

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

Lecture - 41
DSSS Tracking

Hello students, we were discussing the synchronization in direct-sequence spread-spectrum receiver. And in that in the last few modules, in detail we have discussed about the different code acquisition techniques in the receiver. In view of that, we also discussed that synchronization in the receiver spread-spectrum communication receiver mainly consists of two different half halves; and first half is the code acquisition and second half is the code tracking. And remember whenever we are talking about the synchronization, this is related to the code synchronization we are talking about. Apart for the code synchronization in the receiver, there is separate carrier synchronization on the top of it, where you will find that carrier phase-carrier frequency synchronization will be required.

In most of our discussion and analysis, we have assumed that receiver has no clue and no a priori information about the carrier synchronization. Hence, in the most of the cases, we have discussed, we have seen the architecture we have discuss the architecture based on the non-coherent detector architecture. We have also shown that matched filter architecture can give us fast acquisition techniques and most of the case if the short length sequence is utilised for the spreading then matched filter can be a very good alternative also for your code acquisition than the conventional correlator and integrating tamed kind of the code acquisition circuits.

(Refer Slide Time: 02:36)

Pseudonoise Code Synchronization in Direct-Sequence Receivers

- One of the primary functions of a direct-sequence (DS) spread-spectrum (SS) receiver is to despread the received pseudonoise (PN) code
- This is accomplished by generating a local replica of the PN code in the receiver and then synchronizing this local PN signal to the one which is superimposed on the incoming (received) waveform.
- Multiplication or remodulation of the incoming signal by the synchronized local PN code replica then produces the desired despreading process.
- The process of synchronizing the local and received PN signals is ordinarily accomplished in two stages.
 - PN acquisition
 - PN tracking

Indian Institute of Technology Madras

So, with all that knowledge of the code acquisition, today our discussion topic is the direct-sequence spread-spectrum tracking that is the next half of the synchronization. We will quickly recap about the fundamentals of synchronization in the first two slides and then you will slowly enter into the direct-sequence spread-spectrum tracking mechanism. As I told the first two slides will be on the recap. We understand that synchronization is a very primary function of the spread-spectrum communication receiver, because without synchronization codes would not be guaranteed to be aligned with each other. Hence de-spreading process would not be perfect would not be satisfactory and if the de-spreading is not done in a proper way then your code and then at successful data recovery is not possible in the detectors.

So, in the discussion of that synchronization is a very first important block that needs to be performed in the direct-sequence receivers. And basically before de-spreading, we have to carry about that block; and this synchronization is basically code synchronization. Hence, your carrier synchronization will follow the code synchronization in most of the receivers we do we have like that. But I should mention here that there are some architectures synchronization blocks and architectures, where code synchronization block is intelligent enough to assume or estimate or predict the carrier synchronization aspects or the carrier phase and frequency, they can actually predict. And then they can actually there can be; with that information code

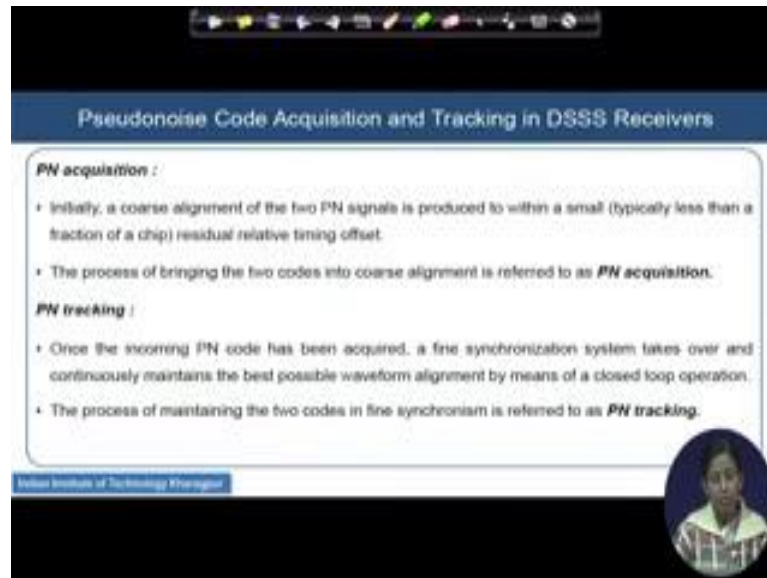
synchronization can proceed. But most of the cases, we will proceed with the fact that the carrier phase and frequency information is not known to us

