Mueller Exhibit 13

View: Next in topic | Previous in topic
Next by same author | Previous by same author
Previous page *(August 1999)* | Back to main 3GPP_TSG_RAN_WG1 page
Join or leave 3GPP_TSG_RAN_WG1
Reply | Post a new message
Search
Log in

Options: Chronologically | Most recent first
Proportional font | Non-proportional font

Date: Wed, 11 Aug 1999 08:44:31 +0900
Reply-To: jaeyoel Kim <[log in to unmask]>
Sender: "3GPP_TSG_RAN_WG1: TSG RAN Working Group 1" <[log in to unmask]>
From: jaeyoel Kim <[log in to unmask]>
Subject: Re: AH10 : R1-99915 "Multiple scrambling code"
Content-Type: text/plain; charset="ISO-2022-KR"

Dear Fredrik,
Thank you for your comment.
As you understand, we proposed shifting scheme before our proposal.
There are several reasons to do that.
Please think about and give us your comment.

Even though initial state loading looks simple, but there are several problems.
Of course, it is o.k. if you want to generate only one scrambling code.
However, we need several generators are needed at the same time in real situation.
One for primary or one for secondary or one for other cells...
Followings are problems of initial state loading method

1. Calculating masking function from initial state is almost impossible (discrete logarithm problem)
2. Complexity of masking is not desirable.

Of course, it is fine too if you want to have multiple independent generators.
I think that requires only one solution and does not provide a flexible and an efficient implementation.
we are not happy too.

However, if we simply change a description, then we can have many way to implement.
If you want to implement with independent genrator, it is fine.
If you want to implement with masking function, it is also easy to do that.

I realized this is more fundamental problem.
With current description, I think we have some problem.

Best wishes
Jaeyoel Kim

-----¿ ø º ≫ ¸ Þ½AAo-----
º ¸ ³ ½ ≫c¶ ÷: Fredrik Ovesjo <[log in to unmask]>
¹ Þ ´A ≫c¶ ÷: [log in to unmask] <[log in to unmask]>
³ ?A¥: 1999³ a 8¿u 10Al E¿aAl ¿AEA 7:58
A¦ ¸n: Re: AH10 : R1-99915 "Multiple scrambling code"

>Dear scrambling code experts,
>
>I have reread (99)915, and I am no longer sure that I can support
>option 2 in that document.
>

>It seems that in (99)915 another generation of the code of number K is
>used than is currently assumed in the specs. To generate code number K
>currently, one only loads the upper register with the binary
>equivalent of K. Very simple.
>
>However, in (99)915 it seems (please confirm), that the initial state
>of the upper register is calculated as a phase shift of the sequence
>initialised by 0000000000...0001. This would mean that some shifting
>algorithm would be needed, as previously proposed by Nokia.
>
>I think we have already had that discussion about the shifting
>algorithm, and that several companies were not too happy with that.
>
>Hence, with this new information (please forgive me for
>misunderstanding before), I would agree with Peter that we should not
>accept the proposed scheme in (99)915. Too me it seems that the
>potential benefits are too small to justify the neccessary
>modifications of our current assumptions.
>
>Best regards / Fredrik
>
>Fredrik Ovesjo wrote:
>>
>> Peter,
>>
>> as I have understood this paper, option 2 is only a redefinition of
>> what code numbers that belong to the primary and its associated
>> secondary scrambling code group.
>>
>> Hence, in order to introduce option 2 in the spec, only a very minor
>> change needs to be done (definition of the groups). I do not see that
>> the generation of the codes needs to be redefined in any way. However,
>> I do not want this masking function to become a part of the
>> specification text, since it is an implementation matter.
>>
>> I indicated to Samsung offline that I would be willing to accept
>> option 2 with the change that we have at least 15 secondary scrambling
>> codes per primary code, and that the text proposal is such that only
>> the grouping of the code numbers was changed in the spec.
>>
>> Regards / Fredrik
>>
>> Chambers, Peter wrote:
>> >
>> > Dear scrambling fans
>> >
>> > I wish to discuss Tdoc 915 (Samsung) which is best discussed by e-mail
>> > rather than at a meeting.
>> >
>> > This paper proposes a non-trivial change to the working assumption on
>> > downlink scrambling
>> > code generation. I wish to show that the problems discussed by the
>> > proponents are not so
>> > severe as they write and that it is safest to keep to the working
>> > assumption.
>> >
>> > Firstly the proponents discuss some "problems with the current method":
>> >
>> > - they say that the simplest method is to have multiple generators but with
>> > a masking function
>> >   that complexity is decreased, but
>> >
>> >   i) this is an implementational matter and not a standards issue.
>> > Manufacturers are free
>> >   to use variable masking if they wish though the extra gates add
>> > complexity.
>> >
>> >   ii) the complexity of the current Gold code generator is minimised in
>> > terms of the fixed
>> >   masking function suggested. A general masking function would add
>> > complexity especially

