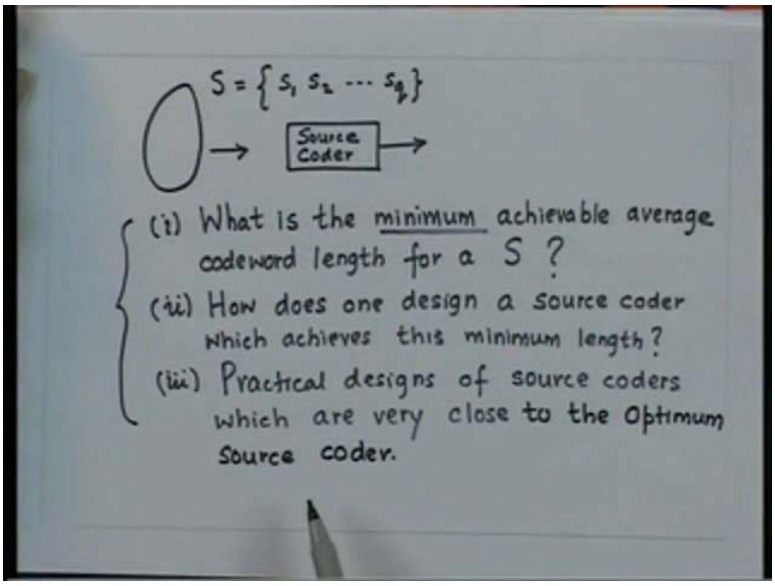
Information Theory and Coding
Prof. S.N. Merchant
Department of Electrical of Engineering
Indian Institute Technology, Bombay

Lecture - 07
Block Code and Its properties

The goal of a communication system is to translate messages from information source to its destination in a most efficient manner. An essential step in order to achieve this goal is to represent or transform a map the sequence of source symbols from the source into another sequence, consisting of symbols called code symbols from another alphabet called code alphabet. This transformation or representation of mapping from 1 sequence to another sequence is achieved by what is commonly known as source coder.

The task of a source code is to represent messages as compactly as possible, therefore the efficiency of a source coder can be judge by the compactness of the messages it achieves. And this can be measured in a way by the average, codeword length of the code with it which it generates. Now, the question that arises is how do i synthesize such codes let us take a simple example.

(Refer Slide Time: 02:45)



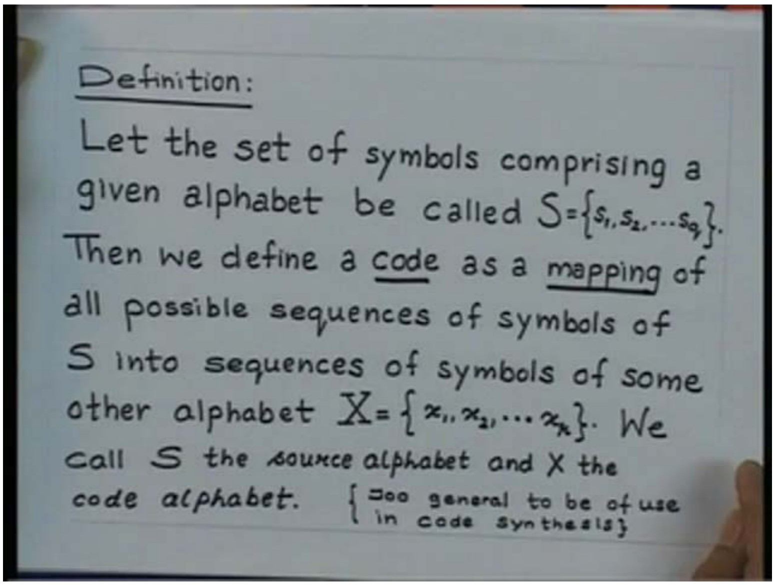
If I have a source S consisting of source symbols, then messages will be generated consisting of the source symbol from this source alphabet S. This messages which are being generated from the source are to be compactly represented by a source coder. As we have said that the efficiency of the source coder will be decided by the compactness of the messages. And this can be measured in terms of the average word length or average code word length, will look at the definition of all this term as you progress today.

Now, the question that arises is the first question that arises is what is the minimum achievable average code word length for a given source. Second question that arises is how does one design a source coder, which achieves this minimum length. Now, we will see that minimum achievable average code word length for a source S will be related to the entropy of the source S. And it is very difficult to design a source coder which achieves this minimum length, it may not be possible for us to design an optimum source coder for all the types of sources.

Only when source satisfied some particular constraint then we will see that is possible for us to design optimum source coder. So, for those sources when it is not possible to achieve the optimum source code designing, source code design then in that case we should look for the strategies to develop design source coders, which come very close to this minimum length.

