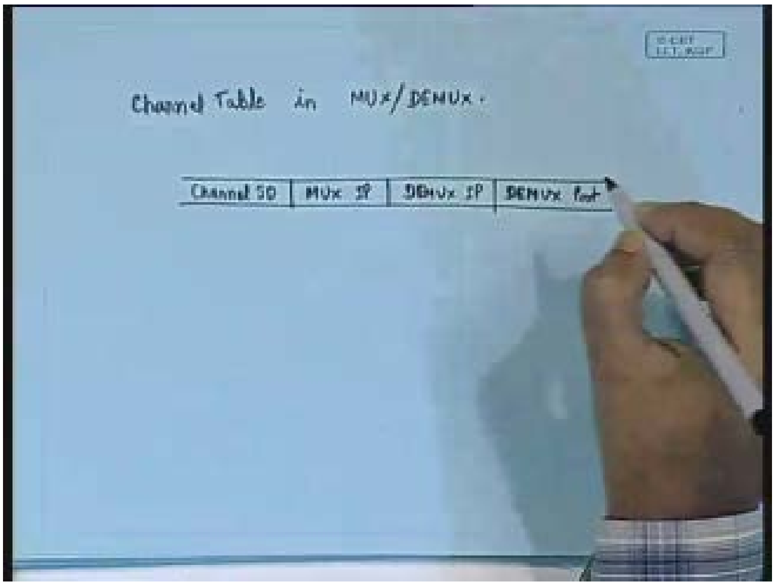
- RTP header fields like sequence no. and timestamp are increased by a constant amount for successive packets in a stream.

Now when I say packets in a stream, to say the same source ID and the same destination ID. Means if it is source 1 sending to destination 5 after that, in the multiplex we may be having source 2 sending to 6, 3 sending to 4, 4 sending to 10 like that it can happen. But again whenever I am having the next packet that source 1 sends to destination number 5 then we will be having its sequence number and time stamp which are augmented by a constant factor. And if the updating of the sequence number and time stamp are there by a constant amount in that case some kind of a differential coding can be employed to compress these fields into fewer number of bits.

Therefore, (Refer Slide Time: 41:45) these two properties of the header definitely leads to some amount of compression. Whenever we do the compression as I was telling you that we can do the compression but at the same time we should not be losing the identification scheme. Still we should be able identify anytime that this part of the stream is corresponding to this source and this destination and this would be the sequence number for that. Now to do that we maintain a table and we will refer to the context in the tables in order to achieve that compression.

First of all that we have a channel table, we have a channel table in the multiplexer and the demultiplexer and what is the format of that channel table; we will be having a channel identifier so there will be a channel ID, there will be the IP address for the MUX, there will be an IP address for the demultiplexing and there will be the port address for the demultiplexing, so DEMUX port.

(Refer Slide Time: 43:16)



Channel Table in MUX/DEMUX.

