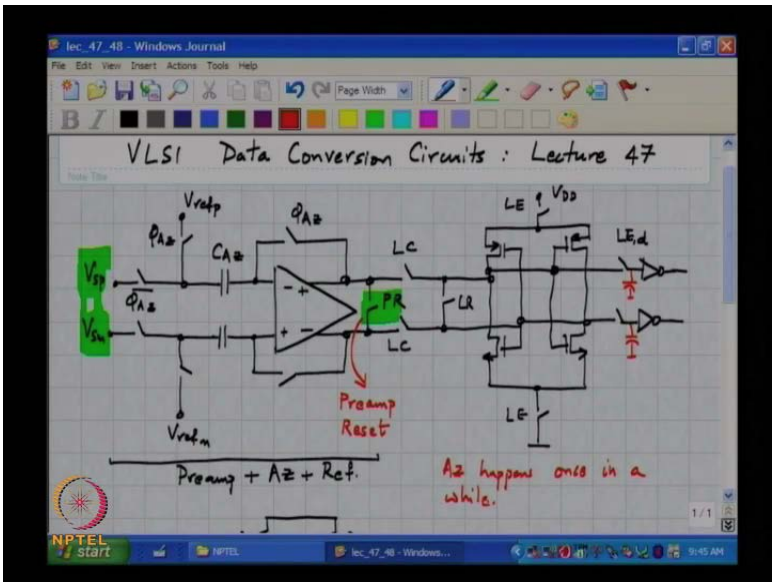
VLSI Data Conversion Circuits
Prof. Shanthi Pavan
Department of Electrical Engineering
Indian Institute of Technology, Madras

Lecture - 47
Flash ADC Case Study

This is VLSI data conversion circuits, lecture 47.

(Refer Slide Time: 00:15)



In the last class, we were looking at a single slice of flash A to D convertor, and wondering what kind of timing schemes we need to use, in order to make sure that every individual block has the maximum possible time available for to it, do it is job. So, the various sub blocks V ref p, V ref m and we call this V sample p and V sample m and this we connected to the latch.

Student: ((Refer Time: 03:35))

So, this shows the circuit diagram of a slice of a representative flash A to D convertor, this is the preamp as we saw, plus auto 0 plus reference storage, and the output of the preamp goes to a regenerative latch. The output of the latch is stored on the parasitic capacitance at these investors, and the clock which goes and enables these invertors or enables the storage, in other words the sampling instant of the output of the latch must begin well after the latches regenerate.

So, if you tried to connect the subsequent in invertors to the latch during the time, the output is building up that might lead to errors when the inputs are, lies just begin it to regenerate you suddenly go and dump some capacitances on those nodes, it is very likely that the decision will flip. So, you want to make sure that, the output of the latch has a regenerated to a sufficiently large degree, before you try anything funny with those nodes.

So, that clock is denoted by L D, and you kind of hold it through the entire clock cycle. So, the sampling instant of the whole comparator slice is what, which instant of time corresponds to the sampling instant of your slice as the diagram here shows.

Student: ((Refer Time: 06:02))

[FL]

Student: ((Refer Time: 06:01))

End of 5 a z bar no that is not right, can you comment on the nature of these wave forms, these are held wave forms from the sample and hold, we saw why it make sense to have a sample and hold in the first place. And the reason to have a sample and hold is that, when you have an array of comparators, even though you propagate a clock across the array, what we will see is that, due to physical limitations, just because these slices cannot all be at the same place.

It follows that the time it takes for a clock edge to reach one slice, is different from the time it takes to reach another slice, so if a sample and hold was not used. In other words if V s p and V s m were directly connected to the continuous time input that we were trying to digitize. Then, you find that the effective sampling instant of different latches is different, which means that all these comparators are not looking at the same input, which will lead to a whole bunch of errors, especially when the input is at high frequency.

In this particular implementation there is the additional problem that, the preamp is now what is called an integrating preamp, where we have specifically exploited the fact that, the input to the preamp is going to be a held quantity. So, if you had an array of latches that is no good from a clock skew point of view, apart from there being too much offset.

So, we add a preamp the ideally of course, the preamp is supposed to be a fixed gain of A with an infinite band width, it is job is to simply reduce the input referred offset of the latch by a factor A you add auto 0 and also store references.

