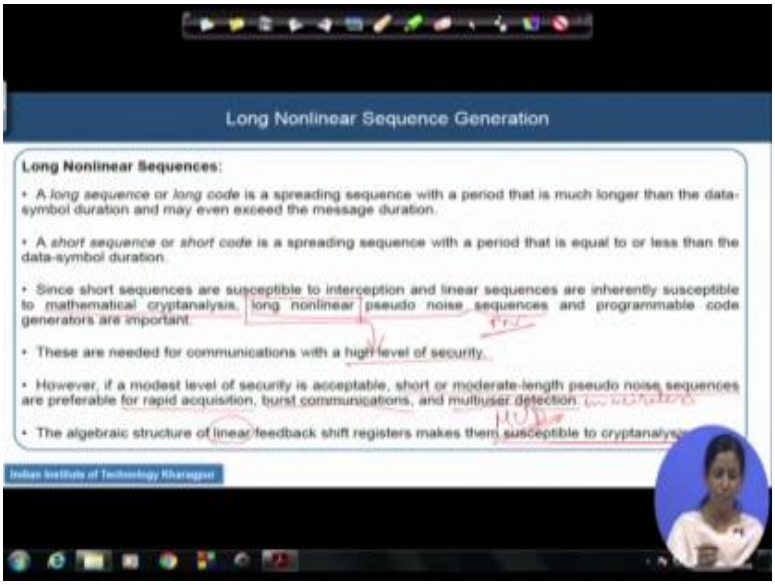
Spread Spectrum Communications and Jamming
Prof. Debarati Sen
G S Sanyal School of Telecommunications
Indian Institute of Technology, Kharagpur

Lecture – 28
Long Nonlinear Sequence Generation

Hello students, today we will discuss the generation of long non-linear sequence. It is important that we will discuss little bit and their usage in specific term communication system needs to be understood. Why we are suddenly coming here because almost with the background of the understanding of the Galois field and polynomials over the Galois field, the generation mechanism, understanding of the characteristic polynomial.

Now, we are in the situation of understanding the way the codes are generated. And there are two different kinds of the codes that are usually used in specific term communications. There may be short length code, they may be linear code and they may also be long sequence code and long non-linear sequence code. So, let us see today what are the corresponding advantage-disadvantages are of all these different type of the codes.

(Refer Slide Time: 01:38)



Long sequence code or the long in a spreading sequence, they are defined in a way that the period of such codes should greater than much greater than the symbol period. Then we can say that they are the long sequence codes. Whereas, the short sequence codes or

the short codes are defined as the period having the period which is much less than the symbol duration.

Data symbol duration see if you are having short sequence code then it will be very easy to intercept that is the jammer can usually very easily crack the code by observing your code pattern over a small duration of the time only. It is just a matter of time for observation, if I observe this for a pretty good mount of the time easily the pattern I can regenerate, I can make some assumption of the pattern, and then easily it can be regenerate. And once actually the code is cracked and if I get the code definitely it will be very easy for a jammer to jam the communication system. Now it will be very easy also to extract your own information that you are transmitting on air and also to jam the information so that receiver does not get intend receiver does not get the actual information. So, here, that is a problem of having a short length of the sequence.

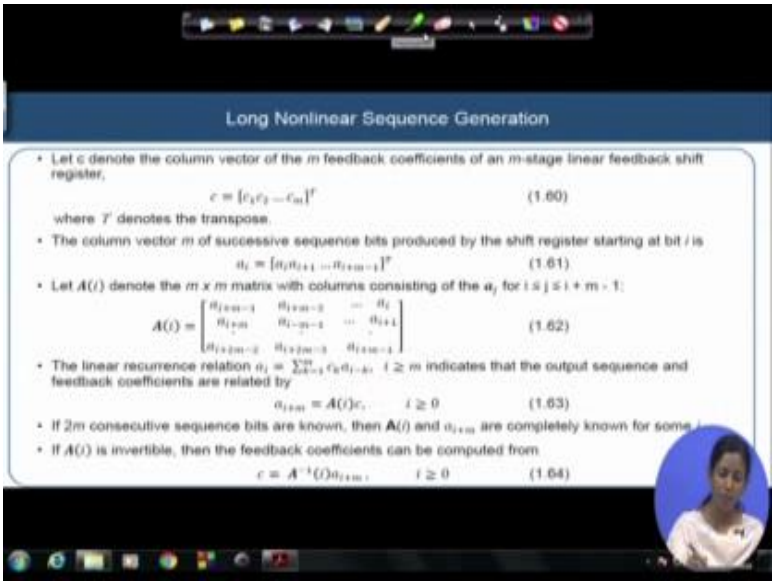
What is a problem of having a linear sequence? If it is a linear sequence then they are inherently susceptible to the mathematical crypt analysis. So, the hackers who are also actually our point of interest in such kind of communication system is and especially today then this susceptibility towards a mathematical cryptanalysis is not a desirable criteria, it is not at all desirable for the system design. So, we are having both the problems with linearity of the course also having a problem of having short code sequence length. In order to avoid, both we usually prefer to go towards the long non-linear pseudo noise sequences, or the PN sequences and the programmable code generators that which are very important now to understand.

