

LDPC and Polar codes in 5G Standard

Professor Andrew Thangaraj

Department of Electrical Engineering
Indian Institute of Technology Madras

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check (SPC) Code

Hello welcome to this week's lecture on soft in and soft out decoding we are going to proceed with looking at soft in soft out decoding for single parity check codes. Once again I will take a simple example and I will generalise from that later on okay.

(Refer Slide Time: 0:37)

The slide contains handwritten notes and a table. At the top, it says 'Single parity check code (SPC code)' and '(n, n-1) linear binary code'. It shows a message vector $m = [m_1, m_2, \dots, m_{n-1}]$ and a code word $c = [m_1, m_2, \dots, m_{n-1}, p]$. An arrow labeled 'Encoder' points from m to c . A note below says $p = m_1 \oplus m_2 \oplus \dots \oplus m_{n-1}$ XOR of all message bits. Below this, it says '(3,2) SPC code :'. A table shows the mapping between message vectors and code words:

| m | c | # of 1s in c : even |
|-----|-------|---|
| 0 0 | 0 0 0 | SPC code: consists of all even-weight vectors |
| 0 1 | 0 1 1 | of length n |
| 1 0 | 1 0 1 | |
| 1 1 | 1 1 0 | |

So what is this single parity check code? Okay so it is an n, n minus 1 linear block code okay and the way it works is as follows if it is n, n minus 1 remember this is k right k as in n minus 1, so your message vector is m_1, m_2, m_{n-1} in most cases it is encoded in systematic fashion this is how the encoder works okay. Your code word is going to be the same message bit appearing in the 1st n minus 1 position and in the last position you will get a parity bit and this parity bit is simply the XOR of all the previous message bits, all the message bits okay so this is the message, this is the parity bit.

So you can see why this is called the single parity check code you have how many even message bits you want you just put out one parity which is the XOR of all the message bits that are being seen so far okay. So let us look at the 3, 2 single parity check code SPC code. I will abbreviate this as SPC code okay. So here the message could be 00, 01, 10, 11 and what will be the corresponding code word be 00 and the parity here is going to be 0 again the XOR of these 2 is 001 you will have 1, 10 you have 1, 11 you will have 0 okay. So there are quite a

few things to notice here for instance the number of 1's in c is always even okay so the single parity check code consist of all the even weight vector length n , the SPC code consist of all even weight vectors of length n okay so this is quite easy to see, so the single parity check code is described in that fashion. Now what about generate a matrix parity check matrix for single parity check code.

(Refer Slide Time: 3:21)

$(n, n-1)$ linear binary code

 $m = [m_1, m_2, \dots, m_{n-1}] \xrightarrow{\text{Encoder}} c = [m_1, m_2, \dots, m_{n-1}, p]$
 $p = m_1 \oplus m_2 \oplus \dots \oplus m_{n-1}$ XOR of all message bits

$(3,2)$ SPC code :

| | | |
|-----|-------|---|
| m | c | # of 1s in c : even |
| 0 0 | 0 0 0 | SPC code: consists of all even-weight vectors of length n |
| 0 1 | 0 1 1 | |
| 1 0 | 1 0 1 | |
| 1 1 | 1 1 0 | |

$G = \begin{bmatrix} I_{n-1} & \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \end{bmatrix}$ $H = \begin{bmatrix} I_{n-1} & \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \end{bmatrix} \end{bmatrix}$

PROF ANDREW THANGARA
IIT MADRAS

$(3,2)$ SPC code :

| | | |
|-----|-------|---|
| m | c | # of 1s in c : even |
| 0 0 | 0 0 0 | SPC code: consists of all even-weight vectors of length n |
| 0 1 | 0 1 1 | |
| 1 0 | 1 0 1 | |
| 1 1 | 1 1 0 | |

$G = \begin{bmatrix} I_{n-1} & \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \end{bmatrix}$ $H = \begin{bmatrix} I_{n-1} & \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \end{bmatrix} \end{bmatrix}$

all 1's all 1's in a row
 $c_1 \oplus c_2 \oplus c_3 \oplus \dots \oplus c_n = 0$