So, now the preamp latch combination, if the preamp gain is I mean is very large and have got infinite band width, is simply thought off as comparator whose input referred offset is very small. And the gain you need is should be large enough to make the offset of the latch divided by A to be much smaller than a fraction of an LSB usually one designs these things, so that the 3 sigma offset in this comparator. In other words, the offset random offset in this comparator slice when referred to the input of this slice is effectively the deviation of this particular comparators threshold, from the ideal desired threshold. What is the ideal desired threshold here?

Student: ((Refer Time: 09:12))

What is this slice comparing the input against

Student: V ref p minus

V ref p minus V ref m and any offset input referred in this slice will cause the threshold to deviate from that V ref p minus V ref m by some amount. Now, that some amount that we are talking about is an something that we have to make a choice about obviously, it is not possible to make the offset 0, there will be some residual offset. Because, the gain of the amplifier is finite etcetera etcetera, so that something is often calculated to be quarter LSB in the 3 sigma sense.

So, all devices here will have a random mismatch in spite of auto zeroing, there will be some small residual offset left in the preamp, more importantly because of the finite gain of the preamp, the offset of the latch when divided by the gain will be some quantity. What we need to do is make sure that, that some quantity is in the 3 sigma sense smaller than quarter LSB. Now, one might say why quarter LSB why not half an LSB, the only you know justification we have for these kinds of choices is that, making it half LSB means that your in the worst case potentially allowing a DNL of minus 1 LSB.

So, the two codes will be a through thresholds will be, assume you have two successive comparator slices, one slice as threshold has come down half LSB, the other slices

threshold has gone up by half LSB. So, the code width of that one particular code will become infinitely small, which is and if the code width changes drastically from the nominal width, you can I mean while it is difficult to quantify, what the effect will be on distortion etcetera.

One can safely say that it will is most likely to be much worse than, when the thresholds are all equally spaced, now to avoid this clock skew problem what we have done, is added a sample and hold upfront, which holds the input to the comparator array to a held value. This way now your dealing with within quotes a D C signal as far as this clock period is concerned, so skew in the clocks to the comparator array does not really affect us.

So, the sampling instant of the entire A to D convertor is, whatever edge is responsible for changing the sample and hold from track to hold, that is the sampling instant of the comparator array. What is the sampling instant of the latch?

Student: ((Refer Time: 12:28))

At what instant is the latch sampling it is input, or what is the voltage that is being regenerated?

Student: ((Refer Time: 12:43)

The falling edge of L C is the instant at which the latch is sampling the input, and when is the output of the latch available.

Student: Falling edge

At the falling edge of L E sub d, so the latency of the flash is what, how do you think you would want to define the latency of the flash.

Student: ((Refer Time: 13:25))

In an ideal flash if you sample the input at some time, the digital code must be available

Student: Instant

At that very instant clearly that is not possible, so how would you define latency

Student: ((Refer Time: 13:38))

You sample the input of the flash at some instant, the digital code comes out at some other instant and the difference between those two times is the...

Student: Latency

Delay or latency of the flash at D C is that clear, now in this particular example the output is available a little bit after L E sub d, please note that this is not all, this has to go to a digital backend where you do

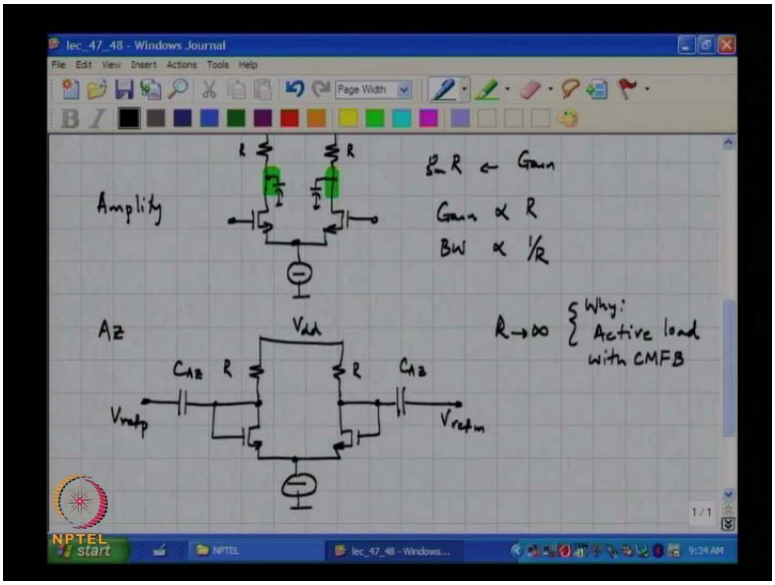
Student: ((Refer Time: 14:13)