Remember this long code sequences with nonlinearity involved in it. They give a very high level of the security involved now it is the designers choice based on the level and the priority of the security demanded in the design system how actually will choose and will do the compromise between the length of the sequence and the nonlinearity involved in the design. So, if the security is acceptable then short or moderate length pseudo noise sequences also we try to utilize, because if the length of the sequence is really long you cannot utilize it for a very rapid acquisition. for long length your acquisition synchronization acquisition in the receiver also need some more time, so convergence of the acquisition algorithms that they are not so fast.

If you are going for towards a security then you have to have a compromise over a long acquisition time, but if you are fine enough with the moderate rate of the security then it is a short or moderate length pseudo noise sequences are preferable for very rapid acquisition, burst communication, as well as the multiuser detection in a wireless communication scenario. The multi user detection or mud is a very well known research topic where the design of the sequence is a most important research criteria of our developing, for establishing a good outer wireless communication system.

The algebraic structure that till now we have learnt for this linear feedback shift register, the name itself linear signifies that the name itself the linear signifies that the linearity is involved in the whole structure because this is only the exorgate involved. If you remember that they only the exorgate is involved in the feedback logic and, hence, it is very easily susceptible to the cryptanalysis. So, this structure till now that you have studied is not enough to generate in very high secured code.

(Refer Slide Time: 06:45)



So, let us see some the criticality in terms of the mathematics also. Says, let us see denotes, some column vector of the m feedback co-efficient. So, these are the co-efficients that we have designed and small m is the number of stages involved in the linear feedback shift register and these are the column vectors and so, you understand that the way we having the high speed form of the linear feedback shift register these are the switches the location of the switches, the typical state of the switches whether it is

equal to 1 corresponds details, it tells us that typical stage is contributing to the feedback logic if its value is equal to 0 it is not contributing. The T, here, is defining the transpose.

Now the bits that are sequence bits that are produced by the shift register let us write the column vector of those m successive bits that are coming out is given by the vector a i where a i will be the a i and a i is coming up with the a i plus 1 a i plus m minus 1 dot dot dot. Now, a i is a vector, so it will be bold.

And now you let us understand that capital A i is a m cross m matrix; where in each every column that we have put it consist of a vector a j that a j is looking like this. So, it is transpose means it will be turned and then, hence, it is written in the column format, but this j value starts from the i and it goes to i plus m minus 1.

So, here if i start with the j value is equal to i plus m minus 1. If I start then slowly I will be ending up with the value of i plus i plus m minus 1. This value of i should be replaced by i plus i plus m minus 1 and, hence, there for last value will be i plus 2 m minus 2; what I am doing, I am just shifting I am just replacing and substituting the value of i by the value of i plus m minus 1 and, hence, I should a end up with a i plus 2 m minus 2.

The next module or the next column sorry which would start with the I plus m minus 2 and if I go by another I plus m minus 1 step I will be ending up with i plus 2M minus 3 in this way if I start and I will be ending up with the value a i who will be where actually I will start and I will end up with I plus m minus 1.