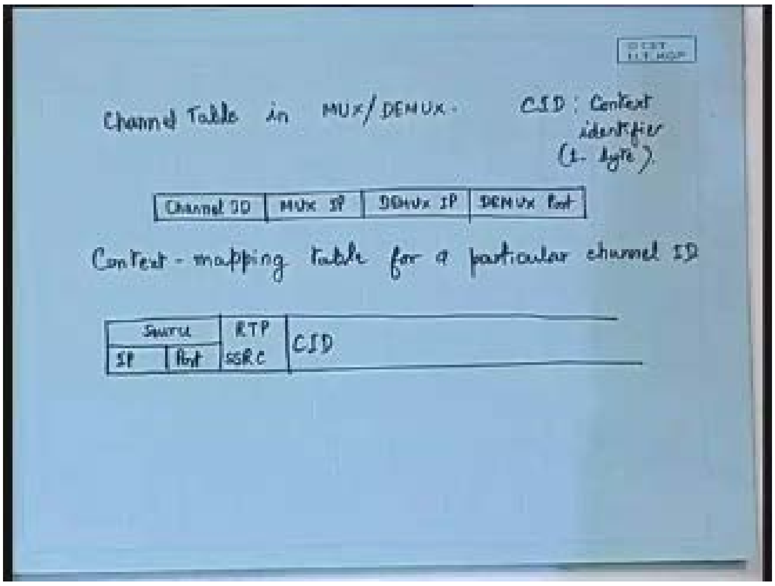
Channel ID	MUX IP	DEMUX IP	DEMUX Port

You see that one multiplexer may cater to different demultiplexers there is no problem because the scheme which I was telling you who says that there has to be only one DEMUX, we may have another DEMUX, we may have another DEMUX like this (Refer Slide Time: 43:38) and not only that, the port address of the DEMUX that also could be different so the multiplexers IP, demultiplexers IP, DEMUX port everything together will define a channel because the channel is from the source to the ultimate destination which is connected to one of the demultiplexer ports.

So, if you explicitly specify that what is the MUX IP, what is the DEMUX IP and what is the DEMUX port and you also give the channel identifier then that defines a complete channel which will be there in the channel table. So like this there may be several channels and each of these channels will be listed in the form of a table which we are calling as the channel table. Then other than the channel table there will be .... for a particular channel ID there will be what is called as a context mapping table so the context mapping table for a particular channel ID in the MUX would be like this.

So, for a particular channel ID we are going to have the format like this that first will be the source information. Now the source information will have two components: one is the IP and the other is the port address then there will be RTP's source identifier that field is referred to as RTP, SSRC then there will be a context identifier CID so CID's full form is the context identifier; rather to say this is a number this is actually a 1 byte information, this is a number that is assigned to one particular stream. So if it is source 1 sending to destination number 5 that will have a unique CID; it will be assigned a particular CID and that CID will be there.

(Refer Slide Time: 46:31)



Channel Table in MUX/DEMUX.

CID : Context identifier (1- byte)

Channel ID	MUX IP	DEMUX IP	DEMUX Port

Context - mapping table for a particular channel ID

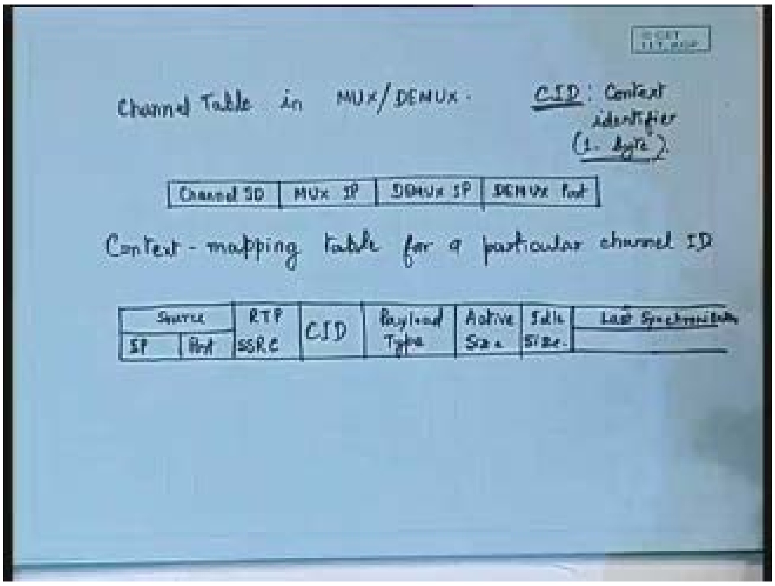
Source		RTP	CID
IP	Port	SSRC	

First of all that when you want.......... when the source 1 wants to communicate with destination number 5 in that case first of all a channel is defined for that where you have to specify that from which multiplexer it is going to which demultiplexer and what is the port number that corresponds to the destination number 5 everything should be specified in this table and then there will be a channel ID.