You do bubble correction and thermometer to

Student: Binary conversion

Binary conversion, that digital backend will also have a combinational delay, that is how combinational logic it will have some delay. So, the latency of the flash convertor is the time instant between the track to hold transition of the sample and hold, that is where your clock starts ticking. And when the digital code at the end of the decoder is available is when the digital data is out, so that represents the latency of your flash convertor.

(Refer Slide Time: 15:22)



Now, one thing that we have done exploiting the fact that the input to the comparator array is D C, is that we have used in not a broad band amplifier whose gain is very large that we know is very difficult to do.

Because, if you want a high gain recall that, this is in principle the schematic of the preamp, we can put active loads etcetera and so on, but that the gain is proportional to g m times R. But, the band width becomes inversely proportional to R, so gain is proportional to R and the band width is proportional to 1 by R. And then, we said this rather than worrying about the time constant of the band width of this preamplifier, what one is actually interested in is the differential voltage developed across the two inputs of the latch.

So, if the input developed at the two inputs of the latch is large enough, that is all we care about and you do not really need the output of the preamp to settle, so then we said we can actually use active loads to the preamp and get a sufficiently large input voltage. So, that leads us to the integrating preamp, now the output of the preamp at the end of the latch connect phase will still be at, if this R is very very large, these parasitic capacitances at the end of the latch phase will be charged to some dynamic gain, which is g m times delta v times, time available during the latch connect phase divided by C p.

So, dynamic gain times, the input voltage will be differential voltage stored on these parasitic capacitors. Now, when we want to go to the next clock cycle we do not want any

Student: Memory

We do not want any memory from the previous input at all, which will unnecessary lead to errors in thresholds depending on the previous input which is called hysteresis. So, as it stands in this diagram, what we would have to do, would be to auto zero the preamp, when we auto zero the preamp what happens, during auto zero this is during amplify. During auto zero for the time being please assume that this resistors R are very large, and are realized using current sources with common mode feedback, but to understand the operation in the auto zero phase.

So, this is the equivalent circuit diagram in the auto zero case, so what is the time constraint in the auto zero phase...

Student: g m

Which g m by which c

Student: ((Refer Time: 19.50))

Why current sources, active loads with common mode feedback, so what is the band width in this phase

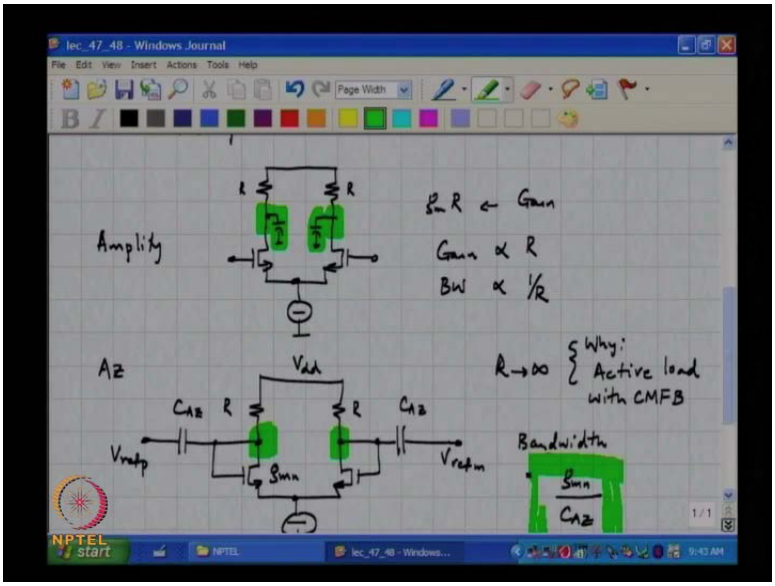
Student: G m by c a

G m n, let us call this g m n, is g m n

Student: By c a z

by C a z that is correct

(Refer Slide Time: 20:23)



So, this means that the settling time of this loop this is nothing but, a negative feedback loop, the settling time of this loop depends on g m n by C a z, C a z we saw a couple of classes back, that must be chosen to be sufficiently large, in order to prevent effects due to attenuation by the capacitor of divider. But, and therefore, cannot be chosen to be too small, so if the output voltage which at the end of the auto zero phase will settle to what value.