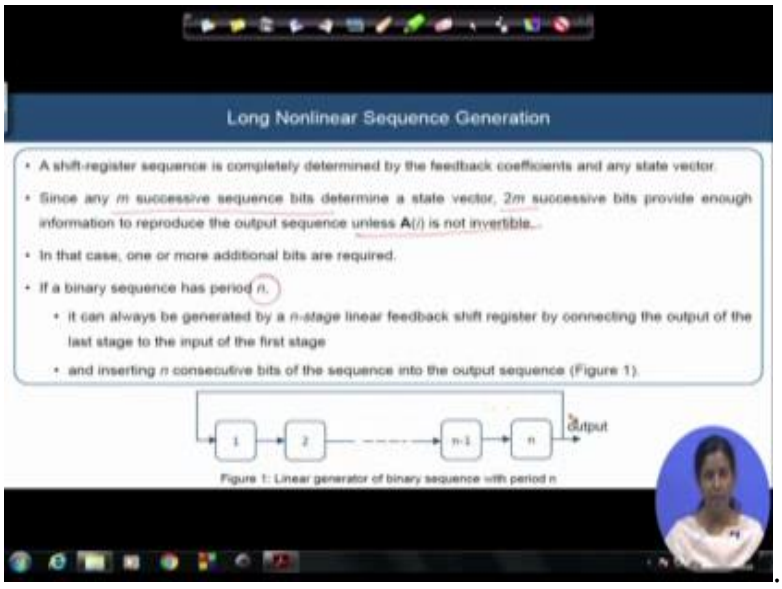
And you can see the sequence such a way that it is something like the invert of this a plus starting with first having the value of a i m plus 1 and then finally, seeing the last value seeing as an a i and, hence, the corresponding column values are coming up. So, fundamentally it is generating a m cross m matrix and a i is denoting that matrix. We understand and please recall that the output is having a linear recurrence relation given by this where the clock pulse value is greater than equal to the stage number small m and, hence, we from following this recurrence relation we can also write that any i plus mth value will have some a i multiplied by c and if this a i multiplied by c, this c is another feedback coefficients, it is multiplied by the feedback coefficients and you will get the value of a i plus m the stage at the output.

Now, as this matrix this is of m cross m and if I see the twice m consecutive sequence bits and if it is known then a i and a i plus m is small i plus m. They are completely known for some value of the i and if believe that this metric is invertible then you can easily get what should be the setting of this coefficient c by this formula. So, now what is the situation?

That you mean if the sequence length is short if you observe it over twice the m bit period versus m sequence and the bit period. So, you will have a complete knowledge of this c i and for some value of the i and if you can if this matrix is invertible then definitely I can get the setting of c matrix the setting of c matrix is nothing, but the setting of the feedback path.

So, now you know the whole everything about the design. The way the key is generated what are the typical c values to understand the whole capital matrix you have a relation of the initial stages, so you can reproduce the code easily.

(Refer Slide Time: 11:51)



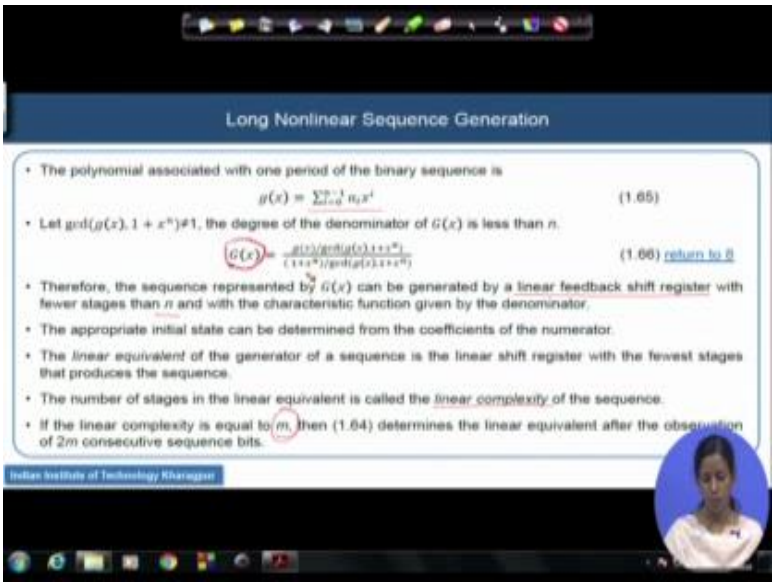
Shift register sequence is completely determined by the feedback coefficients and any state vector say I have the small m successive sequence bits determined by the state vector by a state vector. And 2 m successive bits provide the total information about how to reproduce that output sequence once again unless A i is invertible. So, if A i is not unless A i is not invertible if A i is invertible then this information alone is sufficient to give you the whole information to regenerate the circuit and regenerate the code. And in

that case actually hardly one or more additional bits you may require for you to understand if your A i is not invertible also.