>> > if it were run time programmable.
>> >
>> > - the proponents guess that Tdoc 724 is motivated by a mapping designed to
>> > facilitate a mapping
>> > function based on a mapping "between number and real code". This does not
>> > seem to be the case
>> > as the scrambling code number is simply the number used to initialise the
>> > scrambling code generator
>> > and is a linear function of primary and secondary scrambling code numbers.
>> > The complexity of
>> > initialisation is very low in hardware and software. No masking is
>> > required or seems to be implied.
>> >
>> > - the proponents see problems with over provision of secondary scrambling
>> > codes by layer 1.
>> > If this is problem with higher layer signalling then a limited number only
>> > need by used.
>> > The proponents suggest 4 secondary codes might be used. For spot beams
>> > slightly more
>> > may be used. However over provision is no problem. Lack of provision would
>> > be a problem.
>> >
>> > In the section dealing with the current method two objections are raised:
>> >
>> > 1. code0 would be a problem for a masking implementation.
>> >
>> > 2. there is no simple rule to find masking in the current scheme
>> > (manufacturers would have to implement one by calculation)
>> >
>> > These are really problems with the masking approach which re-use the prime
>> > movers of
>> > m-sequence generators using linear algebra to provide shifted sequences.
>> > This is not
>> > a standards issue but the limitation of the hardware architecture solution
>> > propounded in
>> > the paper. Other architectures which have been discussed by several
>> > companies (both
>> > in Fibonacci and Galois forms) do not suffer from this problem. Indeed in
>> > low power
>> > UEs the current text is more suitable for hardware than it might be (for
>> > DSPs) in the base station.
>> >
>> > For the above reasons I would suggest that Tdoc 915 not be approved.
>> >
>> > regards, Peter Chambers, Roke Manor Research
>>
>> --
>> Fredrik Ovesjo - Radio Access and Antenna Systems Research
>>
>> Ericsson Radio Systems AB Tel: +46 8 404 56 74
>> SE-164 80 Stockholm, SWEDEN GSM +46 70 376 22 50
>> [log in to unmask] Fax: +46 8 585 314 80
>
>--
>Fredrik Ovesjo - Radio Access and Antenna Systems Research
>
>Ericsson Radio Systems AB Tel: +46 8 404 56 74
>SE-164 80 Stockholm, SWEDEN GSM +46 70 376 22 50
>[log in to unmask] Fax: +46 8 585 314 80

Back to: Top of message | Previous page | Main 3GPP_TSG_RAN_WG1 page

View: Next in topic | Previous in topic
Next by same author | Previous by same author
Previous page *(July 1999)* | Back to main 3GPP_TSG_RAN_WG1 page
Join or leave 3GPP_TSG_RAN_WG1
Reply | Post a new message
Search
Log in

Options: Chronologically | Most recent first
Proportional font | Non-proportional font