Student: V offset

What will

Student: V offset

This voltage settle to

Student :((Refer Time: 21:41))

It will settle to

Student: The V offset

Differential voltage closed to V offset, so these two parasitic capacitances which at the end of L C were charged to some large number times the differential input voltage, should at the end of the auto zero phase, come back to some voltage which is closed to zero, because hopefully the offset is small. And that decay is governed by this time constant g m n by C A Z, does it make sense; so this means, therefore that there is a problem with respect to if g m n by C P. So, where g m n by C P governs the dynamic gain, and if that does not settle in the time that one has, it is very unlikely that g m n by C A Z will be sufficiently small.

So, as to get the output to settle, it is important to this phase for the entire circuit to settle, why during the amplify phase it is not necessary that the output settle. But, in the auto zero phase it is important that the output settle, why?

Student: ((Refer Time: 23:29))

During the auto 0 phase we are interested in capturing the

Student: Offset

Offset, so it better the output better settle whereas, in the other phase you are only interested in a gain times, the differential input voltage. So, this timing constraint makes it very difficult to make the whole thing work in a sufficiently small amount of time, so trying to do this will be a difficult thing. Simply because, there is not enough time for this to be a sufficiently small fraction of the auto zero period, and what is the way we have it what is the auto zero period, when can we do auto zero for the preamp.

Student: When the latch is individually placed

When the latch is not being used, that is when there are as far as the preamp is concerned, it is job is over the moment the latch starts processing the previous output, and during that time what is the sample and hold doing, or the track and hold.

Student: Tracking

The track and hold is

Student: Tracking

Is tracking, so during that time the preamp is free to do whatever it likes, and we discussed the tradeoffs between the track and the hold time, assuming both of these are half the clock period. You basically have half a clock period, for auto zero in the best case, because you will always have some you know non-overlap times needed to be enforced and so on. So, the actual period for the auto zero will be, the upper limit is about half a clock cycle, during half a clock cycle that this time constant g m n or C A Z by g m n must be sufficiently small.

Such that, the output of this preamplifier settles to sufficient degrees of accuracy within this half clock period, that become a very difficult thing to do. And is been a used to be considered a big bottle neck in designing high speed flash converters, then somebody realized that well the auto zero, the offset is static. In other words, the offset in the preamplifier does not change from cycle to cycle, and the reference also does not change from cycle to cycle, so there is really no point in trying to...

Student: ((Refer Time: 26:17))

Do estimate the offset and cancel it every clock cycle, which is eating up I mean which is basically making it impossible to run the flash at high speed, because of this constraint. Since, the auto zero it does not have to be done at every cycle, if rather if the auto zero does not have to be done every cycle, then this constraint now becomes I mean you do not have to satisfy this constraint. So, in several systems using flash converters, it turns out that there are dedicated intervals of time during which, the flash A to D converter is not being used at all.

As I mention in the last time a case in point is the disk drive read channel system, where there are intervals where the flash converter is not being used. So, during that time all the preamps are auto zeroed, which means that there are I mean the references are stored on the auto zero capacitors, as well as the offset and then, you just run the preamp in the amplify phase all the time. In other words, this auto zero happens only once in a very long while, so if this happens in once in a very long while, what new problem do we have now, what was the auto zero phase doing to the preamp apart from storing the offset.

Student: Reference in offset

[FL]

Reference in offset that is true, but apart from that what was it doing

Student: Previous value of that

So, during the auto zero phase side benefit was that the previous voltages at the output of the preamp, by getting we are effectively getting discharged through this feedback network. These two parasitic capacitances, which at the end of the amplify phase were being charged to gain times the input difference, at the end of the auto zero phase are getting discharged to voltages, which are independent of the previous value. Which means, therefore that there is a now if this auto 0 phase is only done once in a very long while, we now have the additional job of...