So, for a binary sequence which is having a period of say n that can be is always generated by some n stages linear feedback shift register by connecting the output of the last stage to the input of the first stage as shown in this figure one, and by inserting the n consecutive bits of the sequence into the output sequence that we have shown here. So, this is the very fundamental stuff that if I understand that there are the nth stages of the feedback because there is a period of equal to n so if put the n memory stages here connect output to the input directly and if I know the insert the n consecutive bits of the sequence that we observed here, so automatically the sequence will be generated.

And only part is that you are involving the large number of the stages that is all, but cases of the if the jamming very important your target is to jam somebody, you will not mind putting the large amount of the hardware to regenerate the circuit you need not to generate may be actually in the transmitter this sequence was getting by generated by only 3 stages of the shift registers. Here you are involving all n stages of the shift register that will be the difference only.

(Refer Slide Time: 14:09)



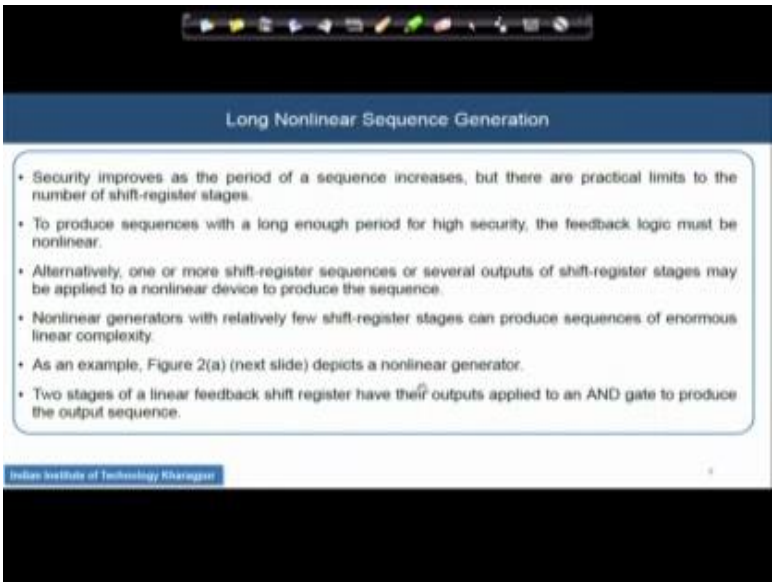
Now, the polynomial which is associated with one period of the binary sequence, the generating polynomial which is associated with that we have seen earlier this is given by A i x to the power i. Now, we also understood that the degree this capital G x which is

the generating polynomial which is given by also G x by 1 plus x to the power n in the last module and if we find see that the greatest common divisor of this G x and 1 plus x to the power n it is not equal to 1. So, then that degree of the denominator here of the G x will be always less than n and, therefore, the sequence that will be generated by this generating polynomial G x. It can be generated by a linear feedback shift register also with very few stages of the n.

So, the condition is that that if there does not exist a greatest common divisor between the G x and 1 plus x to the power n. And the degree of the denominator degree of the denominator is less than this n. Then, actually the generation then the sequence that is getting generated by this generating polynomial they can also be generated by the stages which is less than n. So, shift register stages that you are involving to generate the sequence will be less than small n. And by appropriate initializing the state we can be determining the coefficients of the numerator also and the linear equivalent of this sequence is always call of a any linear shift register sequence with very few number of stages. We call it is a linear complexity of the sequence and if the linear complexity is equal to the small m then the linear equivalent of the observation of 2m successive bits can be observed by the expression in the last, we have seen you have shown in equation 1.64.

So, the fundamental part is that if you see that the generating polynomial which is given by G x plus 1 plus x to the power n is not having any gcd between G x and 1 plus x to the power. Gcd is not equal to 1 and you see that the linear feedback shift register with very fewer stages than n can be generated and utilized to generate the sequence. If we are having the value of this n, if that degree of that polynomial 1 plus x to the power n is less I mean the denominator staff is less then we can generate is with fewer number of the stage and because of that which is defined further by the linear complexity also.