Date: Thu, 22 Jul 1999 15:01:50 +0900
Reply-To: [log in to unmask]
Sender: "3GPP_TSG_RAN_WG1: TSG RAN Working Group 1" <[log in to unmask]>
From: "(Changsoo PARK)" <[log in to unmask]>
Subject: Re: AH10:R1-99915"Multiple scrambling code"
Content-Type: text/plain; charset=EUC-KR; name="mail.txt"

Dear Peter and all,
Thank you for having an interest in our proposal.
I think there are some misunderstanding in our proposal. Please see below.
Actually, there were supports from several companies like Ericsson and Panasonic during offline discussion.
Only comment in the meeting was the number of secondary scrambling code from Ericsson.
They want to have 15 as maximum, and we don't have problem with that.

Our proposal includes one change and two small suggestions for the current text.
Let me address two small suggetion first.

1.
>
>- the proponents see problems with over provision of secondary scrambling
>codes by layer 1.
> If this is problem with higher layer signalling then a limited number only
>need by used.
> The proponents suggest 4 secondary codes might be used. For spot beams
>slightly more
> may be used. However over provision is no problem. Lack of provision would
>be a problem.
>
There is an editor's note in the current text about uncertainty of the number of secondary scrambling code.
This issue is supposed to be handeled before we fix release 99 to reduce signaling overhead.
So, we suggest 4 secondary scrambling code as maximum, and Ericsson suggested 15.
I think 15 is a very safe number, and I am o.k. with that.

Some other company can make comment regarding this.

I think that deciding the number of secondary scrambling code is a L1 issue not a higher layer because that decision
will be based on the real capacity.
2.
>- the proponents guess that Tdoc 724 is motivated by a mapping designed to
>facilitate a mapping
> function based on a mapping "between number and real code". This does not
>seem to be the case
> as the scrambling code number is simply the number used to initialise the
>scrambling code generator
> and is a linear function of primary and secondary scrambling code numbers.
>The complexity of
> initialisation is very low in hardware and software. No masking is
>required or seems to be implied.
>
The reason behind our guess is following.
When we use secondary scrambling code, both BS and UE must have at least two generators at the same time.
One for primary, the others for secondary
For example, one for BCH, one for DCH.
Initialization is of course easy, but we don't want to have seperate generators, so initialization for second
generator is not necessary.
Using maskng function is pretty well known technic for this case.
We can have second generator with almost no complexity increase if a masking function is simple whatever
implementation is done with a hardware or a software.
Furthermore, calculating masking function by two initial state is not possible.
So, we suggest a rephrase of the current text.

3. Therefore, we proposed to change the current way to assign primary and secondary scrambling codes.
Option 2 seems to be the simplest and the best way to minimize complexity.

>Firstly the proponents discuss some "problems with the current method":
>
>- they say that the simplest method is to have multiple generators but with
>a masking function
> that complexity is decreased, but
>
>   i) this is an implementational matter and not a standards issue.
>Manufacturers are free
>   to use variable masking if they wish though the extra gates add
>complexity.
>
>   ii) the complexity of the current Gold code generator is minimised in
>terms of the fixed
>   masking function suggested. A general masking function would add
>complexity especially
>   if it were run time programmable.
>
Regarding first comment,