So, your code synchronization is performed basically by multiplying the incoming signal with a local replica of the PN code. Remember this local replica of PN code is generated locally inside the receiver and so it is not coming along with the transmitter. And once we are multiplying the incoming signal with this locally generated PN sequence code, basically this is a remodulation going on of the transmitted signal or the received signal equivalently. And we expect that the PN signal which is super imposed on the data in the transmitted signal, now actually in this process actually they are that signal will be that PN signal. We will be now getting fundamentally multiplied with this locally generated PN code.

And hence we utilize at remodulation of the incoming signal by the synchronized local PN code. And this process of synchronization is basically aligned in the incoming code with the locally generated PN code; in such a fashion that the correlation between them becomes maximum and hence your code identification and the code location identification become perfect. They are perfectly aligned with each other, such that the output SNR gets maximized and then the remaining portion for the despreading operation can go ahead easily and satisfactorily and that such that the detective can perform, the detector can detect the message without an error.

The process of synchronization we have already discussed, there are many basically two steps. The first step is the PN acquisition and second part is the PN tracking. We have seen lot of the discussion on the PN acquisition already in the last few modules. And it is PN acquisition mainly based on the non coherent detector architecture as well as a matched filter based architecture for both mainly based on the direct-sequence spread-spectrum communication we have done.

(Refer Slide Time: 06:37)



Pseudonoise Code Acquisition and Tracking in DSSS Receivers


PN acquisition :

- Initially, a coarse alignment of the two PN signals is produced to within a small (typically less than a fraction of a chip) residual relative timing offset.
- The process of bringing the two codes into coarse alignment is referred to as **PN acquisition**.

PN tracking :

- Once the incoming PN code has been acquired, a fine synchronization system takes over and continuously maintains the best possible waveform alignment by means of a closed loop operation.
- The process of maintaining the two codes in fine synchronism is referred to as **PN tracking**.

Indian Institute of Technology Patna



In PN acquisition, we basically do the coarse alignment of the two PN signals incoming as well as a locally generated within a duration of the code chip. We try to keep the acquisition alignment miss alignment within the half of the chip duration. We try to keep it actually within the half of the chip duration, the miss alignment we try to keep within half of the chip duration. But at the end of this coarse estimation, which is called also the PN acquisition, they are we end up with the residual relative timing offset.

The remaining timing offset that is remaining actually we try to now correct in the PN tracking section. So, if I try to picturizing, it is something like one is the coarse acquisition, another is the fine acquisition; one is the coarse alignment, another is the fine alignment. Coarse alignment is related to PN acquisition and fine alignment is related to PN tracking. So, you cannot enter into the tracking stage directly without performing a coarse alignment of the two chips that is the logic.

And otherwise actually tracking will be very traverse and very complicated situation; anyway even if you try to club this two mechanism, you have to first align it within some within a gross residual offsets and then you can actually go ahead with the fine tracking with the fine alignment within that. And hence by default even if you go ahead with the combined circuit design, you will be ending up with first doing the coarse alignment then doing the fine alignment of the codes.

Remember once code acquisition is done fine tracking acquisition may be done once, but tracking is a continuous process. We never release actually that tracking mechanism to be turned off when the communication is going on. So, code acquisition may be roughly done once at the beginning of the packet arrival, but the code tracking will be continuously going on for each and every set of the data, every set of the series every set of the chips that you are continuously going on receiving. So, remember tracking is a continuous process; it is followed by the acquisition. It is a fine synchronization, it is a fine alignment between the two PN sequence coarse.

It is the method of collecting the residual timing offset that is left after PN acquisition, and usually the order or the range of this acquisition of the range of this offset lies between the half of the chip duration interval. And this process of maintaining these two coarse in the fine synchronization is basically called PN tracking which is a topic of our discussion today. So, this was a recap about the synchronization that the brushing up the fundamentals of synchronization and also the understanding once again the basic difference between PN acquisition and PN tracking, they are interdependency and also to a understand that what is the range of the error that we are trying to correct in between in acquisition and in tracking.