(Refer Slide Time: 17:06)

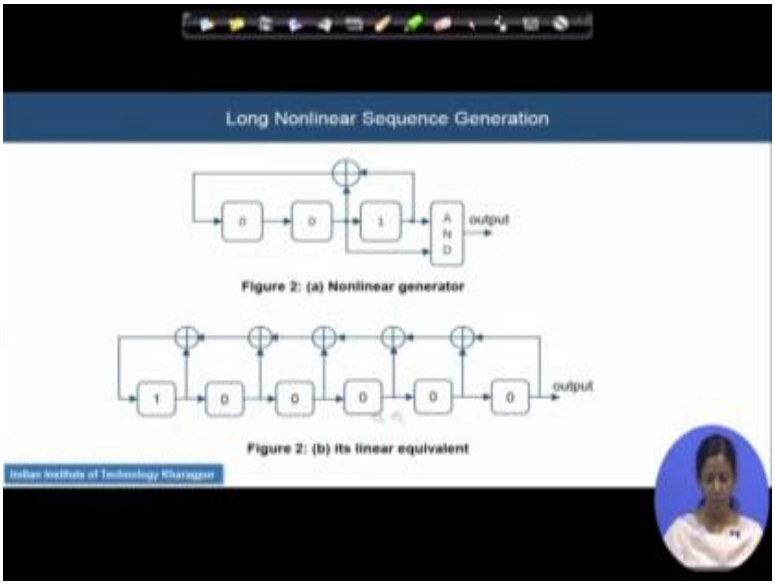


Look, one part is really clear if we increase the length of the sequence; that means; a period is increasing period means the length of the sequence is increasing. If we increase the length of the sequence definitely the security increases, but there are some practical limits to the number of shift registers stages that we can utilize for the generation of such sequences. So, involving a huge number of the shift register sequences in the circuit to generate a long sequences not at all practically advisable assistance design. So, what we can do is, actually we can equivalently generate the non-linear actually; we are trying to improve the secrecy by increasing the length. But equivalently we can improve the secrecy if you are giving the feedback non-linear. So, better you do not go improve the security by means of a length you improve the security by means of improvising the nonlinearity inside the feedback circuit. So, it will also serve the same purpose.

And otherwise you do one thing actually. So, output of the two or three stages that can be also feedback before taking the output directly from the last stage. We can actually take the output of the two or three stages and then we can add them into non-linear circuit and non-linear device which will from the output of the non-linear device. We can take the output of the non-linear device can be the final output: I mean. So, either way either you do the non-improvise nonlinearity in the feedback path or you improvise nonlinearity by combining any one or two stages of the feedback shift registers.

Feeding them inside a non-linear device and take the final output from the output of that device, we will see this kind of the structures. Both the structures where actually multiple output stages will be feed to a non-linear device may be here you have taken an example where we will feed it to an, and gate to produce output sequence and another non-linear generator which is coming next.

(Refer Slide Time: 19:32)



This is the circuit and non-linear generator is shown in the figure number 2 a, where we are having three stages- this is values is kept inside here is nothing, but their initial values. And this was the well known structure as we were discussing or since beginning that the stage number 2 and stage number 3 is contributing in the feedback path. We have improvise nonlinearity in order by changing the feedback architecture by, but by taking the output of the stage number 2 and stage number 3 and feeding it to an and gate and final output is taken from here.

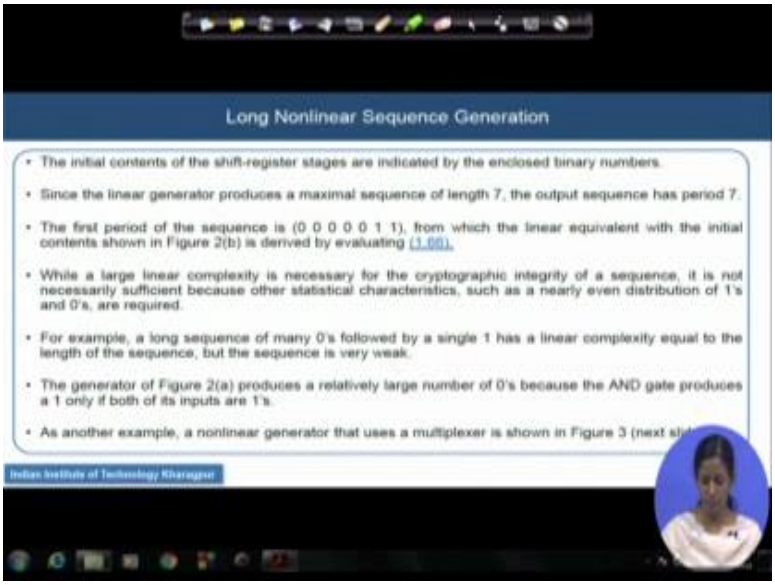
If you see actually the structure of the code that is getting generated at the output of the AND gate and you will be able to see that the sequence that is getting generated is a sequence that is generated that can be also be produced by its linear equivalent; where the total number of the stages that is getting generated is equal to 7 from here, you will be able to generate 7 stages and here actually we have kept all the 6 stages n minus n number of the stages the repetition is 7. So, here, he will keep 6 number of the stages; here and all the stages if I linearly combine in the feedback path and if we feed it back to

