3GPP TSG-RAN WG2 #115-e R2-21xxxxx

Online, 16-27 August 2021

Agenda Item: xx

Source: MediaTek Inc.

**Title: Report of email discussion [Post114-e][071][NR16] CandidateBeamRSList set to release (MediaTek)**

Document for: Discussion, decision

# 1 Introduction

This document is a report on the following email discussion, initiated after RAN2#114-e:

* [Post114-e][071][NR16] CandidateBeamRSList set to release (MediaTek)

Scope: how UE shall handle the extension field of candidateBeamRSList. The intention is to agree a 38.331 clarification CR in next meeting. Could consider option 2 and option 3 proposed in R2-2106115 as a starting point. This was also discussed in [AT114-e][022].

Intended outcome: Report, agreeable CR.

Deadline: Long

The discussion will proceed in two phases, first to determine an agreeable mechanism for handling the extension field and second to converge on an agreeable CR. The deadlines are as follows:

Phase 1: Friday 2 July 1700 UTC

Phase 2: Friday 6 August 0900 UTC

# 2 Contact Points

Respondents to the email discussion are kindly asked to fill in the following table.

|  |  |  |
| --- | --- | --- |
| Company | Name | Email Address |
| MediaTek (rapporteur) | Nathan Tenny | nathan.tenny@mediatek.com |
| ZTE | Liu yu | liu.yu3@zte.com.cn |
| Intel | Sudeep Palat | Sudeep.k.palat@intel.com |
| Nokia, Nokia Shanhhai Bell | Tero Henttonen | tero.henttonen@nokia.com |
| Ericsson | Håkan Palm | hakan.l.palm@ericsson.com |
| Huawei, HiSilicon | Simone Provvedi | Simone.provvedi@huawei.com |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 3 Background

## 3.1 Original options

The discussion from [1] considered three options drawn from the discussion in [2]:

* **Option 1:** The UE releases the entire concatenated list, both the entries configured with *candidateBeamRSList* and the entries configured with *candidateBeamRSListExt-v1610*.
* **Option 2:** The UE releases only the extended entries that were configured with *candidateBeamRSListExt-v1610*.
* **Option 3:** The *release* branch is not used, and the UE treats *candidateBeamRSList* and *candidateBeamRSListExt-v1610* as a single concatenated field with Need M. The extended list *candidateBeamRSListExt-v1610* is only included when *candidateBeamRSList* is included and fully populated.

## 3.2 Updated options for this discussion

In the discussion, option 1 had less support compared to options 2 and 3. Rapporteur also understands that in continued offline discussion (separate from the official email discussion), a network-based restriction was proposed, in which the network is required to signal the extension (*candidateBeamRSListExt-v1610*) whenever it wants the extension entries to remain unchanged in the UE, and the “release” option on the extension list is used only when the network intends to reconfigure the UE to a number of entries fitting within the original list. This option (option C below) disambiguates the UE behaviour by having the network always indicate explicitly the fate of the extension entries.

Accordingly, this discussion considers three options:

* **Option A:** When *candidateBeamRSListExt-v1610* is set to *release*, the UE releases only the extended entries that were configured with *candidateBeamRSListExt-v1610*.
* **Option B:** The *release* branch is not used, and the UE treats *candidateBeamRSList* and *candidateBeamRSListExt-v1610* as a single concatenated field with Need M. The extended list *candidateBeamRSListExt-v1610* is only included when *candidateBeamRSList* is included and fully populated.
* **Option C:** The network is required to signal the extension (*candidateBeamRSListExt-v1610*) whenever it wants the extension entries to remain unchanged in the UE, and the *release* option on the extension list is used only when the network intends to reconfigure the UE to a number of entries fitting within the original list.

The details of the options may require some clarification (e.g. how to define the “extended entries” in option A), so the following discussion subsections include space for discussion of the details of each option.

## 3.3 Examples

For clarity, this section illustrates how options A/B/C would operate in two example scenarios.

**Example 1:** The network reduces the list size while extension entries are configured, and the resulting list is still larger than the legacy list size:

1. Network sends a *BeamFailureRecoveryConfig* containing a fully populated *candidateBeamRSList* (16 entries) and a partly populated *candidateBeamRSListExt-v1610* (2 entries).