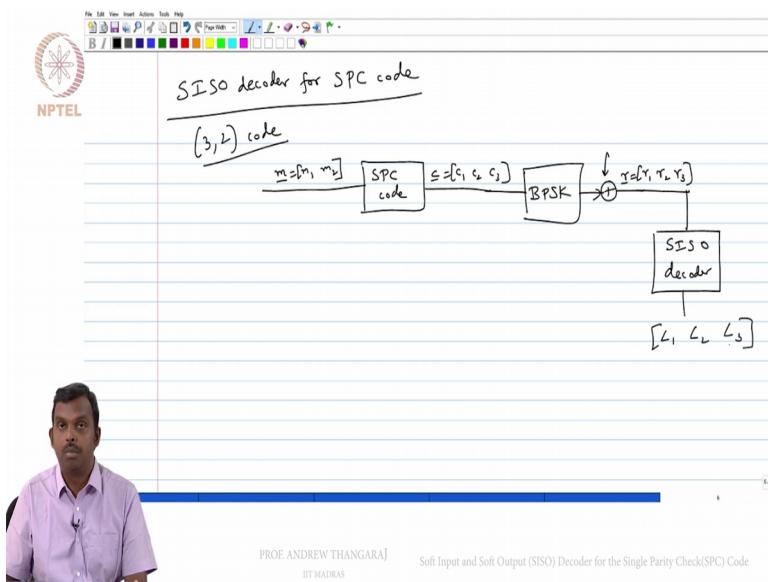
PROF ANDREW THANGARA
IIT MADRAS

The generator matrix is quite easy to write down you have I_{n-1} okay so once you have done with I_{n-1} the parity check part has only a single column and there is just all 1's it is called $n-1$ then you have all 1's coming up ((3:49)) in a distinguished way you have all 1's coming up on the right side okay, so the identity part is a systematic part and then you have sort of a division here and then you have the parity part which is just all 1's.

Now if you convert this into a parity check matrix, the parity check matrix if you remember from the dimensions is just 1 cross n it is basically just one row all 1's okay this is just all 1's in a row okay so this is quite easy to see why this is true. The only parity check that you have is C1 XOR C2 XOR C3 XOR so on till Cn equals 0 okay, so this is the characteristics of single parity check codes so if you take a code word of their single parity check code XOR all the bits together you are going to get 0 okay so this is the single parity check code.

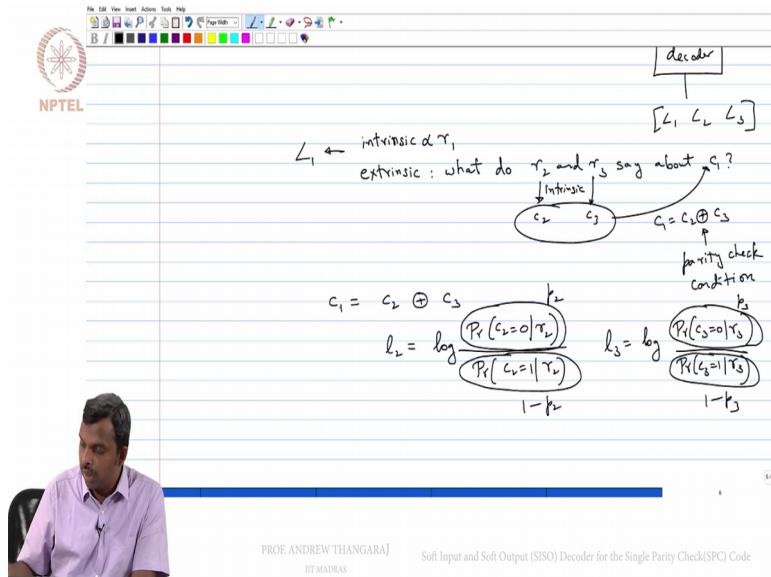
I showed you a small example and larger examples will give you many more code words, so for instance if you look at 10, 9 single parity check code you will have 512 code words, 2 power 9 code words which are all even weight of length 10 okay so that is the description of the single parity check code okay. So it looks quite complex but it turns out even for this code you can do a very simple SISO decoder simple as in you will see eventually the description will become simple but it is something very interesting you can do a very nice and simple implementation for the SISO decoder okay so that will be our focus in this lecture.