the initial stage, we will be able to see that both the circuits are generating the same sequence.

So, idea is that instead of giving all the very big number of the very big circuit and involving huge number of the stages to generate a non-linear sequence. So, we can come back to a minimization of the number of the stages involved by adding one nonlinear device.

Please, do this exercise at home first generation see the sequence generated here and also by default by giving this circuit with this initial situation try to see actually what exactly is the sequence, if you are getting and if you see that both the sequences are same then definitely this both the structures are also equivalent.

(Refer Slide Time: 21:46)

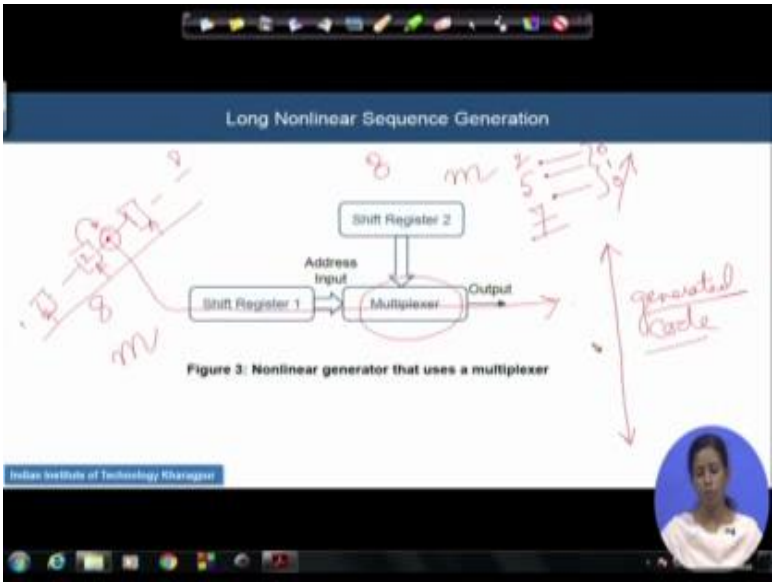


So, these are some explanations that I have already told earlier see when a long linear complexity when the larger linear complexity is definitely required. Because, if we wish to give a high value of the security for the cryptographic integrity of a sequence. So, remember it is not true always that if you are having a very long sequence and such a very even distribution. Suppose there are only the long sequence and it is and it does not mean that only having a very long sequence it will improvise actually the very good amount of the linear complexity involved. It should have some other properties also for example- the number of the 1s and the number of the 0s that you are generating in that sequence there should be very well balance between the 0s and 1s. So, contiguous 1s and

contiguous 0s that should not come also, the statistical property of that sequence is also very important when we are talking about the linear complexity.

And the sequence will be very weak see for an example, if a long sequence have so many number of the 0s followed very single one and then again actually long number of the 0. So, the linear complexity may very high, but this signal is very weak very easily it can be cracked. So, another example of this generation of long non-linear sequence, we will take in the next slide let us see.

(Refer Slide Time: 23:32)



So, here the idea is, suppose, you have a two different set of the shift register sequences. The shift register sequence one may have let us first consider that both are having the same number of the stages are equal to same initial conditions are not same; and what you are doing if you are involved a multiplexer in the code generation process.

On the shift register output of the first one or if there is a shift register number two, he is giving fast some code sequence. So, the output you have taken trapped the output from certain stages of this shift register architecture and whatever the bit that are getting generated for a typical clock pulse, you have taken it and you are reading it as he added address. So, that address the multiplexer will send to the shift register one architecture according to that address the bits that should be picked up from the shift register stage and that will be directly connecting to the multiplexer and coming out as an output for example.