(Refer Slide Time: 10:15)



DSSS Tracking

Tracking DSSS signals:

- This is the process of keeping the receiver's locally generated code sequence in phase with the incoming code sequence.
- This is necessary for the receiver to correctly decorrelate the signal to extract the data.
- It also spreads any interfering signals, although proper synchronization is not necessary to accomplish that.

Delay Lock Loop (DLL):

- While there are numerous techniques for implementing DSSS tracking, most are based on the **delay lock loop (DLL)**.
- The DLL relies on the theoretical triangular shape of the correlation function of m -sequences.

Now, let us enter into the PN tracking mechanisms of direct-sequence spread-spectrum signals. So, we understand that the process of keeping the receivers locally generated

code sequence continuously aligned in phase continuously aligned in phase with the incoming code sequence is the process. And the process name is tracking. And why is it continuously needs to be monitored, because you cannot actually give up on the small changes in the frequency which may happen any moments dynamically because of the change of the environment because of the incorporation of the Doppler shifts and a lot of other issues in their communication path. And hence the continuous tracking is continuous monitoring is required. And continuously if some error is there, immediately it is correction is required to be done by changing the clock offsets in to the clock circuit so that again both of them become aligned.

And why it is so important because in the absence of that miss alignment, the data detection will be a under high amount of error which cannot be a till which mainly due to the drop of the package even because package error it will be so high that it would not accepted by the MAC layer next. This tracking also spreads any interfering signals although the proper synchronization is not necessary to accomplish it.

Remember that when we do the acquisition again we utilize the PN sequence, basically like your PN acquisition. When you do the tracking, we again utilize the same PN sequence. So, basically if some interferes are remaining after the acquisition process, you have a second chance of actually spreading that interfering signals once again inside the tracking circuit and then further (Refer Time: 12:26) effect of the interfering signals in the next despreading process before interfering into the despreading process.

The most promising and famous circuit for this direct-sequence spread-spectrum tracking is the delay lock loop, we call it the DLL circuit. And there are several techniques, so that implement that direct-sequence spread-spectrum tracking but this is the most popular one. And it relies basically on the fact that the m-sequences will give always a triangular shape after correlation. So, if it is a m-sequence getting used for spreading, and it utilizes a structure utilizes the fact that the correlation of m-sequence autocorrelation of two m-sequences as well as autocorrelation part of that m-sequences always turns down to be a triangular shapes. So, based on that triangular shape, the delay lock loop logic and architecture is established.


(Refer Slide Time: 13:32)

DSSS Tracking

Figure 1 (next slide):

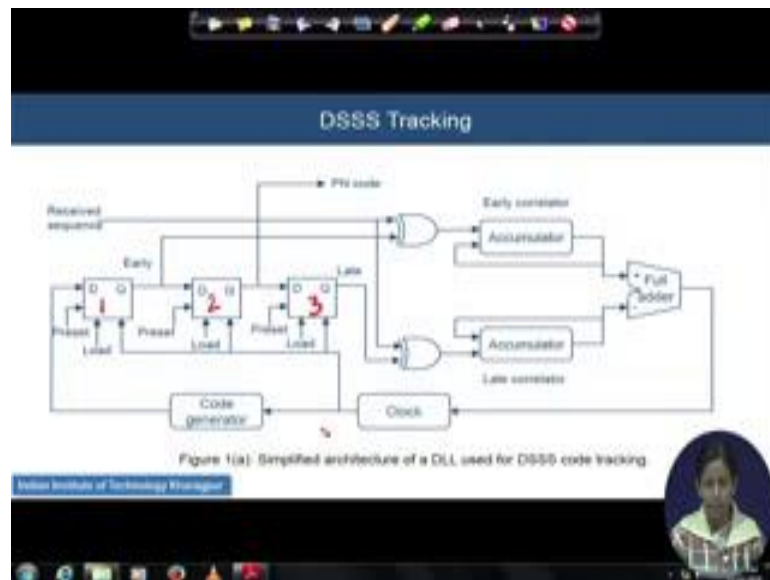
- It shows an example of DLL architecture.
- The locally generated code sequence is sent to a three-stage shift register.
- The last stage, which represents a delay of three chips in the sequence, is the late code.
- The code sequence from the middle stage is used to decorrelate the incoming signal, and the sequence from the first stage is the early code.
- The incoming code sequence is modulo-2 added to the early local PN sequence in the upper path of Figure 1(a).
- If they are synchronized then the correlation function shown at the top of Figure 1(b) is generated by the early correlator.
- The correlation function shown in the middle of Figure 1(b) is implemented through this

Indian Institute of Technology Madras



Let us see how. Let us first I wish to go to the next slide first to show you the DLL architecture.