Student: Discharging

Discharging the

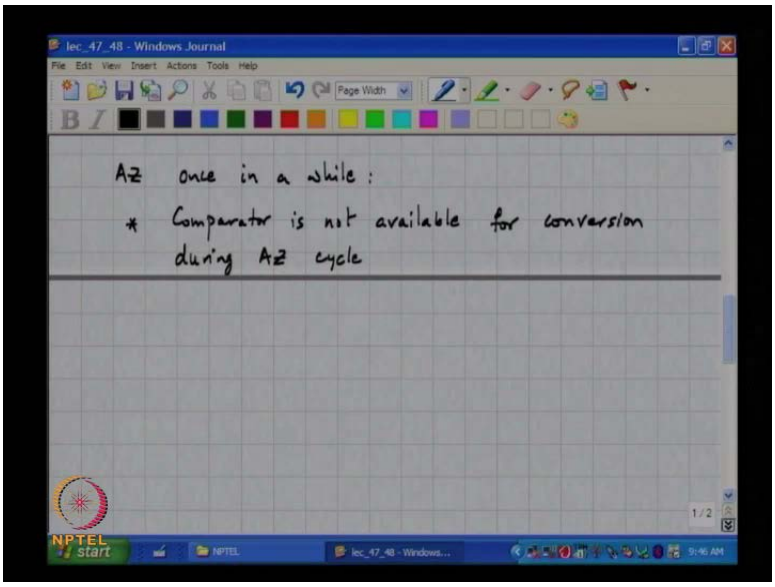
Student: ((Refer Time: 29:28))

The preamp outputs every clock cycle, so as soon as the latch connect opens, we need to discharge the, the parasitic capacitances of the at the output of the preamp. So, what can you suggest something, what do you think we can do, how do we eliminate memory from the previous cycle as far as the preamp output is concerned.

Student: ((Refer Time: 30:09))

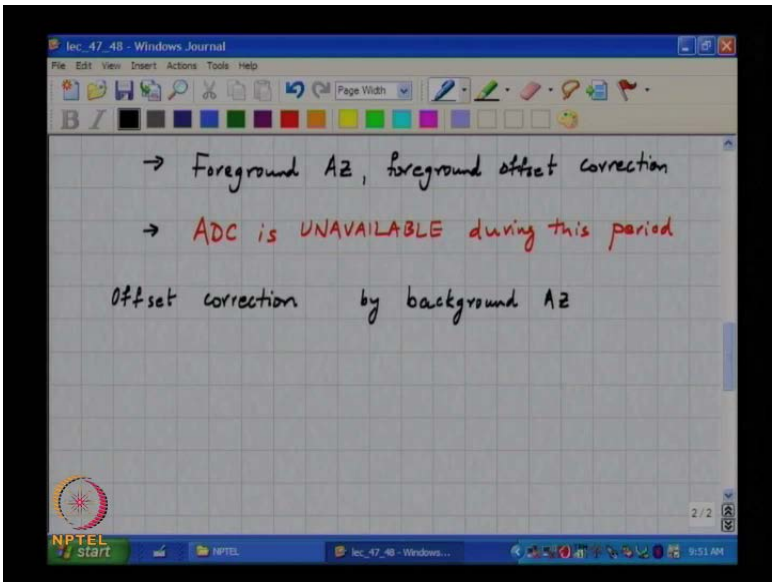
As we did with the latch, we have an additional phase P R and P R stands for Preamp Reset, does this make sense, and auto zero happens once in a while.

(Refer Slide Time: 31:07)



So, let me now introduce you to some standard terminology that we have to keep, and for encountering when you read these papers, so auto zero can be done once in a while. And this means that clearly the comparator is not available for conversion during an auto zero cycle, is it not. So, if one has to auto zero all the comparators at once, it means that during that time the A to D convertor itself is, is not available.

(Refer Slide Time: 32:12)



So, this kind of auto zero cancellation is called foreground auto zero, or foreground offset correction, and the key point is to note that the ADC is unavailable during this period. This is perfectly acceptable in some systems, and is not at all acceptable in other systems, so this is a call that you have to make, once you understand the kind of system into which the ADC goes.

One might argue that, why do not I just do offset correction and storage of references, just during the time of power up, after all every I C has to be powered up, during that time until the whatever biased voltages etcetera settle, system is not working, is it not. So, during that time one can also conceivably estimate offset and store it, and the argument, then is why not I just do it during power up and leave it like that, do you have any comments on that.

I other words, why do not I just estimate offset once, and store the references once in other words, put this in auto zero mode just once and be done with it.