Now, with the help of this channel ID we should be able to get the complete information about this and one of the very important fields which we have to note over here is the field that is called as the CID, this is a 1 byte field and it is referred to as the context identifier. So this context identifier this field will remain the same for a particular session, this field will never change for a particular session. So CID will be there then there will be the payload type.

We are not going into the details of the individual fields because these details are not within the scope of our study. Just to just for the sake of completion we are just writing the different fields: active size, idle size, do not have to really bother about these fields but where we have to again note some information is that some information corresponding to the last synchronization is also necessary to be recorded in this context mapping table. By last synchronization like this that the headers which will be associated with the individual streams there could be three formats for those headers. One of the header formats is what is called as the uncompressed miniheader.

(Refer Slide Time: 00:48:15)



Channel Table in MUX/DEMUX.

CID: Context identifier (1. byte)

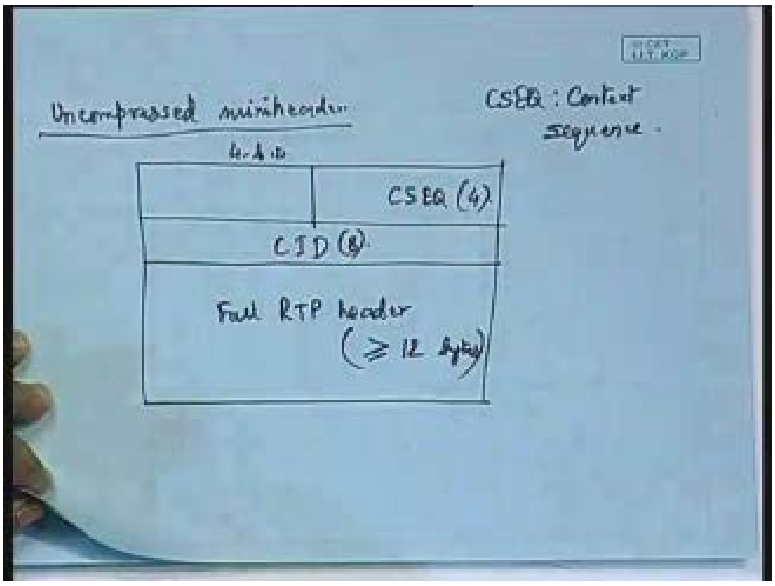
Channel ID	MUX IP	DEMUX IP	DEMUX Port

Context-mapping table for a particular channel ID

Source		RTP	CID	Payload	Active	Idle	Last Synchronization
IP	Port	SSRC		Type	Size	Size.	

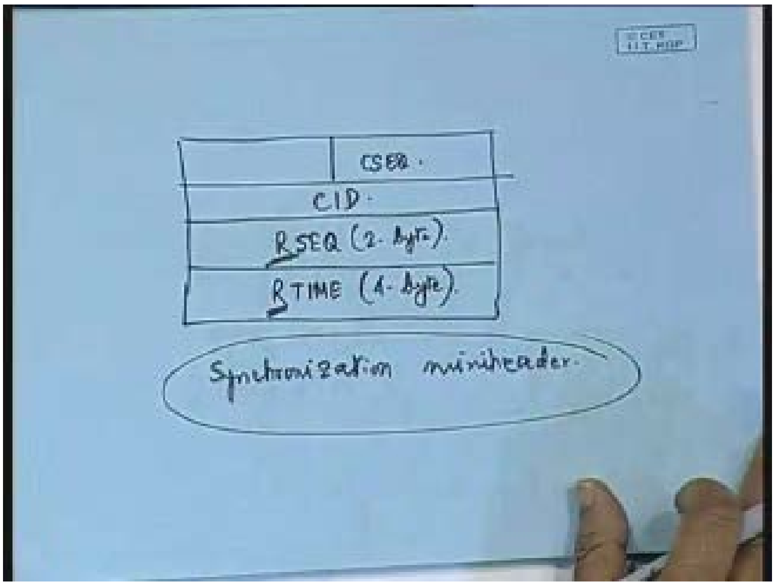
In the case of uncompressed miniheader we will be having the sequence number. This is the context sequence CSEQ corresponds to the context sequence. This in fact gets updated. With every packet this gets updated. So C sequence will get updated for every packet then there will be the CID the context identifier which I said just now. So this is 1 byte of information. In fact, there will be some identification information in this 4 bits and the C sequence will be a 4 bit information whereas CID will be 8 bit information and followed by this there will be the full RTP header full RTP header and the full RTP header will be greater than or equal to 12 bytes. This is where the complete information will go. But this is again a miniheader because we have dispensed with whatever was there as the constant fields in the IP/UDP headers we have eliminated that so this is the uncompressed form of the miniheader.