(Refer Slide Time: 13:39)



The DLL architecture looks like this figure 1-a, where you will see that the architecture will be basically a combined combination of three shift register stages, stage 1, stage 2 and 3. Remember the output of the stage two is utilized for the correlation of the incoming signal with this locally generated PN codes. So, second stage is always considered to be mostly corrected in sync locally generated sequence that is directly

multiplied with the incoming signal for the despreading process. And the shift register 1 is the early called the early stage and the shift registers stage 3 is called the late stage. So, this is called the early gate, this is the late gate, and the central part is the actual one I mean mostly corrected one I should say having minimal error and minimal phase alignment this alignment with the incoming signal, and hence it is always getting used for the despreading purpose.

The other components that are present in the circuit are something like this. There are two XORs; actually one XOR is XORing the incoming signal with the early gate output, and another XOR gate the lower one he is XORing the incoming signal or the received signal with the late gate output. The output of both the XOR gates the upper one and the lower path, there are outputs are getting accumulated here, we call it actually basically set the accumulator they are the correlators, this is a correlation operation going on inside it. And this early correlator because he is utilizing the early code and the accumulator output of lower one is called the late correlator because he is a correlating the late gate output.

And after the correlated output early gate as well as late gate correlated output is the fed to the full adder circuit, where it required the values of both of them are getting basically compare as well as added and then they are basically added inside the adder. And whatever the difference the adder is creating and whatever actually the equivalent value of this early gate output and the late gate output is getting created, based on that actually the clock output of the adder gate output is fed to the clock circuit. And the clock whether the clock needs to be delayed or needs to be a start early based on the output of this adder the decision is taken and hence the clock is driving the code generator either delaying the code generation process or actually spreading up the code generation process or shifting it back actually. Either delay it or start it early like that the code generation is getting control.

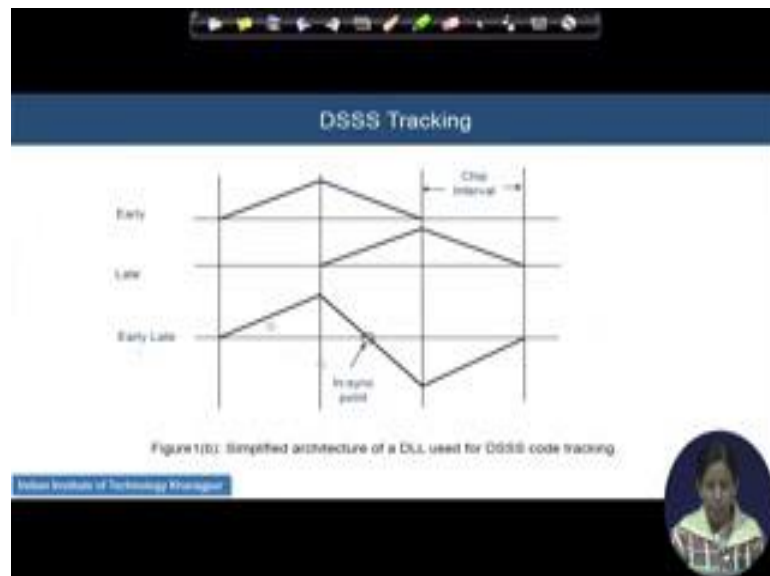
And output of this code generation is way to the first D flip flop and as we understand the shift register architecture the output of the first D shift register is fed to the second one; Second ones output is should be the fed to the third one like that the shift register architecture is generating the local PN code. And the clock here is not only actually feeding the code generation; the clock is also to same with the shift registers stages. So, that everything in that the whole circuit in the receiver is in the sync, so that is all about

the understanding the architecture of the whole circuit, and let us look now how this circuit is performing at tracking jam.