So, practical designs of source coders which are very close to the optimum source coder, based on the definition of information measures and entropy calculation for the sources will try to answer this three questions. So, let us look little more into depth into the connection between the source coding and the information theory concepts, which we have studied so far. In order to do that let us define formerly what is a code?

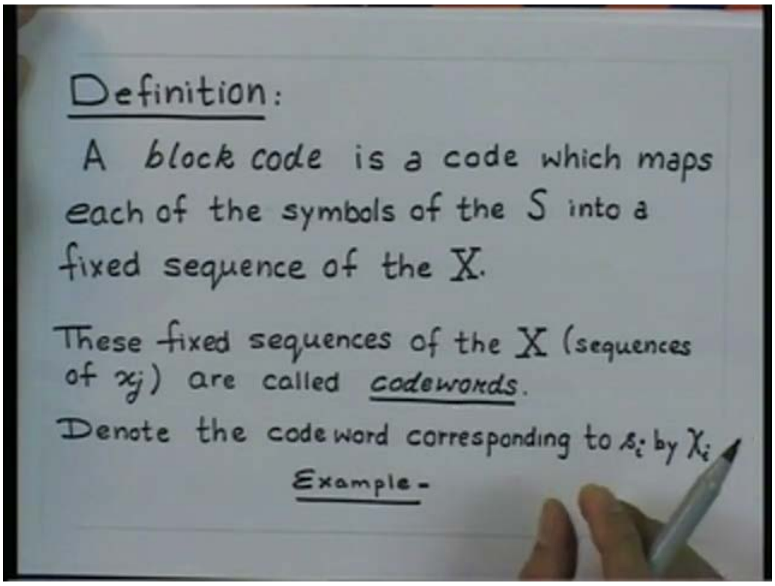
(Refer Slide Time: 08:00)



So, definition of code will be given as let us consider a set of symbols comprising a given alphabet and that alphabet is called as S that this alphabet S consists of q symbols or q letters. Then they define a code as a mapping of transformation or representation of all possible sequences of symbols of source alphabet S into sequences of symbols of some other alphabet x. This alphabet x consist of our symbols of our letters and this symbols or letters of this alphabet x are known as code letters or code symbols. We call S the source alphabet and x the code alphabet.

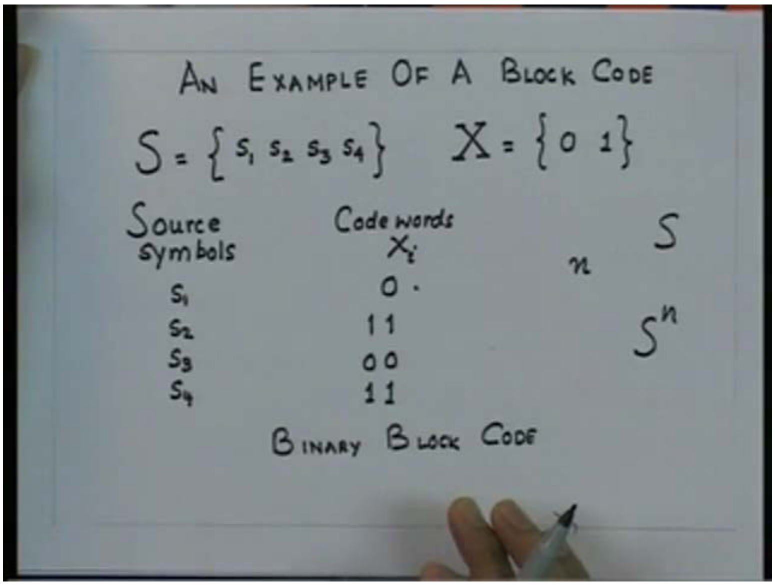
So, in short code is nothing but a mapping from a sequence of source symbols to a sequence of code symbols. Now, this definition of code is to journal to be of much use in code synthesis, so what it implies that we should try to put some more restrictions on the definition of code. The first properties which we require for a code to satisfy is what is known as a block code. So, let us look at a definition of a block code.

(Refer Slide Time: 10:20)



A block code by definition is a code which maps each of the symbols of the source alphabet denoted by S into a fixed sequence of the code symbols from the code alphabet X. This fix sequences of the code alphabet X that is sequences of x j are called code words. Each code word corresponding to a particular symbol in the source alphabet would be denoted by xi. So, xi is the code word corresponding to the symbol si in the source alphabet. Let us look one example to understand this definition.

(Refer Slide Time: 11:23)



So, an example of a block code, let me assume that I have a source alphabet consisting of four source symbols denoted by s 1, s 2, s 3 and s 4 I have my code alphabet X consisting of code letters or code symbols which are 0 and 1. So, this would be an example of a binary block code because the code alphabet consists of only two elements or two letters to code symbols, which are 0 and 1. So, let us design one code for this designing of a code for this source would be trying to find out the codeword's for each of these source symbols. So, the codeword's which are denoted by Xi would be for s 1 let me have 0 for s 2 let me have 1 1 for s 3 let me have 0 0 and for s 4 let me have 1 1 again.