(Refer Slide Time: 5:35)



Okay so I will take the 3, 2 code as an example and show it and then we will generalise later on okay, so how is the 3, 2 code working. You have the message vector being $m_1 \ m_2$ okay so this goes into the SPC code encoder, we produce a code word which is C_1, C_2, C_3 this will go into BPSK and then Gaussian noise and then you get your received vector which is r_1, r_2, r_3 now what do they want to do with this, I want to build SISO decoder. Remember what do they want to produce here my capital L_1 , capital L_2 , capital L_3 okay beliefs about the 3 bits C_1, C_2 and C_3 okay so this is something interesting. One can look at what the single parity check code means and all that I will talk about it as we derive this okay.

(Refer Slide Time: 7:07)



So to think about this let us look at L_1 , how do we compute L_1 ? L_1 is going to have an intrinsic part which is just r_1 it is going to be proportional to r_1 okay we know that and what about the extrinsic? So to answer this look at what the question really means? Want to r_2 and r_3 say about C_1 , right? So that is the question about the extrinsic information, right? So this was r_1 was quite easy, what does r_1 say about C_1 ? It is just proportional to that we saw that in the channel LLR calculation. Now for the extrinsic part what do r_2 and r_3 tell about C_1 okay r_2 say something intrinsic about C_2 likewise r_3 also say something intrinsic about C_3 but what do C_2 and C_3 tell about C_1 ? Okay C_1 equals $C_2 \oplus C_3$ is this comes from the parity check condition okay.

So this is the route we have to take r_2 say something intrinsic about C_2 , r_3 says something intrinsic about C_3 and I know C_1 is $C_2 \oplus C_3$ okay so now take the intrinsic information from r_2 , take the intrinsic information from r_3 into C_2 and C_3 and then use this XOR to compute what it says about C_1 and that will be the extrinsic information for C_1 and then I add intrinsic and extrinsic I get my total capital L_1 is that okay? So that is the strategy we are going to take and it turns out this is not too bad computation you can do it without too much trouble, so let me show it in a slightly different way here.

So I have $c_1 = c_2 \oplus c_3$ okay and somebody tells me that l_2 is the log of probability of $C_2 = 0$ given r_2 divided by probability of $C_2 = 1$ given r_2 and then l_3 is log of probability of $C_3 = 0$ given r_3 divided by probability of $C_3 = 1$ given r_3 alright so this is given to me. From here one can go to this so maybe I will call this guy as p_2 I will call this guy has p_3 just for simplicity okay so remember once I call that as p_2 this is actually 1

minus p_2 okay so this also denominator as 1 minus p_3 okay so this is your probability of 0 or 1 both of them have to add together to give you one, so this is the same thing. So there are various ways to do this changing around so I_2 is $\log p_2$ by 1 minus p_2 right so from here you can find out p_2 as well so you can write down what p_2 will be in terms of I_2 and all that we will come to that a little bit later but just remember this conversion okay. So given p_2 and p_3 what is p_1 okay so that is the extrinsic okay.

(Refer Slide Time: 10:46)

Given $R^k Ps$: what is $P(c_1=0|R_2, R_3)$?

| | | |
|-------|-------|-------|
| c_1 | c_2 | c_3 |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

$$p_1 = p_2 * p_3 + (1-p_2)(1-p_3)$$

$$1-p_1 = p_2(1-p_3) + (1-p_2)p_3$$


PROF ANDREW THANGARA
IIT MADRAS

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check(SPC) Code

So how do I compute p_1 , so what is the probability, so given these 2 p_2 and p_3 what is probability of C_1 equals 0 let us say giving r_2, r_3 okay so that is the main question here. So if I can crack this if I can figure out from p_2, p_3 what this probability of C_1 equals 0 will be I would be ((0)(11:10) okay so now what is... so remember the r_2 and r_3 of a particular code word are independent otherwise they are not so needs to be little bit careful in this computation but I am not going to go through it is so rigorously I will... motivated and give you the final answer in a reasonable way okay. So if you look at the cases when C_1, C_2, C_3 . C_1 is 0 if C_2 and C_3 are 0 and C_2 and C_3 are 1. If C_1 is one if C_1 and C_2 01 or 10 okay, so this is how C_2 and C_3 decide what C_1 will be.