As I explained there are three stages of the shift registers in the circuit. The last stage, which represents the delay of the three chips in the sequence, it is the late code. So, it is having a three actually respect to the first one, it has the delay of three. Middle one is used for to de correlate to the incoming signal. And the first sequence, the first sequence path the first stage output is called the early code. So, we have three codes, one is the late code; second is the early code and middle one is actually the correct code I should say. The incoming code sequence although they are modulo-2 added to the early local PN sequence in the upper path as we have explained in the diagram.

And if there are some there are synchronised if both of them are synchronised then the correlation function at the top one will be showing how, so what I am saying is now let us see a situation when this upper code or upper XOR if I see the output of this upper XOR, if he is having an input from the early code and he is having a received signal. We think a situation when this early code and the received signals both are in sync. So, they are perfectly matching in phase.

(Refer Slide Time: 19:25)



Then what will happen that the output of this accumulator early correlator output I will be able to see like this. I mean this early gate will be slowly increasing over the shift interval and then the next shift interval it will come down slowly. Similarly if I look the

lower one, this is having an input from the late gate and this is the received sequence also. So, the output of this second correlator here, we will get a late gate output like this. Remember, if you consider that there is a gap of the chip interval is equal to 1, so then the late gate will start one chip interval delayed than the early gate. So, if I compare this with respect to this, you will get actually one chip interval delay if I considered that this guy is having zero, it is a correct synchronization point kind of correct code state.

So, late gate will be giving me the structure like this. So, what is happening at the output of the full adder? So, accumulator output one, he is giving a diagram like this. Late gate output is giving me the output of like this. So, the full adder the output of the full adder, if I try to go ahead and try to look into it, see actually where the voltage is coming down and here the voltage is going up. So, the voltage will come down and then finally it will go down even because it is for the late gate again driving towards the down.

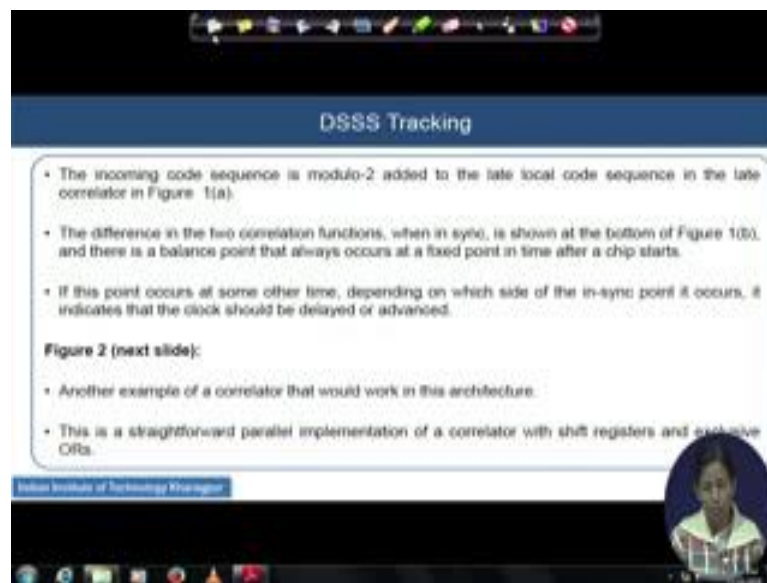
So, you will get a point when there is a crossover happening, we call it is in-sync point. Remember the point this - in-sync point, this is a basically a balancing point between the early gate as well as the late gate. This balancing point we will indicate exactly will happened at the middle of this chip duration. If you are actually when they when there is a perfect it should be happening at the centre part of it, if the code sequences are perfectly aligned and they are having the same length and all.

But what will happen that if something in some extra (Refer Time: 21:57) also not getting added between them. And if you are not getting the in-sync point here, if you are actually the output voltage of this adder full adder circuit is somewhere such that you are either get you are here or you are here. So, in that situation the adder output will never give you zero. The adder output will give you either some positive value or some negative value.

If you are getting some positive or negative value that means, the phases needs to be corrected from by the clock circuit based on where you are, you are in the early stage or you are in the left side of the sync point or you are in the right side of the sync point the phase will be either delayed or it will start early. And hence that instruction will going through the code generator by controlling the phase of the code generator and the timing adjusting the timing point of the code generation. And the correction will be done and applied accordingly in generation of the local PN sequence, so that is a concept that early

gate and late gate get with a perfect synchronization with the perfect synchronization of the received sequence they give actually a very symmetric architecture. And if this synchronization is not there that is why actually you will be either in the left side or right side of the balance point, and based on that you are trying to adjust it such that you are coming back to the balance point once again. So, that is about the direct-sequence tracking mechanism by the architecture of the early late gate.