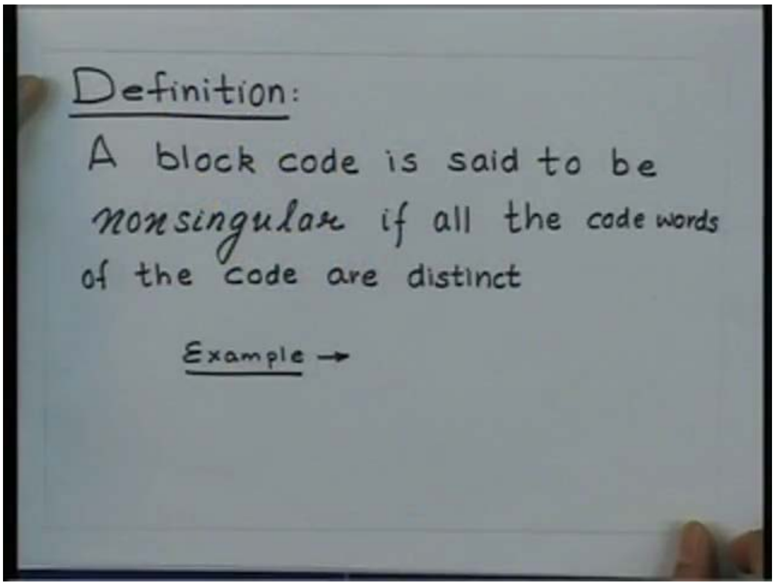
So, this are the codeword is corresponding to the source symbols this is the codeword for the source symbol s 1 this is the codeword for s 2. And similarly, this the codeword for source symbol s 4, this would be an example of a binary block code. At a first glance the requirement that the encode source symbols one at a time into fix sequences of code symbols, seems quite severe. Note however that if a code maps all sequences of source symbols of length n and that is small n into a fix sequences of code symbols, than the code maps each and every symbol from the nth extension of the source, S original source S into a fix sequence of code symbols with code alphabet S n. This property will see very shortly in the light of this property, we will realize that this definition of block code is not that severe.

A set of rules transforming a source alphabet into a code alphabet, we will satisfy our definition of block code only when we consider the symbols from the nth extension of the source S. Now, if you look at this code which I design this would be the this I will call as a code with a particular code for this source S and this code has been formed based on this code alphabet 0 and 1, and these are the code word.

Now, let us if you want to really use this code in a practical situation there some problem is that if you look at the code words for the source symbol s 2 and s 4, they are identical. So, even if I consider messages of length one that means messages consisting of source symbols of length one. And when I received the code symbols 1 1 then the ambiguity for me to decode the value of the source symbol 1 1 could correspond either to s 2 or it could correspond to s 4.

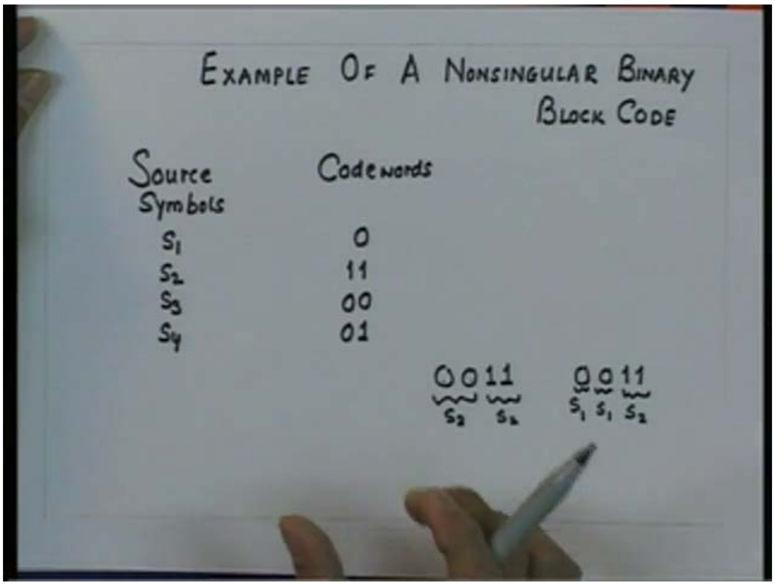
What this example implies that we should put further restrictions on block codes. The first natural restriction which we should put for a block code is all the codeword, which you form for a code should be distinct. So, a block code should be formed in such a way that all the codeword is in the code are distinct, and such a block code is known as a non-singular block code. So, let us formally define a non-singular block code.