Some issue is related with implementation complexity.
Are you saying implemetation complexity is not important?
Then, I like to hear about your opinion on Nokia's contribution Tdoc 588.
That is also about suggestion to decrease complexity and I think it is also valuable.
With the current method, we can not avoid some complexity increase whatever the method is (Hardware, software, Fibonacci or Galois)
We need a rule to assign codes anyway in the text, then we should have a nice rule to minimize a complexity.
Regarding second comment,
I think you assumed seperate secondary scrambling code gerneerator which is not our subject.
>
>In the section dealing with the current method two objections are raised:
>
>1. code0 would be a problem for a masking implementation.
>
>2. there is no simple rule to find masking in the current scheme
> (manufacturers would have to implement one by calculation)
>
>These are really problems with the masking approach which re-use the prime
>movers of
>m-sequence generators using linear algebra to provide shifted sequences.
>This is not
>a standards issue but the limitation of the hardware architecture solution
>propounded in
>the paper. Other architectures which have been discussed by several
>companies (both
>in Fibonacci and Galois forms) do not suffer from this problem. Indeed in
>low power
>UEs the current text is more suitable for hardware than it might be (for
>DSPs) in the base station.
>
Actually, I don't understand whole pharagraph because the current description of the text is in Fibonacci form.

Moreover, regardless of the expression, all problems we mentioned are valid.
Fibonacci form is different from Galois form in terms of H/W logic.
But two different forms are specific representation forms of LFSR sequence, and

they are equivalent each other in terms of LFSR Theory. (See ""Shift Register", - Golomb, Holiday)
Mask problem depend on a base finite field of sequence, not, H/W logic.
That is, the masking operation is based on the fact that a shift version of LFSR sequence can be represented by a sum of some shift version.
The reason is as follows.
Actually, LFSR sequence is based on the finite field, and there is a one-to-one

mapping of each shifted version onto exactly one element in finite fields.

Therefore, a finite field is equivalent to a set of all shifted versions of LFSR sequence.
Since the finite field is a vector space, there is a basis element of a finite field, and the corresponding

shifted version is a basis element of the set of all shifted versions of LFSR sequence.
Hence, all shifted versions of LFSR sequence are represented by a linear combination of some basis element

This proposal recommend simply to change the current assignment rule for primary and secondary scrambling code.
This proposal does not harm anything, but provide simple and nice implementation.
If you want implement several seperate generators, then you can do that with our proposal. No extra cost.
However, if you want to share most of logic, then you will have a benefit from our proposal.
I think most of misunderstanding comes from the assumption about secondary scrambling code.
Regardless of software or hardware implementation, having several seperate generators is not desirable at all.
With option 2, you can only change input from the primary code generator for the generation of secondary scrambling code. No other operation is necessary from the generator of primary scrambling code.

I hope this give you a proper explanation.

Best wishes
Jaeyoel Kim

-----¿ ø º ≫ ¸ Þ½AAo-----
º ¸ ³ ½ ≫c¶ ÷: Chambers, Peter <[log in to unmask]>
¹ Þ ´A ≫c¶ ÷: [log in to unmask] <[log in to unmask]>
³ ?A¥: 1999³ a 7¿u 21Al ¼o¿aAl ¿AAu 1:56
A¦¸n: AH10 : R1-99915 "Multiple scrambling code"

>Dear scrambling fans
>
>I wish to discuss Tdoc 915 (Samsung) which is best discussed by e-mail
>rather than at a meeting.
>
>This paper proposes a non-trivial change to the working assumption on
>downlink scrambling
>code generation. I wish to show that the problems discussed by the
>proponents are not so
>severe as they write and that it is safest to keep to the working
>assumption.
>
>Firstly the proponents discuss some "problems with the current method":
>