(Refer Slide Time: 49:52)



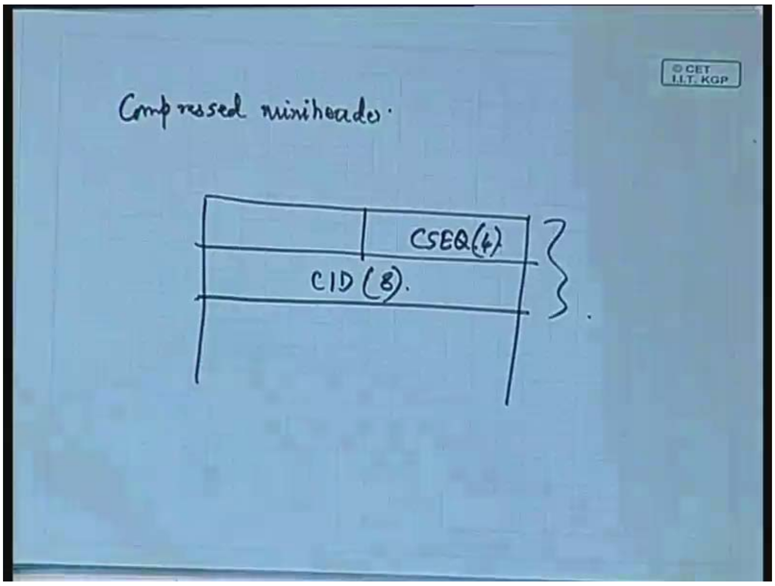
But we can also have some compressed form of miniheader which goes like this that in the compressed form we will be having....... the first byte 4 bytes will be reserved for C sequence, the next byte will be reserved for the CID and 6 bytes of information will be there one is what is called as the R sequence. Now this is the overall sequence number; overall sequence number means the sequence number which is there in the multiplexed stream and this is a 2 byte information and there is a time stamp which is............. so RTP sequence and this is the RTP time. RTP sequence is the accumulated sequence and R time is the accumulated timestamp accumulated timestamp is 4 byte information. So naturally this is some form of a compressed form but this is called as a synchronization miniheader, this is called as the synchronization miniheader and why this is called as synchronization miniheader is that even if we lose synchronization, using this miniheader we will be able to get back that what the timestamp and what the overall sequence numbers are there because we may miss some of the streams in between and in that case there should not be any error that should happen in our timestamp calculation or sequence number calculation. Because if we are relying only upon the CID in that case, CID and C sequence in that case there is a risk that this synchronization could be lost. So periodically we have to send this synchronization miniheader information.

(Refer Slide Time: 51:49)



And the third form of synchronization is the compressed form. The compressed form of miniheader which is only a 2 byte information. The compressed miniheader just contains some identification that it is a compressed miniheader but there will be a context sequence number C sequence of 4 bits followed by the context identifier of 8 bits only that is all so just two bytes of information.

(Refer Slide Time: 00:52:29)



Normally this 2 bytes of information should be enough because what you can do is that using this CID one can completely identify the stream by looking into the context table (Refer Slide Time: 52:42).

Now in this class I could not give example, rather there was no time to give some example pertaining to this. But in the next class I will be taking of some specific examples and I suppose that that will make the concepts more clear that how one can compress the header information and then we will be talking about the performance in the multiplexed domain of H dot 323. Thank you.