(Refer Slide Time: 17:37)



So, definition of a non singular block code would be a block code is said to be non-singular, if all the codeword of the code are distinct. And example of such a non-singular block code would be.

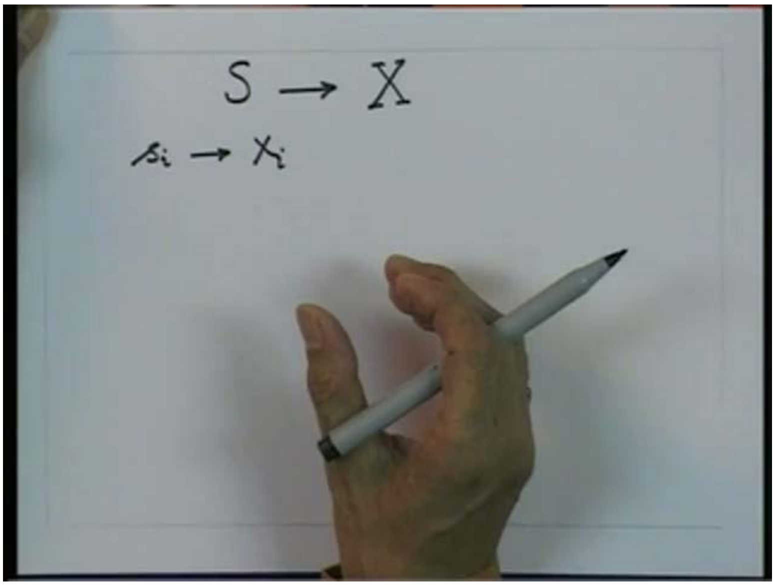
(Refer Slide Time: 18:01)



Let me again consider source symbols s 1, s 2, s 3, s 4 corresponding to the source symbol, let us assume that the codeword is are 0 11 00 01. Now, this code satisfies the definition for non-similarity of a code because all the codeword's out here are distinct. But again there is some problem with the design of this code, take a simple example if I were to receive a sequence 0 0 1 1 then when I receive this sequence. I can decode the sequence as s 3, s 2 or I could decode the sequence as s 1 followed by s 1 or followed by s 2 so on the received of the sequence the code symbols 0 0 1 1 there is an ambiguity about the transmission of the source symbols.

So, though this quote was not singular in the sense when I considered the source symbols as individual entity, but when I consider the transmission of source symbols in block, then there is an ambiguity in the decoding. So, it means that we should put further restrictions on non-singular block codes, in order to arrive at the codes which are more useful. Now, in order to do that let us try to define what is known as nth extension of a block code, we have defined nth extension of a source S. So, let us try to define in a similar manner an nth extension of a block code.

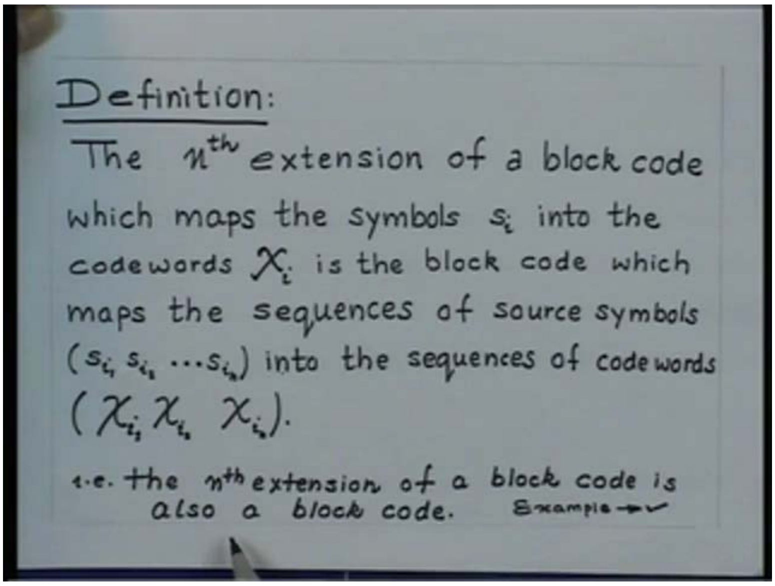
(Refer Slide Time: 21:30)



So, in order to do that let us consider I have a block code which maps all the source symbols from the source alphabet S into a fix sequence of core symbols from a code alphabet X. Now, this soul itself s could be another extension of some others source. It is not of importance to us since we are restricting our discussion to block codes, then we have a natural elementary unit.