I am given only C_2 and C_3 I want to figure out what C_1 is this is how it works, so if you want to look at p_1 okay which is probability of C_1 equal 0 so maybe I should call this as p_1 okay. This is going to be right p_2 times p_3 it is the 1st term here plus 1 minus p_2 times 1 minus p_3 okay that is the 2nd term here okay so how do I know what C_1 is? C_1 is 0, C_2 is 0, C_3 is 0 that is p_2 times p_3 , C_2 is 1, C_3 is one which is one minus p_2 times 1 minus p_3 . The same way can also write 1 minus p_1 which is the probability of C_1 is 1 and that will be p_2 into 1 minus p_3

plus 1 minus p2 into p3 okay. This is nice okay seems decent enough okay, so the most interesting thing... so of course you can start working with this and see what you can derive, et cetera. So it turns out there is a way to slightly simplify this computation.

(Refer Slide Time: 13:10)

A screenshot of a presentation slide from NPTEL. The slide shows a 3x3 matrix with columns labeled $c_1 \leftarrow c_2 \oplus c_3$, c_2 , and c_3 . The rows are labeled 0, 0, 0; 0, 1, 1; 1, 0, 1. To the right of the matrix, the inverse is calculated as follows:

$$p_1 = p_2 p_3 + (1-p_2)(1-p_3)$$

$$1-p_1 = p_2(1-p_3) + (1-p_2)p_3$$

$$p_1 - (1-p_1) = p_2(p_3 - (1-p_3))$$

$$+ (1-p_2)((1-p_3) - p_3)$$

$$p_1 - (1-p_1) = (p_2 - (1-p_2))(p_3 - (1-p_3))$$

Below the equations, there is a portrait of Prof. Andrew Thangara.

A screenshot of a presentation slide from NPTEL. The slide shows the derivation of extrinsic information l_1, l_2, l_3 from the received symbols r_1, r_2, r_3 . It starts with the equation $c_1 = c_2 \oplus c_3$. The extrinsic information l_1 is given by:

$$l_1 = \log \frac{Pr(c_1=0|r_1)}{Pr(c_1=1|r_1)}$$

The extrinsic information l_2 is given by:

$$l_2 = \log \frac{Pr(c_2=0|r_2)}{Pr(c_2=1|r_2)}$$

The extrinsic information l_3 is given by:

$$l_3 = \log \frac{Pr(c_3=0|r_3)}{Pr(c_3=1|r_3)}$$

Below the equations, there is a portrait of Prof. Andrew Thangara.

NPTEL

extrinsic: what do r_2 and r_3 say about c_1 ?
 ↓
 Intrinsic
 $c_1 = c_2 \oplus c_3$
 $l_2 = \log \frac{P_r(c_2=0|r_2)}{P_r(c_2=1|r_2)}$
 $l_3 = \log \frac{P_r(c_3=0|r_3)}{P_r(c_3=1|r_3)}$

Given r_2 & r_3 : what is $P_r(c_1=0|r_2, r_3)$?
 $\log \frac{p_1}{1-p_1}$: extrinsic LLR

$c_1 \leftarrow c_2 \quad c_3$
 $0 \quad 0 \quad 0$
 $0 \quad 1 \quad 1$
 $1 \quad 0 \quad 1 \quad \dots \quad 1 \quad 1 \quad 1 \quad \rightarrow L$



PROF. ANDREW THANGARA]

IIT MADRAS

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check(SPC) Code

So for that one needs to look at p_1 minus 1 minus p_1 okay, so I am going to subtract these 2 and when you subtract these 2 you will see what will happen is you will get this product okay. So you will get p_2 times p_3 minus 1 minus p_3 plus 1 minus p_2 times 1 minus p_3 minus p_3 , right? So you can take things common here and you will get p_2 minus 1 minus p_2 times p_3 minus 1 minus p_3 .

It is a simple little observation but this will help us simplify a lot of things okay p_1 minus 1 minus p_1 equals p_2 minus 1 minus p_2 times p_3 minus 1 minus p_3 okay. So sounds simple enough but it is nice to do okay. So what people do? See remember I want to go from l_2 and l_3 to l_1 okay so that is what extrinsic or not to l_1 , l_2 and l_3 to the extrinsic information about p_1 , so actually what I am interested in, I am interested in the computation of the $\log p_1$ by 1 minus p_1 , so this is the extrinsic information, extensive LLR, right? So I need to compute this guy okay.