1. UE concatenates the fields into a single list of 18 entries.



1. Network sends a *BeamFailureRecoveryConfig* containing a partly populated *candidateBeamRSList* (15 entries) and omitting the *candidateBeamRSListExt-v1610*.
   1. With option A, the UE populates a list of 17 entries, and the handling of a future *release* indication depends on the interpretation of the option (see section 4.1 below).



* 1. With option B, the UE populates a list of 15 entries; the 2 entries from *candidateBeamRSListExt-v1610* are released.



* 1. With option C, this step is not allowed; if the network intends to reconfigure the UE to a list of 15 entries, it needs to include the 15 entries explicitly along with the *candidateBeamRSListExt-v1610* set to *release*, and if it intends to reconfigure the UE to a list of 17 entries, it needs to include the 17 entries explicitly.

**Example 2:** The network reduces the list size while extension entries are configured, and the resulting list fits inside the legacy list size:

1. Network sends a *BeamFailureRecoveryConfig* containing a fully populated *candidateBeamRSList* (16 entries) and a partly populated *candidateBeamRSListExt-v1610* (2 entries).



1. UE concatenates the fields into a single list of 18 entries.



1. Network sends a *BeamFailureRecoveryConfig* containing a partly populated *candidateBeamRSList* (10 entries) and omitting the *candidateBeamRSListExt-v1610*.
   1. With option A, the UE populates a list of 12 entries, and the handling of a future *release* indication depends on the interpretation of the option (see section 4.1 below).



* 1. With option B, the UE populates a list of 10 entries; the 2 entries from *candidateBeamRSListExt-v1610* are released.



* 1. With option C, this step is not allowed; if the network intends to reconfigure the UE to a list of either r10 or 12 entries, it needs to include the entries explicitly along with the *candidateBeamRSListExt-v1610* set to *release*.

# 4 Discussion (Phase 1)

## 4.1 Details of option A

During the previous discussion of option A, it became clear that there are two potential understandings of the definition of “only the extended entries”.

* Approach A.1: The UE remembers which list entries were *initially* configured by *candidateBeamRSListExt-v1610*, and subsequently treats these as being the extension entries. Thus, even if the list is later shortened to a length that fits within the original list size, some entries may be marked as “extended entries” and can be released with the extension field.
  + In Example 1 from section 3.3, sending a *release* would release the 2 entries that were initially configured by *candidateBeamRSListExt-v1610*, leaving the UE with a list of 15 entries.



* + In Example 2 from section 3.3, sending a *release* would release the 2 entries that were initially configured by *candidateBeamRSListExt-v1610*, leaving the UE with a list of 10 entries.



* Approach A.2: The UE treats the entries from the two list fields as a single undifferentiated list (as usual for lists without ToAddMod structure), and the extension field only addresses entries beyond the size of the original list. Thus, if the list is shortened to a length that fits within the original list size, the UE considers that it has no more extended entries, and setting *candidateBeamRSListExt-v1610* to *release* becomes vacuous.
  + In Example 1 from section 3.3, sending a *release* would release the 1 entry that exceeds the legacy list size, leaving the UE with a list of 16 entries.



* + In Example 2 from section 3.3, sending a *release* would have no effect.



**Question 1.1:** Which of the two approaches do companies prefer, within the scope of option A?

|  |  |  |
| --- | --- | --- |
| **Company** | **Preferred Approach** | **Comments** |
| MediaTek | A.1 | The extension field is Need M, which means that if it is omitted rather than set to *release*, it should be considered to maintain its previous contents. This suggests that we should view *candidateBeamRSListExt-v1610* as a field separate from *candidateBeamRSList*, and the UE should be aware of which entries were configured with the extended list (so that it knows what to maintain in case the field is omitted). We don’t find this to be a problem for the implementation to remember, |
| ZTE | A.1 | If UE and NW store the R15/R16 list separately, NW can reconfigure the R15 list and R16 list separately, and this can save the signaling length effectively. |
| Intel | A2 | This is based on our understanding of A1 and A2 as given in response to Q1.2. |
| CATT | A.1 |  |
| Nokia, Nokia Shanghai Bell | A2 but no strong view | Generally, the UE is supposed to concatenate the entries to a single list so A.2 seems most aligned with that option. However, it has also some issues as per our replies for Q1.2 (see below). |
| Ericsson | A.1 | A.1 is more aligned with the “separated list” view, as indicated by MediaTek and Intel. But there are open ends as explained by Intel. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Rapporteur’s summary:** The responses showed a majority for A.1, but there are some concerns about the understanding of both options as discussed in Q1.2 below.