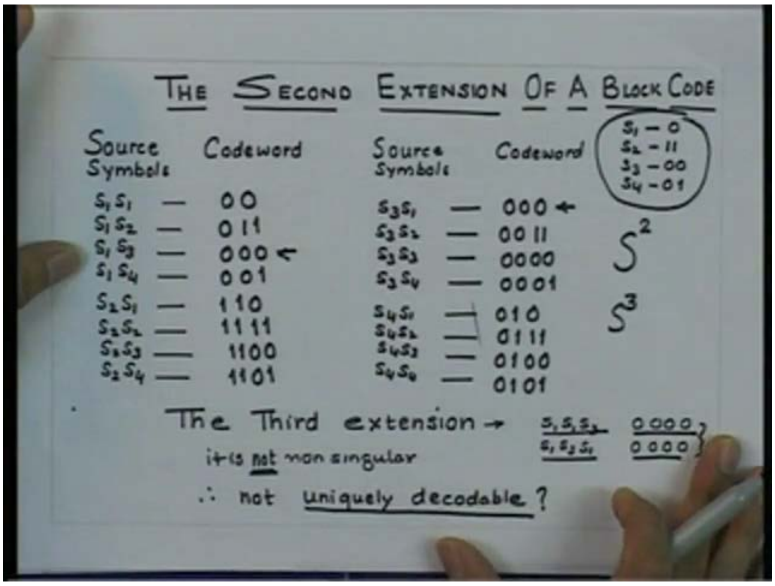
Namely for each symbol in S we have fixed sequence of code symbols given by a particular code Xi, this forms one unit. Now, putting this building blocks together just as we did for nth extension they were successive symbols from elementary source, were put together to define nth extension of a sources. So, similarly, we can define nth extension of a block code based on this elementary unit. Let us try to define nth extension of a source of a block code.

(Refer Slide Time: 23:20)



So, the nth extension of a block code which maps the symbols Si into the codeword's Xi is the block code which maps the sequences of source symbols si 1, si 2 upto si n into the sequences of codeword xi 1 xi 2 xi n. Xi 1 is the code word corresponding to the symbol si 1. Similarly, Xi 2 is the corresponding codeword for s i 2 and finally, xi n is the corresponding codeword for the symbol si n. So, you form the sequences of source symbol of length n then to get the sequences of code symbols for this sequence is just concetta the codeword is the corresponding code words. And that is how we form, the sequence of the code symbols for this. Now, by definition that nth extension of a block code is also a block code, let us try to understand this concept with the help of a simple example.

(Refer Slide Time: 24:58)



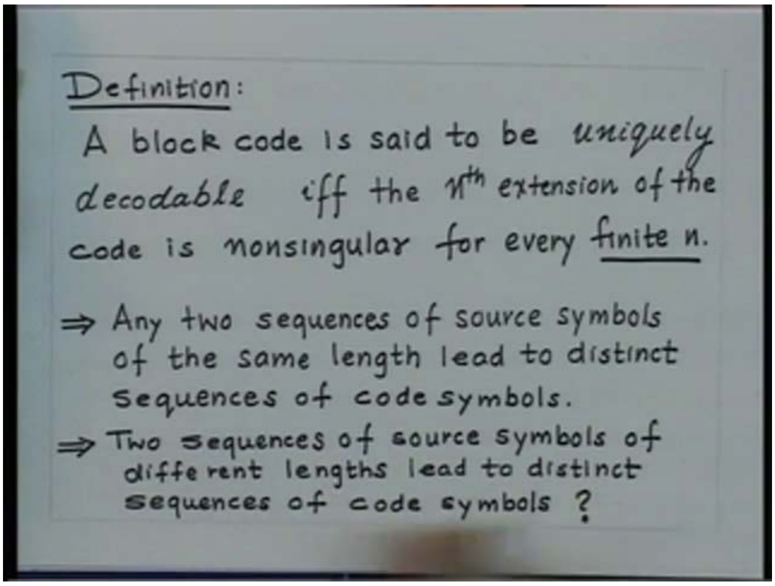
Let us again consider the block code, which we had designed earlier this is a block code binary non-singular block code which we had talked of so I have four symbols and I have four different codeword's for it. So, this is an example of non-singular block code. Now, the second extension of this would be the second extension of the source s 1, s 2, s 3, s 4 will comprise of sixteen symbols. So, the second extension of the source s 2 will comprise of sixteen symbols which are being given him listed here. Now, to form the code words corresponding to the sixteen symbol is very easy look at the code words for each symbol and concatte them.

For example, take s 1 s 1 the codeword corresponding for s 1 is 0 to the code word corresponding to s 1 s 1 is 0 0. Similarly, s 1 s 2 would be 0 1 1 s 1 s 3 would be 000 s 1 s 4 would be 0 0 1. Concatenations of the codeword's respective codeword is give us the codeword for the extension so s 3 is 0 0 s 1 s 1 s 3 is 0 0 1 1 2, this is the code which generate for a second extension. Now, if you look at this code which we are generated for the second extension by the definition of nth extension of a block code, we will find that the second extension this code which we are generated is not a single is not non-singular. Because if you look at s 1 s 3 and s 3 s 1 the code words for both this are identical. Now, if the code word is identical for the source symbols then by definition this is not non-single.