(Refer Slide Time: 14:36)



PROF ANDREW THANGARA]

IIT MADRAS

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check(SPC) Code



PROF ANDREW THANGARA]

IIT MADRAS

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check(SPC) Code

So how do I compute this? It turns out you can divide throughout by this guy, basically divide throughout by one and write 1 as different things here, this is allowed it is all just one I am writing it in a different way, so now I can divide by p_1 okay divide this ratio on the left-hand side by p_1 , this ratio by p_2 , this ratio by p_3 so you will get her 1 minus 1 minus p_1 by p_1 by 1 plus 1 minus p_1 by p_1 .

These are all just algebraic tricks but nevertheless it plays an important role in the implementation you will see how it works 1 plus 1 minus p_2 by p_2 1 minus 1 minus p_3 by p_3 by 1 plus 1 minus p_3 by p_3 okay. Okay so this is the expression I have, now remember I am interested in the extrinsic LLR okay $\log p_1$ by 1 minus p_1 okay so I am going define this as 1

$l_{ext,1}$ which is $\log p_1$ by 1 minus p_1 is the extrinsic LLR about bit 1 I am interested in this okay. So let us write everything in terms of the extrinsic LLR and l_2 and l_3 .

(Refer Slide Time: 15:45)

NPTEL

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

odd function: $\tanh(-x) = -\tanh(x)$

$$C_1 = C_2 \oplus C_3 \stackrel{\text{tanh rule}}{\Rightarrow} \tanh\left(\frac{l_{ext,1}}{2}\right) = \tanh\left(\frac{l_2}{2}\right) \cdot \tanh\left(\frac{l_3}{2}\right)$$



PROF ANDREW THANGARA]

IT MADRAS

Soft Input and Soft Output (SISO) Decoder for the Single Parity Check(SPC) Code

So how do I write that? So if you notice here this is 1 minus e power minus $l_{ext,1}$ why do I get the minus? It is because the ratios is inverted we will get the minus here 1plus e power minus $l_{ext,1}$ okay so likewise here you will have e power minus l_2 by 1 plus e power minus l_2 times 1 minus e power minus l_3 by 1 plus e power minus l_3 okay. So once again I will remind you about the tan hyperbolic function it is e power x minus e power minus x by e power x plus e power minus x .

In fact one can write this as 1 minus e power minus $2x$ by 1 plus e power minus $2x$ okay so it is a same thing. So once you notice that you will see that this product is just the tan hyperbolic of $l_{ext,1}$ by 2 equals tan hyperbolic of l_2 by 2 times tan hyperbolic of l_3 by 2. Okay so this is a very famous formula for XOR so if you have C_1 equals C_2 XOR C_3 it results in less tan h rule, so you can call it tan h rule.

You remember in the repetition code when C_1 and C_2 and C_3 were the same these extrinsic was simply $l_{ext,1}$ equals l_2 plus l_3 okay you just add the 2 okay in the repetition code you add the other incoming LLR to get extrinsic okay. In the SPC code single parity check code when you have C_1 equals C_2 XOR C_3 given l_2 and l_3 what is $l_{ext,1}$ you have to use the tan hyperbolic rule, tan hyperbolic of $l_{ext,1}$ by 2 equals tan hyperbolic of l_2 by 2 times tan hyperbolic of l_3 by 2 okay. Looks a little complicated but one can do this with some plots. In fact what people usually do is the following simplification okay. So tan hyperbolic is an odd

function okay and also if x is negative tan hyperbolic is less than 0, if x is positive tan hyperbolic is greater than 0 okay, so the sign of x and the sign of tan hyperbolic of x is exactly the same okay.