Student: ((Refer Time: 34:24))

[FL]

Student: ((Refer Time: 34:26))

So, whatever your storing is charge on this capacitor and charge on the capacitor will leak due to junction leakage of all the various switches, that are connected to the capacitor which means that, it is only a matter of time before all those all that charge leaks off from CAZ, thereby corrupting the voltage that you have stored. So, in other words it is not acceptable to do offset and reference storage just once and say, let me carry on for a long while, so this is easy operation has to be repeated every, so often depending on the leakage through the capacitors and such.

The other way of correction is in the background and what do you think the idea here would be, and where do you think this would be useful.

Student: Wherever you do not have the

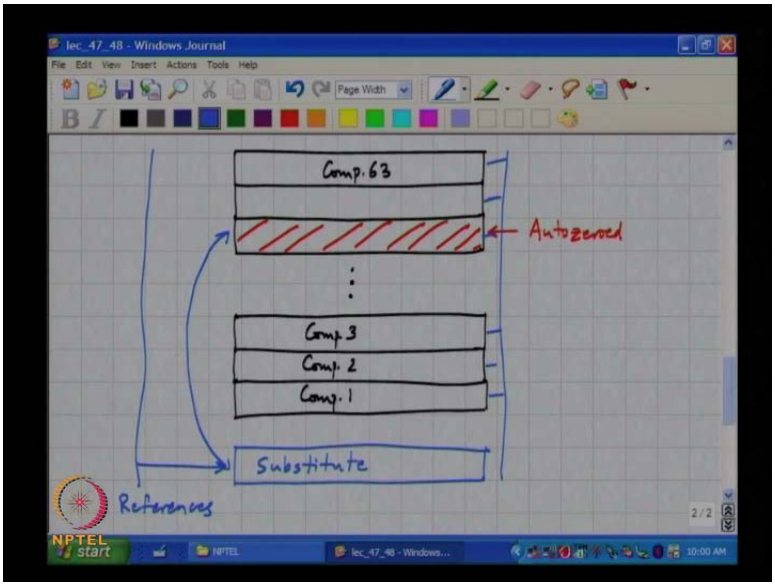
So, whenever you cannot effort to remove the converter from the system, I mean foreground offset correction, is basically something where you do not have the converter

available for conversion, during the auto zero period. During background auto zero, I mean there are several systems where this is not possible, but you do need to correct for offsets and this offset correction, therefore has to happen, either every cycle, which makes the converter very difficult to design.

Or can be done in the background one conceivable way of doing it is to pick one slice, effectively pull it out of the array auto zero it, and when this slice is being auto zeroed obviously, it is not available for conversion. One could in principle replace this by another slice, which is being already auto zeroed, so you send in a substitute you pullout one guy send in a substitute. And you have the entire array and you keep doing this for all the slices in the array of course, you can see that this kind of a messy affair, because that substitute as to now effectively replace.

First is to replace comparator 1, comparator 2, 3, 4 and so on, all the way up to 64, and it is not as if all this 64 comparators are sitting in the same place. So, you can expect there it to be a quite a messy situation from view point of layout, you understand.

(Refer Slide Time: 38:40)



So, the basic idea is like this, so this is comparator 1, comparator 2, comparator 3 and so on, let us say 63 this is a 6 bit flash. So, let us say we choose some comparator here to be auto zeroed and then, we have here a substitute whose is already been auto zeroed and replaces this comparator, when that comparator is being auto zeroed. And therefore, what is the meaning of replacement, you cannot physically remove this slice and put in

another slice, it must be that the signals which come to the substitute must be routed back to the output of that comparator, which is being auto zeroed.

In other words, effectively this substitute must be connected to all the comparators in some fashion, and all the references must also come to the substitute, because the substitute has to assume the role of every single comparator in the array. So, this is one way of doing background offset correction, and now one does not have to worry about the amount of time it takes for the preamp to settle during the auto zero phase, because a voices is any better a solution.

Student: ((Refer Time: 41.42))

No that is ok, but why does it make life easy during the auto zero phase, that is the whole problem we are trying to solve is it not, we cannot afford to do it every cycle. So, in the background offset corrections scenario, the entire array was being auto zeroed at once in any case the A D C is not needed, so I will make the auto zero period a very very long time. Therefore, there is no problem with accurate settling of the auto zero capacitors, now with background correction, I have a substitute in place of a comparator I am auto zeroing.