(Refer Slide Time: 23:48)



DSSS Tracking

- The incoming code sequence is modulo-2 added to the late local code sequence in the late correlator in Figure 1(a).
- The difference in the two correlation functions, when in sync, is shown at the bottom of Figure 1(b), and there is a balance point that always occurs at a fixed point in time after a chip starts.
- If this point occurs at some other time, depending on which side of the in-sync point it occurs, it indicates that the clock should be delayed or advanced.

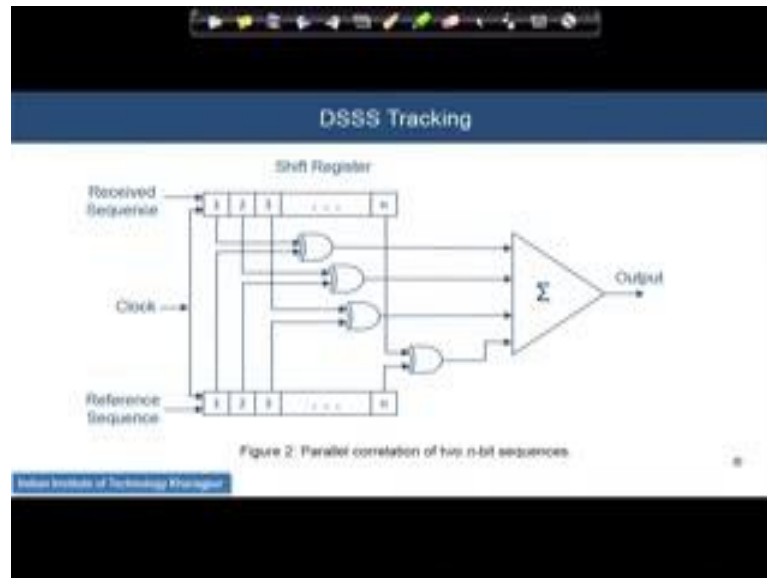
Figure 2 (next slide):

- Another example of a correlator that would work in this architecture.
- This is a straightforward parallel implementation of a correlator with shift registers and exclusive ORs.

And this is the explanation whatever we have already discussed in the last slide. This difference in the two correlation functions when they are in-sync that will come to the bottom. And if they are in-sync for the upper cases it will come in the upper one. And there is a balance point which always occurs at a fixed point in time after a chip starts. And if this point occurs at some time rather some other time then the central point of a chip duration then actually your synchronization is disturbed you have to re establish it, either by delaying or by advancing the clock, so that is the final idea.

Remember that this architecture and this philosophy also can be generated some different kind of the receiver architecture and that we will discussing the next slide that is an another example of that correlator that would work in this architecture. And this is a straightforward parallel implementation of the correlator with the shift register sequence and the exclusive or gates, how?

(Refer Slide Time: 25:01)



Look at this architecture. So, this is the combination of the all XOR gates and the shift registers in the circuit. And the received sequence are getting fed and stored in the upper shift register. Reference sequence the locally generated PN sequence norm in if they are generated and they are stored here. So, n-bit or n-chip locally generated PN sequence is stored in the lower register and incoming signal of n-chip incoming signal is stored in the upper shift register.

Remember clock is synchronised between the reference sequence and the received sequence, such that whenever they bit-by-bit XORing will be happening and for each and every bit we have a dedicated XOR gate architecture. And from 1 to small n number of the chips, we have n number of the parallel paths. And they are getting bit-by-bit XOR operation is going on output actually also you are feeding to the adder circuits. So, it is equivalent to the circuit that we have earlier discussed. And this can be equivalent they both of them are. And if you try to see the output, you will be able to see that whenever the things are not in-sync you will be getting adder output will be will not been the synchronised at the sync point, and you will get some value based on that, if the value is positive or negative based on that clock pulse can be adjusted.

And you can be either it clock pulse can be either delayed or it can be advanced, so that you can control the generation of the locally generated PN sequence phase. Basically you are controlling the waveform generation of your local generator code generator, such a

way that you get aligned with a code sequence within the remaining error that is left by the code acquisition mechanism by tracking.