(Refer Slide Time: 18:11)

Signs : $\text{sgn}(l_{\text{ext},1}) = \text{sgn}(l_2) \cdot \text{sgn}(l_3)$

Absolute value : $\tanh\left(\frac{|l_{\text{ext},1}|}{2}\right) = \tanh\left(\frac{|l_2|}{2}\right) \tanh\left(\frac{|l_3|}{2}\right)$

$$\log \tanh\left(\frac{|l_{\text{ext},1}|}{2}\right) = \log \tanh\left(\frac{|l_2|}{2}\right) + \log \tanh\left(\frac{|l_3|}{2}\right)$$

$$f(|l_{\text{ext},1}|) = f(|l_2|) + f(|l_3|)$$

$$|l_{\text{ext},1}| = f(f(|l_2|) + f(|l_3|))$$

$$\text{sgn}(l_{\text{ext},1}) = \text{sgn}(l_2) \text{ sgn}(l_3)$$

$f'(x) = 1 - \tanh^2(x)$

$f''(x) = 2\tanh(x)(1 - \tanh^2(x))$

PROF ANDREW THANGARA
IIT MADRAS

SISO decoder for SPC code

$(3,2)$ code

$m = [m_1, m_2]$

$\underline{c} = [c_1, c_2, c_3]$

SPC code

BPSK

$r = [r_1, r_2, r_3]$

SISO decoder

$[l_1, l_2, l_3]$

$l_1 \leftarrow$ intrinsic of r_1 ,
extrinsic : what do r_2 and r_3 say about c_1 ?

c_2 c_3

$c_1 = c_2 \oplus c_3$

parity check condition

PROF ANDREW THANGARA
IIT MADRAS

So this equation involving tan hyperbolic can be written in 2 different ways one is sign of $l_{\text{ext},1}$ equals sign of l_2 times the sign of l_3 this is just the signs okay and then absolute value okay, so you can take the absolute value separately and you will get tan hyperbolic of absolute value of $l_{\text{ext},1}$ divide by 2 equals tan hyperbolic of absolute value of l_2 divided by 2 times tan hyperbolic of absolute value of l_3 divided by 2.

So what is nice about dealing with the absolute value and positive things, product can become addition by taking log okay once everything is positive I can take log, when I take log

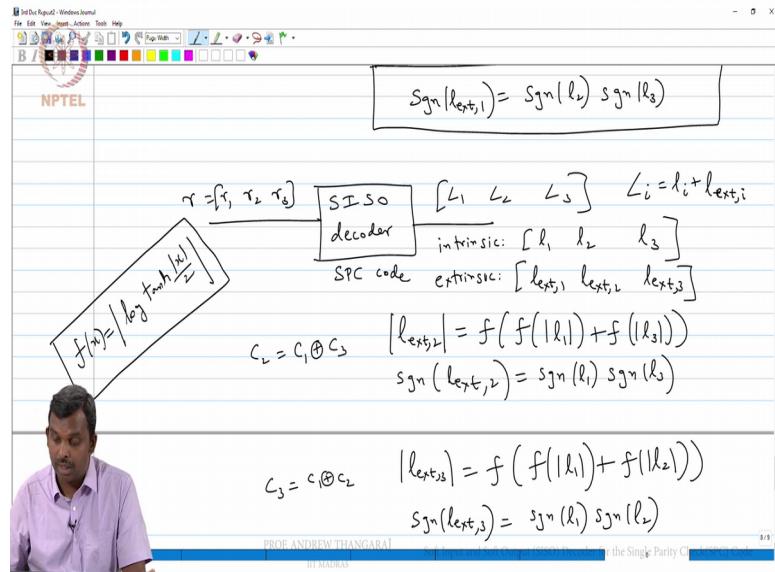
everything becomes a bit different so let us look at this log tan hyperbolic of absolute value of $l_{ext,1}$ by 2 equals log tan hyperbolic of absolute value of l_2 by 2 plus log tan hyperbolic absolute value of l_3 by 2 okay so that is a nice enough formula here I am not doing any multiplication I am just adding, so at this point we are going to define function F which conveniently captures this log tan hyperbolic.

So you see the log tan hyperbolic of the quantity by 2 is occurring again and again so it is good to define function F which captures this log tan hyperbolic so let us do that for x greater than 0 we are going to define f of x has absolute value of log tan hyperbolic of absolute value of x by 2 okay so this is the definition. So if you look at it closely tan hyperbolic of absolute value of x by 2 sort of captures this and give you take log you will get log that is fine but it turns out that this log actually...