Similarly, if you take a third extension of this source will have 64 combinations and one of combination you will get s 1 s1 s 3 and for that the code word will turn out to be 0 000 and for s 1 s 3 s 1 it will turn out to be 0000. So, again you will find that this is one example that could be many more like this when you try to go the third extension of the block code.

Now, this is for this case you will find both the code word are same for two symbols in the third extension of my preliminary source given earlier. So, what it means that if you want the code to be uniquely decodable it means that, when I consider symbols from the nth extension of the source then the code word is generated based on the principle of nth extension of block code should be non-single. So, with this let us define uniquely decodable non-singular block codes.

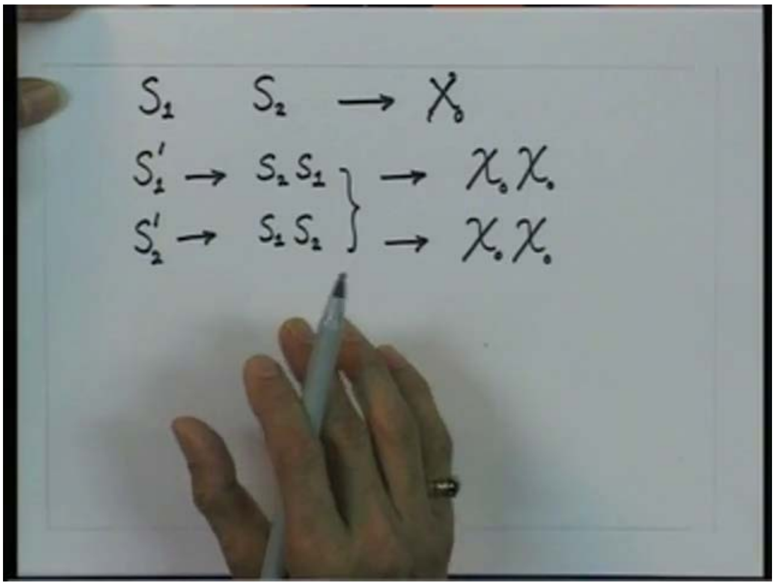
(Refer Slide Time: 29:32)



So, block code is set to be uniquely decodable if and only if the nth extension of the code is non-singular for every finite n. In the previous example, we saw that when we take the second extension of the block code there are two code words, which are same for two symbols from the second extension of the source. So, that condition with this condition is not satisfied then that the non-singular block code cannot be uniquely decoded.

Now, once if I have this definition uniquely decodable codes, where the nth extension of the code will not single of every finite n what it implies that any two sequences of source symbols of the same length will lead to distinct sequences of code symbols. This is not very difficult to see because it is by definition of uniquely decodable, but we also desire that two sequences of source symbols of different lengths lead to different sequences of code symbols. Now, this is not very clear from this definition so let us try to prove this by contradiction.

(Refer Slide Time: 31:13)

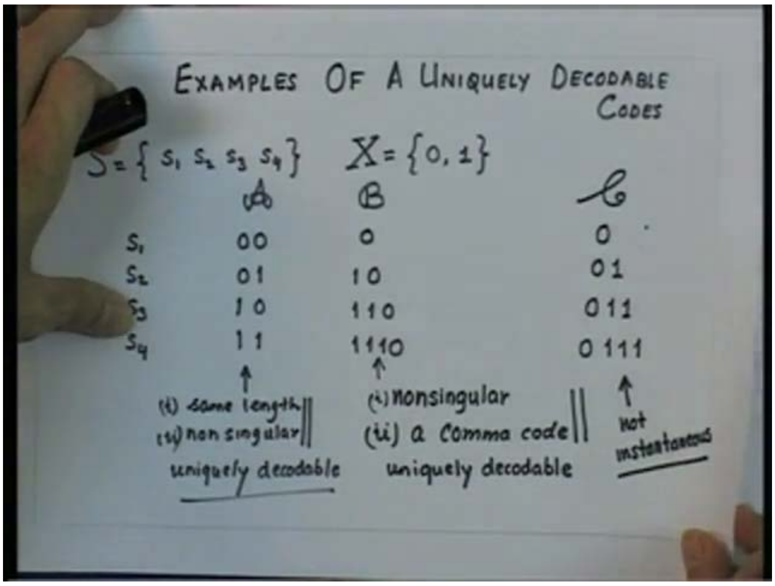


So, let me assume that there are two sequences S 1 and S 2 which gave rise to the same code word that is called that is X 0. So, both these sequences S 1 and S 2 they could get the same length or may not be the same length give rise to the code word X naught, let us assume that is same for both. Now, let us have another sequence which is S 1 prime and that is obtain by S 2 followed by S 1 and let us have another sequence S 2 prime which is S 1 followed by S 2.