So, just before I pull a particular comparator out of the array electrically, the substitute must be charged to the same reference, as the comparator which is going to be pulled out, this is all common sense, there is nothing profound about this. So, but you know how is it solving my problem as far as settling of the preamp output is concerned.

Student: ((Refer Time: 43:04))

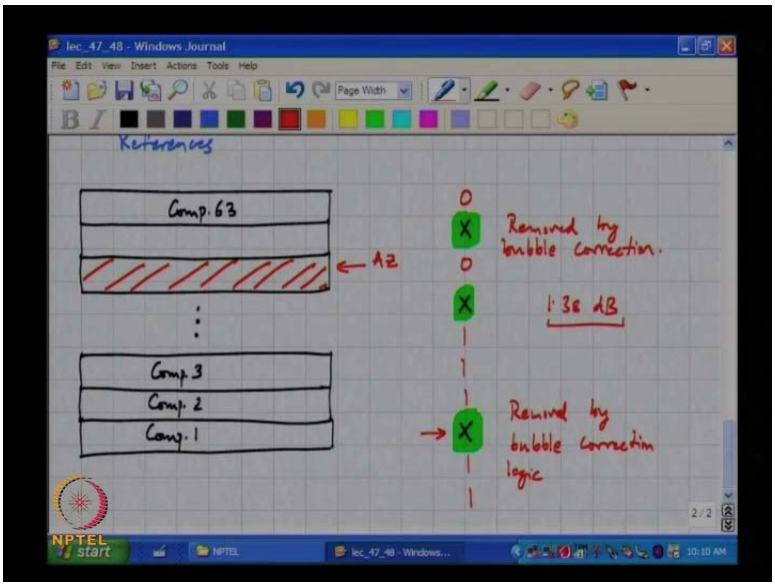
So, as long as the substitute has been properly auto zeroed and the right reference is been stored, the substitute can remain in the field for a sufficiently long time. So, that the main comparator can also be given as large time as necessary for the auto zero capacitors to get stored, to have their voltages stored correctly. And then, you can replace it back and but, you need to be aware that, you need to do this now for all the, whatever in this particular case 63 comparators, which means that by the time you come back to the first comparator.

The time must be sufficiently short, so that what you had stored earlier on the first comparator has not leaked off, turns out that that is usually not a problem, because even if you say I am going to allow like 10 clock cycles for auto zero which is a really really long period. You basically are looking for at 20 cycles ((Refer Time: 44:30)) 10 cycles which charge the substitute, and 10 to kind of you know this thing. So, you can see that as you go through the entire array, you are only a few I mean about a 1000 cycles of time is what it takes, for you to go through the entire array.

Flash converter is typically operated very very high speeds, which means that a 1000 cycles is basically nothing. In other words when compared to the time needed to leak all the charge of the auto zero capacitors a few 1000 cycles is really nothing. The trade off unfortunately is that, the practical implementation of this kind of substitute replacement becomes a very messy affair. Because, without getting into the details one thing that we are, one can see immediately is that the substitute must electrically be connected to all the outputs at some stage.

Either right at the output of the comparator array or somewhere in the logic or whatever, slightly different approach which avoids the substitute altogether, is exploits the following idea.

(Refer Slide Time: 46:07)



So, let us say we are interested in auto zeroing this slice, now if you are interested in auto zeroing this particular slice, there are only few possibilities, please recall that the output of the comparator array ideally should be a string of...

Student: ((Refer Time: 46.39))

All once followed by a string of 0's, so if the output of the comparator array would ideally be this, and we are auto zeroing say this comparator, what does that mean?

Student: The signal is far away from the threshold voltage I mean particular

Correct

Student: And you can actually afford to have offset

No, see if this particular comparator is being auto zeroed, then what do you think will happen, what can you say about the digital output of that slice, if a particular comparator is being auto zeroed what does it mean as far as...

Student: 0

Is that slice available or not

Student: It is not available

It is not available, so the output is

Student: 0

Not necessarily 0, it will be is something that you do not know, because the latch will continue to latch it will take some random input and latch. So, this will be some x, we know that we are auto zeroing this comparator and the output is x, but looking at the code do you think this is a problem.