So tan hyperbolic will be a value between 0 and 1, so if you take log you will get something negative okay so I do not want the negative part there I just want the absolute value, so put absolute value here okay so that is how it works and one curious fact about this function is f inverse of x is actually the same as f of x okay so this is the function here and this is important okay. So once we have this function we can conveniently express this result in terms of its absolute value and its sign in a very easy manner okay.

This equality is F of $l_{ext,1}$ absolute value equals f of l_2 plus f of l_3 , right? Okay so now how do I invert this? Take f on both sides we will simply get absolute value of $l_{ext,1}$ equals f of f of absolute value of l_2 plus f of absolute value of l_3 , so is this another way to write down the tan h rule. This and sign of $l_{ext,1}$ equals sign of l_2 times sign of l_3 okay. So this was the derivation for the...this soft input soft output decoder for single parity check code, so now let me summarise this is a bit of long derivation and I think...so the derivation is slightly important to know where these things come from but finally the implementation is equally important let me tell you where all of these get finally implemented as.

(Refer Slide Time: 22:03)



So when you want to build SISO decoder for the SPC code what is going to be your capital L1, capital L2 capital L3 okay, so you have the intrinsic which is l1, l2, l3 and you have the extrinsic which is l ext,1 l ext,2 l ext,3 okay and your capital Li is small li plus l ext,i okay just the addition of these 2. How do I find l ext,1? You use this formula here how to find l ext,2? Okay so is not very hard you simply use the same formula before except that you use C2 equals C1 XOR C3 okay so how do I find what C2 is? What C1 and C3 are telling me about C2.

So wherever you have 2 and 3 you replace them with 1 and 3 you will get the l ext,2 okay so here we will have absolute value is f of f of absolute value of l1 plus f of absolute value of l3 then sign of l ext,2 sign of l1 multiplied by sign of l3 okay and then let us just look at l ext,3 which comes from C3 equals C1 XOR C2 here again f of f of absolute value of l1 plus f of absolute value of l2 then sign of l ext,3 equal sign of l1 times sign of l2 okay so this is how it works.

So there are 3 values to compute here, the softer output l1, l2 and l3 and you need this option F okay once again let me remind you what this function f is so what is f of x let me remind you what f of x is we define this little bit earlier f of x is absolute value of log tan hyperbolic of absolute value of x by 2 okay so this is f of x and we use f of x in this decoder. So now if you stand at it a bit more closely you can do a little bit of simplification here okay so you can compute all the three values in one interesting way.

(Refer Slide Time: 24:53)

So what you can do is you can find 2 values here, one is capital S which is f of absolute value of l_1 plus f of absolute value of l_2 plus f of absolute value of l_3 then you can find the capital P here which is sign of l_1 times sign of l_2 times sign of l_3 okay and what is $l_{ext,1}$? Is S minus f of l_1 okay times P times sign of l_1 okay so convince yourself that that is true, this is the absolute value...I am sorry I think I missed f of okay.

Okay it is f of S minus f of l_1 and this is you can see this is the sign, right? What happen P is a product of all the signs, I am multiplying with sign of l_1 , so this will become sign square of l_1 but sign square of l_1 is just one okay so it will just be sign of l_2 times sign of l_3 okay. So likewise you can also find $l_{ext,2}$ is f of S minus f of l_2 times P times sign of l_2 and $l_{ext,3}$ is f of S minus f of l_3 times P times sign of l_3 okay so this is the 3 extrinsic LLR and once you find the 3 extrinsic LLR's you can simply add to the intrinsic LLR and you will get the total output LLR.

So this is the derivation of the SISO decoder for the single parity check code that brings us to the end of this lecture. What I will do in the next lecture is write some simple Matlab code to illustrate how the repetition SISO decoder for repetition code can be written down, how the SISO decoder for the single parity check code can be written down well these are simple relatively simple implementations and then we will see some simplifications of the decoder, so in fact one of the things that is not very nice about this single parity check code decoder is this nonlinear function log tan hyperbolic, so how do we get rid of that? Are there good approximation to get rid of that nonlinear function? Turns out the answer is yes we will do

that we will do that in the next lecture, so the next lecture will have some Matlab coding along with a simplification for the SISO decoder for SPC, thank you.