Now, both the sequences are obviously of the same length even if S 2 and S 1 are not of the same length. Now, the block code if you assume that the code of block code then the code the sequence of code symbols corresponding to S 1 prime would be nothing but X naught X naught. And the sequence of code symbols corresponding to a sequence is to prime would be again equal to X naught X naught. Now, if we assume that the code is uniquely decodable than what happens is that for two different sequences I am getting the same code words.

Now, this is in contradiction to a assumption that the code is uniquely decodable for uniquely decodable codes, you cannot have two code words of to be the same for the same length of source symbols. So, what it implies that if you have a uniquely decodable codes then sequences of same length, or either different lengths will give rise to sequences of code symbols, which are different. Let us take an example of two uniquely decodable codes, so let us look at the examples of uniquely decodable codes.

(Refer Slide Time: 34:16)



Again I consider that I have a source alphabet consisting of four symbols, I have my code alphabet which is binary and I am interested in a binary uniquely decodable codes. So, let us say I have s 1, s 2, s 3 and s 4 and let me have the code words is as 00 01 10 11. So, this are the code words for this symbols. Now, let me consider another code found based on this code alphabet, which is given as 0 10 110 1110. So, let me call this code as code A and the other code as B, if you look at the code A the code words have some unique properties first of all the code words, in this code are you non-singular.

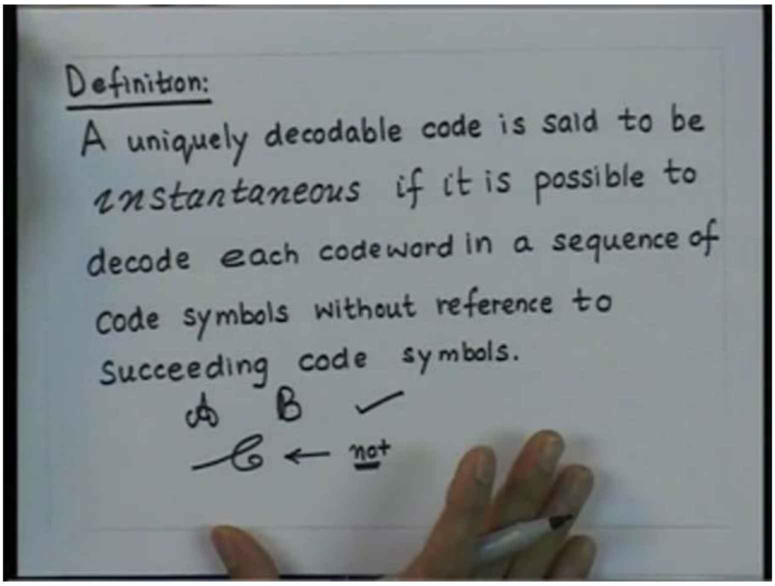
Second all the code words are of the same length, now if this two properties are satisfied than so they are the same length and it is non-singular, if this two properties are satisfied than it is sufficient to ensure that the code is uniquely decodable. Let us considered the code B if we considered the code B, again it is non-singular and it is a comma code. What I mean by that is that 0 x as a comma to separate a one code word from the next.

So, this again is implies that code B is uniquely decodable from both this example what it appears that the ability to tell, when a code word immersed in a finite sequence of code symbols comes to an end, can be seen to be central to the construction of both types of uniquely decodable codes discuss here. So, this property is at the heart of all uniquely decodable codes, lets us take another example of uniquely decodable code. Let us say that I have code c where I have my code words is given as 0 01 this code c is again non-singular. And this is a uniquely decodable code, but there is a difference between this code and previous to codes A and B, the differences is that when I receive the sequence of code symbols, then am unable to decode the sequence of code symbols code words by code words as it is received.

For example, let us say I have received some sequence of code symbols and I start scanning, and the first two symbols which are received are 0 1. Now, when I receive 0 1 at the instant of time and the time instant I receive one I cannot decode as s 2 at the time of instance I have to wait and find out what is the next code symbol, because in the next code symbol is 1 1 then what it means that it could be either now s 3, it is possible that it could be s 4 because when I received the symbol s 3.