**Question 1.2:** Any other comment on the details of option A?

|  |  |
| --- | --- |
| **Company** | **Comments** |
| Intel | I tried to summarise my understanding of A1 and A2 but some of it is still not clear to me (shown with ?).  Initial condition: # of entries >16   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces only the elements that was previously signalled by original list? | Replaces only the original list | | Extension list without legacy list | Replaces entries signalled previously by ext list? | Replaces entries >16? | | Ext list with release | Release entries that were previously signalled by extension list | Release entries >16 | | Original+ext list (ext list configures new elements) | Replaces entries previously signalled by the original list and entries signalled by the ext list | Replaces both lists  (conf of ext list allowed if signalled original list is more than 16?) | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases entries signalled by the ext list? | Replaces entries <16 and releases entries >16 ? |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces the entire list? Including entries previously signalled by ext list? | Replaces the entries signalled by original list? | | Extension list without legacy list | Replaces entries previously signalled by ext list? | N/A | | Ext list with release | Release entries that were previously signalled by extension list | N/A | | Original+ext list (ext list configures new elements) | Replaces both lists (is it allowed if original list is less than 16?) | Replaces both lists (allowed if original list is more than 16?) | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases entries signalled by the ext list? | Replaces entries <16 and releases entries >16 ? | |
| Nokia, Nokia Shanghai Bell | Below is our interpretation of the options A1/A2 (red text shows differences to Intel version)  Initial condition: # of entries >16   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces only the elements that was previously signalled by original list, extension list elements remain | Replaces the entire list, i.e. releases also the extended list? | | Extension list without legacy list | Adds or modifies entries signalled previously by ext list (i.e. CANNOT be used to modify legacy list entries!) | Adds or modifies entries signalled previously either list (i.e. CAN be used to modify legacy list entries!) | | Ext list with release | Release entries that were previously signalled by extension list (e.g. what if there are 17 entries, 15 of which were originally added with extension list, all 15 of those are released and only 2 remain) | Release entries >16, but it's unclear which entries (e.g. what if there are 17 entries, 15 of which were originally added with extension list?) | | Original+ext list (ext list configures new elements) | Replaces entries previously signalled by the original list and entries signalled by the ext list | Replaces both lists  (conf of ext list is only allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases (any) entries signalled by the ext list? | Replaces entries <16 and releases entries >16, but it's unclear which entries (e.g. what if there are 17 entries, 15 of which were originally added with extension list?) |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces the entire legacy list, (i.e. only entries configured by the legacy list, extended elements remain) | Replaces the entire list (regardless of where the entries were signalled) | | Extension list without legacy list | Only allowed when legacy list size = 16 (i.e. CANNOT be used to modify legacy list entries) | Only allowed when legacy list size = 16, but only to add new entries | | Ext list with release | Release entries that were previously signalled by extension list | N/A (if list size < 16) | | Original+ext list (ext list configures new elements) | Replaces legacy list and adds to the extended list (only allowed if legacy list size = 16 and new list size > 1) | Replaces both lists (only allowed if new list size is more than 16) | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases entries signalled by the ext list (the resulting list size is always the same as the newly signalled legacy list size, i.e. only those entries remain after this operation) | Releases the whole list (i.e. legacy entries and extended list entries) and adds the new legacy list (the resulting list size is always the same as the newly signalled legacy list size, i.e. only those entries remain after this operation) |   We would note that this is our first assessment on how the options work - there are still questions and ambiguities with both A1 and A2, so we are not sure these are fully correct. |
| MediaTek (2) | Thanks to Intel for the table. Our understanding of the options is as follows in the table format (generally aligned with Intel’s):  Initial condition: # of entries >16   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces only the elements that was previously signalled by original list (entries that were configured by the extension list are preserved) | Replaces only the first 16 elements (irrespective of whether they were configured by the original list or the extension list) | | Extension list without legacy list | Replaces entries signalled previously by ext list (entries that were configured by the original list are preserved) | Replaces entries >16 (note: in line with Nokia’s comments above, this assumes that the UE keeps the entries in a known order—e.g., if there are 17 entries, 15 of which were originally configured by the extension list, the UE stores first the two “legacy” entries and then the 15 “ext” entries) | | Ext list with release | Release entries that were previously signalled by extension list | Release entries >16 | | Original+ext list (ext list configures new elements) | Replaces entries previously signalled by the original list and entries signalled by the ext list | Replaces both lists  (conf of ext list allowed if signalled original list is more than 16?) [MTK: We understand this would be possible in principle, but there seems to be no advantage in doing it, and it would be OK to prohibit this configuration to simplify implementations and guard against implementation divergence] | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases entries signalled by the ext list | Replaces entries <16 and releases entries >16 |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)  Note: After discussion with Intel, we understand that the entries in the first line of this table were accidentally switched in Intel’s comment—we have corrected this in the version below.   |  |  |  | | --- | --- | --- | |  | Option A1 | Option A2 | | Legacy list without extension list | Replaces the entries signalled by original list | Replaces the first 16 entries, irrespective of whether they were originally configured by the original list or the ext list; entries above 16 are preserved | | Extension list without legacy list | Replaces entries previously signalled by ext list (entries that were configured by the original list are preserved) | N/A | | Ext list with release | Release entries that were previously signalled by extension list (entries that were configured by the original list are preserved) | N/A | | Original+ext list (ext list configures new elements) | Replaces both lists (is it allowed if original list is less than 16?) [MTK: See above comment—we think the case when the original list is included with <16 entries could be excluded.] | Replaces both lists (allowed if original list is more than 16?) [MTK: See above comment—we think the case when the original list is included with <16 entries could be excluded. We assume “more than” is a typo here for “less than”, as the original list can’t be more than 16 entries.] | | Original+ext list (ext list set to release) | Replaces entries previously signalled by the original list and releases entries signalled by the ext list | Replaces entries <16 and releases entries >16 | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Rapporteur’s summary:** The tables of different cases provided in the comments helped to narrow down the questions to certain cases:

* Legacy list signalled without extension list: This case is clear for option A.1, but for option A.2, one company expressed the understanding that it would replace the entire list (including entries >16). The understanding from other companies is that in this case, option A.2 would replace the first 16 entries while preserving any entries >16. Rapporteur proposes to follow the majority interpretation.
  + Option A.1: Replaces all entries that were originally signalled with the legacy list, and preserves entries that were originally signalled with the extension list
  + Option A.2: Replaces all entries <=16, and preserves entries >16
* Extension list (set to *setup*) signalled without legacy list: The behaviour in this case seems to have consensus. Specific to option A.2, all companies expressing a view understand that if the currently stored list has <=16 entries, this case is not applicable—i.e., if we take option A.2, the network fully populates the first 16 entries of the stored list before adding any entries >16.
  + Option A.1: Replaces all entries that were originally signalled with the extension list, and preserves entries that were originally signalled with the legacy list
  + Option A.2: Replaces all entries >16, and preserves entries <=16; only valid if the stored list has at least 16 entries
* Extension list (set to *release*) signalled without legacy list: The behaviour in this case seems to have consensus, but one company expressed that there could be some uncertainty in option A.2 about which entries are released. Rapporteur interpretation is that this possibility implies that, for option A.2, the UE would need to store the entries in a deterministic order, so that the identification of the entries >16 is unambiguous.
  + Option A.1: Removes all entries that were originally signalled with the extension list, and preserves entries that were originally signalled with the legacy list
  + Option A.2: Removes all entries >16, and preserves entries <=16; only valid if the stored list has more than 16 entries
* Legacy list + extension list set to *setup*: There is agreement that this case replaces the entire stored list (from both fields), but several companies raised the question of whether this case is allowed when *candidateBeamRSList* (without suffix) is not fully populated (<16 entries). Rapporteur view is that this situation can be further discussed if there is a preference to take some form of option A.
  + Option A.1: Replaces the entire stored list
  + Option A.2: Replaces the entire stored list
  + Further discussion may be needed to determine if this is allowed when the legacy list is not fully populated
* Legacy list + extension list set to *release*: The behaviour in this case seems to have consensus, with no evident open issues.
  + Option A.1: Replaces all entries that were originally signalled with the legacy list, and releases all entries that were originally signalled with the extension list
  + Option A.2: Replaces all entries <=16, and releases all entries >16