Suppose this value of the m is equal to say 8, for example, and we have tapped the value from stage number 2 say stage number 5 and say stage number 7 and at a particular instant of when the clock pulse has arrived I am seeing the values of 2, 5 and 7 th stage output is such that it is giving a value of 0 1 0. In this shift register also there are having the 8 number of the stages. So, the shift stages are like this going for 1 to say number 8 and this is the stage 0 1 0 that is the stage number 2 value is getting asked for. So, the step will be from the stage number 2: whatever the value at this movement when the clock pulse came here and when you gave the clock pulse here whatever the value is generated at the output of the stage number 2 now, it is getting connected directly to the multiplexer and coming out as an output.

So, if you keep for which an every clock pulse you will keep on changing, you will keep on generating the new addresses definitely. Because based on a initial condition of the initial state condition of the shift registers you are generating these values, is it clear. So, whenever, you are generating this values new values you are generating, based on that either the stage number 2 or stage number 1 or stage number 3 or stage number 7, you are picking up the values from the different stage outputs. So, the generation of the address of the stage is done by the first shift register and the typical output is getting taken from the second shift register architecture is an instantaneous value that is getting generated is taken out as a code bit. So, if I now observe over, say pulse over a pulse of is 10 or 15 I will get a series of the bits coming out from the different stages output of the shift register 1 and that is my generated code.

So, now the good part of it is, whenever, your generated code is like this and its period is needs to be controlled you can actually do that easily because it may happen that either you are not taking the variable address for the variable 1 or you are may have a situation; where you the address whatever you are generating it is always equal to 2 so always you are getting stuck to, but for each and every clock pulse of value output of the 2 is varying, hence, actually the generated code structure will be something else.

So, either you keep on sticking to the typical address in the shift register two and hack at and then being are typical stage output and on the arrival of the clock pulse continuous arrival of the clock pulse keep on checking that output and you pass that output. So, the multiplexer and generate the codes or for each and every clock pulse that is coming here and here for each and every clock pulse you keep on changing the address you make it

the make the address variable and then you shift actually the point where from the output should be picked up and that can be also be code.

So, whoever is whatever be the architecture require for the kind of the secrecy actually you can design it, accordingly. So, here you see instead of the AND gate the multiplexer is giving you the incorporating you the nonlinearity in the whole architecture. It need not to be done exactly the way I have explain this is just an example- how the (Refer Time: 29:12) with the help of multiplexer and to different shift register architecture you can do it may also happen that it can be vice versa I mean you can have a at one shot you are you can using you generating the address from here and picking up the value from here in the next cycle you are generating the address from the fast shift register and then picking up the value from the shift register 2.

So, that, what way you will combine is completely your design constraint, but this is one example and there are several examples- like that the way you can improvise a nonlinearity inside the circuit that will not increase the number of stages in the shift register sequence generation. But, it will definitely help you to improve the security level of the circuit, remember one thing whatever you are doing.

I mean shifting the logging into a single shift register and taking the value from here or you keep on changing the address for each and every clock pulse or even changing the shift registers where from the final output is taken out. But always check finally, the generated codes auto code relation as well as the cross code relation property because this auto code relation that we have learnt already that the auto code relation as well as the cross code relation properties and related power spectral density of the generated code. They are actually really very important issue in terms of the synchronization aspects and that code acquisition code tracking in the receiver circuitry you should not generate a code because you are trying to improvise high security do not allow and generate a code such a way that in its power spectral density.

There are several high value picks then you will the definitely you will get falls lock in the receiver. So, security will be very high may be, but recovering your own signal may be in under tie because you are enable to do any code synchronization in the receiver in if such kind of the generated code in the receiver. And, hence, your own detection process in the receiver will be hampered a lot.

So, there is a again you have tradeoff between the high secrecy involved and the level of the complexity that you are trying to improvise in your design towards the generate power spectral codes power spectral density and the peaks that are coming with respect to the main peak.

So, I hope that we have covered a lot about the sequences and the sequence generated long non-linear sequence generation was the last part of the code. Its discussion about the codes sequence spreading codes that are involved. In future I hope you will be able to generate your own course for designing such kind of the systems.