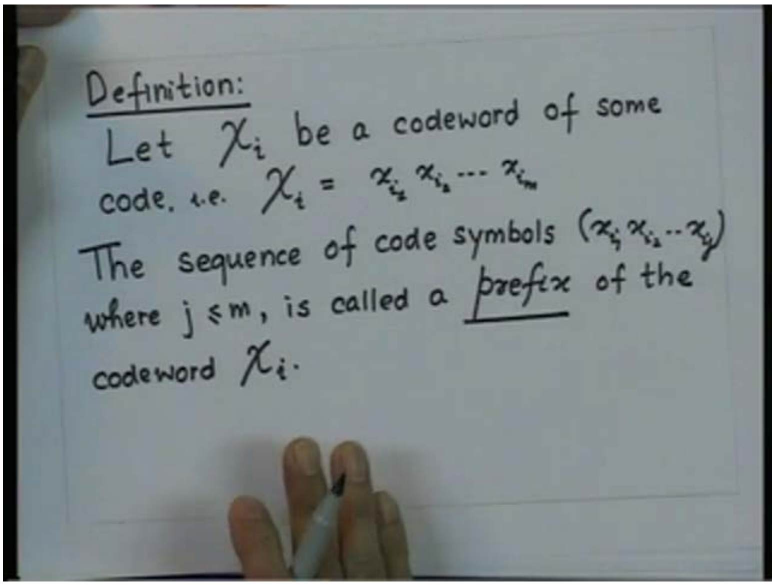
Now, after this s 3 if I receive 0 then I can say that 0 1 1 is s 3, but if I receive 1 than I have to say that it is as s 4. So, what it means that when I have this code, I cannot decode code words by code words as it is received there is a time lag in the decoding process I have to look forward to the succeeding code symbols before I take a decision to decode. So, this is a distinction of the code c from code A and B though this is uniquely decodable, but this there is a time lag in the decoding process.

(Refer Slide Time: 42:14)



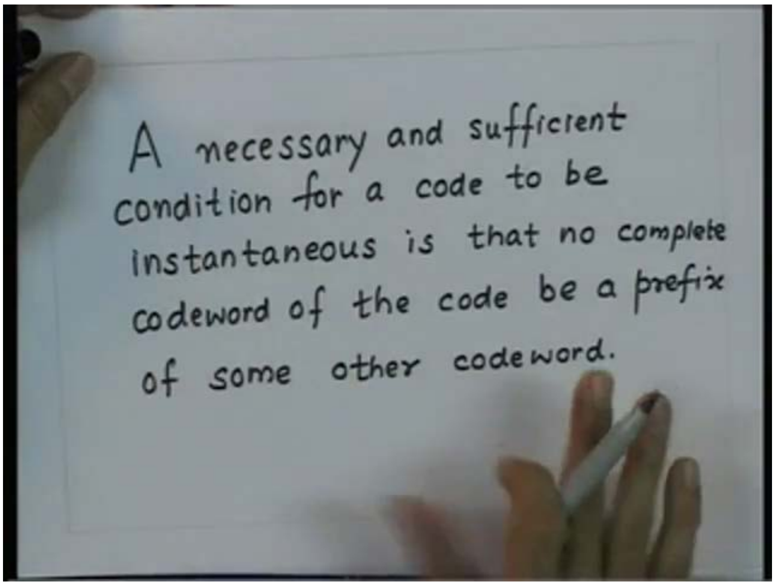
So, what we say that is this code is not instantaneous. So, let us try to define what is an instantaneous code. So, definition of an instantaneous code would be a uniquely decodable code is said to be instantaneous, if it is possible to decode each code word in a sequence of code symbols without reference to succeeding code symbols. So, in the light of this definition code A and code B both are instantaneous whereas, code c is not instantaneous. Now, it was very easy for us to find out, whether a uniquely decodable code is instantaneous or not instantaneous in this 3 examples. But the question is it a possible to find out or devise a general test based for a code, which tells when a code is instantaneous or not instantaneous. In order to do that let us define something what is known as a prefix of a code.

(Refer Slide Time: 45:06)



So, let us define a prefix of a code let X i be a code words of some code that is X i consists of sequence X i 1, X i 2 to X i n chosen from the code alphabet X. The sequence of code symbols given by X i 1, X i up to X i j where j is less than equal to m is called a prefix of the code word X i. Now, based on the definition of a prefix of a code word X i we define a test, which will tell us whether code is instantaneous or not. So, that test may be stated formerly as…

(Refer Slide Time: 47:25)



A necessary and sufficient condition for a code to be instantaneous is that no complete code words of the code be a prefix of some other code word. If this condition is in this test is satisfied, then we can say that a code is a instantaneous uniquely decodable non-singular block code. Now, what it means here is if you take the example which we had considered to look at this you will find that the code c does not satisfy the prefix condition, because the code words corresponding to the symbol s 1 forms a prefix to the code words of s 2.

Similarly, the code words s 2 use a prefix of s 3 and s 3 is a prefix of s 4 in fact s 2 s 1 is prefix of all the code words. Similarly, s 2 is a prefix of s 3 s 4 and s 3 is a prefix of s 4. So, by definition this is not a instantaneous code because it does not satisfy the tests of prefix. Now, will try to see whether this test is really a necessary and sufficient condition for the code to be instantaneous in the next class.