>- they say that the simplest method is to have multiple generators but with
>a masking function
> that complexity is decreased, but
>
>   i) this is an implementational matter and not a standards issue.
>Manufacturers are free
>   to use variable masking if they wish though the extra gates add
>complexity.
>
>   ii) the complexity of the current Gold code generator is minimised in
>terms of the fixed
>   masking function suggested. A general masking function would add
>complexity especially
>   if it were run time programmable.
>
>- the proponents guess that Tdoc 724 is motivated by a mapping designed to
>facilitate a mapping
> function based on a mapping "between number and real code". This does not
>seem to be the case
> as the scrambling code number is simply the number used to initialise the
>scrambling code generator
> and is a linear function of primary and secondary scrambling code numbers.
>The complexity of
> initialisation is very low in hardware and software. No masking is
>required or seems to be implied.
>
>- the proponents see problems with over provision of secondary scrambling
>codes by layer 1.
> If this is problem with higher layer signalling then a limited number only
>need by used.
> The proponents suggest 4 secondary codes might be used. For spot beams
>slightly more
> may be used. However over provision is no problem. Lack of provision would
>be a problem.
>
>In the section dealing with the current method two objections are raised:
>
>1.   code0 would be a problem for a masking implementation.
>
>2.   there is no simple rule to find masking in the current scheme
>   (manufacturers would have to implement one by calculation)
>
>These are really problems with the masking approach which re-use the prime
>movers of
>m-sequence generators using linear algebra to provide shifted sequences.
>This is not
>a standards issue but the limitation of the hardware architecture solution
>propounded in
>the paper. Other architectures which have been discussed by several
>companies (both
>in Fibonacci and Galois forms) do not suffer from this problem. Indeed in
>low power
>UEs the current text is more suitable for hardware than it might be (for
>DSPs) in the base station.
>
>For the above reasons I would suggest that Tdoc 915 not be approved.
>
>regards, Peter Chambers, Roke Manor Research

Back to: Top of message | Previous page | Main 3GPP_TSG_RAN_WG1 page

View: Next in topic | Previous in topic
Next by same author | Previous by same author
Previous page *(August 1999)* | Back to main 3GPP_TSG_RAN_WG1 page
Join or leave 3GPP_TSG_RAN_WG1
Reply | Post a new message
Search
Log in

Options: Chronologically | Most recent first
Proportional font | Non-proportional font

Date: Thu, 12 Aug 1999 18:37:37 +0900
Reply-To: jaeyoel Kim <[log in to unmask]>
Sender: "3GPP_TSG_RAN_WG1: TSG RAN Working Group 1" <[log in to unmask]>
From: jaeyoel Kim <[log in to unmask]>
Subject: Re: AH10 : R1-99915 "Multiple scrambling code"
Content-Type: text/plain; charset="us-ascii"

Dear all,
It seems that everybody is back from vacation and many people get involved in this issue now.
Let me speak my opinion briefly.

I think people start to think the possibility for generating mutiple scrambling code simultaneously.
To me, simple loading is not that simple when we consider to generaton of multiple codes at the same time.
As I said before, initial state loading cannot be implemented by a masking function because calculation is almost impossible.
So, we need several independent generators for primary, secondary or other primary scrambling code.
This complexity burden goes to UE rather than Utran.
I think this gives us only one way to implement and make impossible to implement by software.

What I don't understand is why people think we need a big change.
We just need to change a description for initial state a little bit.
Furthermore, we can also have the same implementation method with the current one.(several independent generators with rather simple calculation as esko mentioned before)
We don't have to change a figure in the text.

Dear Esko,
I think you understand most of issue correctly.
There is small error in your second paragraph. I think you did not intend.
Option 2 is not idea of shift of initial state but shift of sequence since this option assume shifting scheme.
I think there are several possible ways to keep the current implementation (initial state loading) with new descrption.
1. Utran and UE get initial state with simple calculation since offset distance is known and constant.
2. Utran and UE get initial state by shifting upper generator, it will delay 8192 cycle for maximum since we only use 16*512 =8192 codes.
3.Utran calculate state and let UE know as you mentioned. In this case we need 18 bits to broadcast and UE may have no idea if it does not have an information for other cell.

Dear Peter,
I appricate your effort to compromise.
I think your figure 3 looks like option 3 of our proposal.
Would you elaborate this little bit more?
I think whatever we have a initial state loading method or shifting method, we have no uncertainty about distinction.
We can define 2^18 distinct codes in both methods.
Would you explain this too?

Best wishes
Jaeyoel

Back to: Top of message | Previous page | Main 3GPP_TSG_RAN_WG1 page