Student: No

This is not a problem, why

Student: ((Refer Time: 48:15))

We know clearly that I am auto zeroing this, so I do not need to trust this decision, and the 1 0 transition is happening here, which means that I can disregard this x, you understand. And still even though I am auto zeroing a particular slice, I do not incur any error, because that particular slice is not available, I mean all this is saying is that, in a flash A D C the only comparators that really matter are those which are close to the...

Student: The 1 0

The 1 0 transition in the thermometer code, I mean fundamentally there's a lot of redundancy in a flash, which is what is giving you the speed, but you do not need to know I mean we know that, if a number is greater than 2, it is definitely greater than 1. There is no point in actually trying to, try and figure out if it is greater than 1, if you already know it is greater than 2, but that is what is being d 1 in flash. So, the fact that there is a lot of redundancy allows us to exploit this redundancy, and saying that any comparator which is far away from the 1 0 transition can be safely auto 0zeroed, without any penalty at all.

And this is completely transparent to the user

Student: But, that 1 0 transition will be single dependent

Yes, correct I will come to that, so clearly as he points out, how can we ensure that the 1 0 transition in the thermometer code, does not occur...

Student: ((Refer Time: 50:08))

Precisely at the comparator that we are auto zeroing, we cannot ensure that if we cannot ensure that, then we should be prepared to accept some error. So, anyway before we get to that point, let us see what happens to this x here, how do you propose to get rid of x in a completely transparent manner. What is happening to the outputs of the comparator arrays are they going directly to the...

Student: Poles

That the transition detector or there is there something else in the middle

Student: ((Refer Time: 50:51))

This is a bubble correction logic which we use anyway, so if you now have an x somewhere like this, and you have bubble correction, what will the bubble correction logic do.

Student: ((Refer Time 51:05))

What is the bubble correction logic

Student: ((Refer Time: 51:09))

It is majority of 3 for instance, so if this x occurs far away from a transition from the 1 0 transition, automatically it will get corrected by...

Student: The bubble correction logic

The bubble correction logic, so there is the really nothing we need to do, we can safely go ahead and auto zero, you understand. So, this is removed by the bubble correction logic, similarly if it is far away from the transition here, these kinds of things are also removed by the bubble correction. So, if the auto zero occurs here ((Refer Time: 52:15)), then if it happens to be the output is random, if it happens to be 1 there is no problem, correct if it happens to be a 0 what error have you made...

52:33

We have made an LSB of error, and you can do some probabilistic calculations where, you can find the probability that the comparator slice you are auto zeroing, happens to be exactly the same as the binary I mean the 1 to 0 transition point in the comparator. So, with that probability the quantization error will increase by 1 LSB, and you can do the maths yourself depending on the various positions of the auto zero comparator with respect to the transition.

You can come up with a bunch of probabilities, and the quantization error will change, depending on whether it is just above the transition, just below the transition and so on. So, once you have these probabilities it is very straight forward to calculate the quantization noise as a function of this, these probabilities which will mean, therefore that the ideal quantization noise which is delta square by 12, will now do you thing increase or decrease?

Student: Increase

The mean square error will increase, because the error is always being increased with respect to it is ideal value. So, after all this math it turns out that for a 6 bit converter, if you employ bubble correction logic, and do this background cancel auto zero cancellation, as you keep going along. You would lose about 1.38 d B from a 6 bit converter I really need 38 d B or so, you lose 1.38 d B, it seems like a fair trade off given that, there is no additional complexity as far as hardware implementation is concerned.

And this assumes that you are doing this auto zero continuously of course, once you go through the array of 64 comparators, you can actually wait you do not need to start the next auto zero cycle again, because you can hold this offsets for a while. Which means that, this 1.38 d B is a very is a worst case value, if you duty cycle all this by a factor of 10 for instance. So, you auto zero this the entire array, then you wait for 10 times the amount of time you took to auto zero, the entire array during that time the A to D is behaving like a perfect A to D converter without, any auto zeroing happening at all.

So, the effective increase in error is now going to get divided by a factor of 10, the effective increase in the quantization error with respect to delta square by 12 is has gone down by a factor of 10, which means that the effect will loss in S N R you can go and calculate to be like 0.2 or 0.3 d B or something like that. So, that is something that we have used in one of our designs, so these are various ways of doing the flash A to D conversion, we will continue in the next class.