## 4.2 Details of option B

Rapporteur understands that option B is fairly straightforward and there may not be many details that need clarification. This section is provided for any comments on the details of option B.

**Question 2.1:** Any comment on the details of option B?

|  |  |
| --- | --- |
| **Company** | **Comments** |
| ZTE | We think the release branch(i.e. release in SetupRelease) can be used in option B, and the need code of *candidateBeamRSListExt-v1610* can be ‘Need M’.  For option B ,we think the key concerns are:   1. The UE treats *candidateBeamRSList* and *candidateBeamRSListExt-v1610* as a single concatenated list;   2) The first 16 entries are configured by *candidateBeamRSList*, and the entries over 16 are configured by candidateBeamRSListExt-v1610;  3) The need codes of *candidateBeamRSList* and *candidateBeamRSListExt-v1610* all are ‘Need M’;  4) If NW wants to release the entries over 16, NW uses release branch of *candidateBeamRSListExt-v1610*.  **For** **example,**  Step1 (NW wants to add 16 entries):  NW only includes 16 entries by *candidateBeamRSList*;  Step2(NW wants to add 6 more entries, and maintains the first 16 entries unchanged):  NW only includes 6 entries by *candidateBeamRSListExt-v1610*(setup), does not include *candidateBeamRSList*( because the field is ‘Need M’, when the UE receives the second message, the UE stores the new 6 entries concatenated to the first 16 entries. After processing the second message, the UE has 22 entries. )  Step3(NW only wants to modify any entry from the first 16 entries):  Because the later 6 entries are unchanged, NW only includes the first 16 entries by *candidateBeamRSList*;  Step4(NW only wants to modify any entry from the later 6 entries):  Because the first 16 entries are unchanged, NW only includes the later 6 entries by *candidateBeamRSListExt-v1610*(setup);  Step5(NW wants to delete any 10 entries):  Because there are 12 remaining entries after 10 entries are deleted, NW should include the 12 remaining entries by *candidateBeamRSList*, and simultaneously include release command by *candidateBeamRSListExt-v1610*(release). After processing this message, the UE stores the 12 remaining entries. |
| Intel | The main “motivation” of option B is that it follows the principle we had previously agreed that non-AddMod lists are always replaced and there is no delta configuration of partial replacement or release of the elements. It is also simple as there is only one list in the UE and network and behaviour is common for the whole list. There is no release mechanism but then there was no release of the original list anyway.  I tried to summarise my understanding of option B:  Initial condition: # of entries >16   |  |  | | --- | --- | |  | Option B | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | Replaces entries >16 | | Ext list with release | N/A | | Original+ext list (ext list configures new elements) | Replaces both lists  (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | N/A |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  | | --- | --- | |  | Option B | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | N/A | | Ext list with release | N/A | | Original+ext list (ext list configures new elements) | Replaces both lists  (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | N/A | |
| Nokia, Nokia Shanghai Bell | Below is our interpretation of the option B (red text shows differences to Intel version)  Initial condition: # of entries >16   |  |  | | --- | --- | |  | Option B | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | Adds or modifies entries when list size >16 (otherwise not allowed) | | Ext list with release | N/A (i.e. the list can never be released fully and will always contain at least one entry) | | Original+ext list (ext list configures new elements) | Replaces both lists  (conf of ext list only allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | Either 1) N/A OR 2) Replaces the entire list (i.e. UE releases the entire list and replaces it with the legacy list) |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  | | --- | --- | |  | Option B | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | N/A (only allowed if legacy list size == 16) | | Ext list with release | N/A (list size < 17) | | Original+ext list (ext list configures new elements) | Replaces both lists  (conf of ext list only allowed if signalled legacy list size = 16) | | Original+ext list (ext list set to release) | Either 1) N/A OR 2) Replaces the entire list (i.e. UE releases the entire list and replaces it with the legacy list) | |
| MediaTek (2) | To ZTE’s comment, we have some confusion about taking option B but also allowing the release branch to be used. Doesn’t that bring us back to the question that options A.1 and A.2 attempt to answer—which entries exactly should be released when the extension list is set to release?  Nokia’s table above also seems to reflect some cases in which the release branch would be allowed. We understood that the guiding point of option B was to avoid using the release branch. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Rapporteur’s summary:** Of the four companies responding to this question, two indicated that option B could include some cases in which the *release* branch is used. Since the description of option B starts with “The *release* branch is not used”, rapporteur view is that this would constitute yet another option, and it is not clear if there is enough interest/support to pursue a detailed proposal in this direction. The understanding of the behaviour of the original option B seems to be consistent with the following cases:

* Legacy list signalled without extension list: Replaces the entire stored list
* Extension list (set to *setup*) signalled without legacy list: Replaces all entries >16 (only valid if the stored list has at least 16 entries)
* Extension list (set to *release*) signalled without legacy list: N/A (*release* branch is not used)
* Legacy list + extension list set to *setup*: Replaces the entire stored list (only valid if the legacy list has 16 entries)
* Legacy list + extension list set to *release*: N/A (*release* branch is not used)

## 4.3 Details of option C

Option C was introduced in informal discussion and may need some analysis to make sure that all the implications are understood. Rapporteur understanding is that this option is intended to have no UE impact (i.e., to be compatible with any UE handling of the *release* branch), since the *release* branch is only used when the original field is populated and the list fits within the original entries.

**Question 3.1:** Companies are invited to provide details of their understanding of option C**.**

|  |  |
| --- | --- |
| **Company** | **Comments** |
| MediaTek | Our understanding is described in the paragraph above; this option should be possible to specify without UE impact. As in the examples from section 3.3, we think it successfully disambiguates the handling of the extended entries. |
| ZTE | For option C, when NW only wants to modify R16 list, whether NW needs to include R15 list or not? (i.e. Do the UE and NW need to store the R15 list and R16 list separately? If yes, we think option A.1 is better than option C)  In addition, the behavior that ‘the network is required to signal the extension (*candidateBeamRSListExt-v1610*) whenever it wants the extension entries to remain unchanged in the UE’ conflicts with the need code ‘Need M’ of the field *candidateBeamRSListExt-v1610*. |
| Intel | The description of C in the introduction is not fully self contained in our view. We think option C is similar to option B in that a common list is maintained in the network and UE except that the release is still applicable for option C and it avoids the signalling scenarios which could avoid some ambiguous scenarios based on network restrictions.  I tried to summarise my understanding of option C:  Initial condition: # of entries >16   |  |  | | --- | --- | |  | Option C | | Legacy list without extension list | N/A | | Extension list without legacy list | N/A | | Ext list with release | Release entries >16 | | Original+ext list (ext list configures new elements) | Replaces both lists (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | Replaces entries <16 and releases entries >16 |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  | | --- | --- | |  | Option C | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | N/A | | Ext list with release | N/A | | Original+ext list (ext list configures new elements) | Replaces both lists (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | Replaces entries <16 and releases entries >16 | |
| Nokia, Nokia Shanghai Bell | Below is our interpretation of the option C (red text shows differences to Intel version): Like Intel, we also see similarities to option B, and also to option A in some sense.  Initial condition: # of entries >16   |  |  | | --- | --- | |  | Option C | | Legacy list without extension list | N/A (network has to explicitly release the ext list entries!) | | Extension list without legacy list | Either 1) N/A or 2) allowed, and can modify any field entry | | Ext list with release | Either 1) Release entries >16 (signalled via extension) or 2) N/A (network has to signal the original list together with "release" to avoid ambiguities) | | Original+ext list (ext list configures new elements) | Replaces both lists (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | Replace both lists (i.e. legacy list replaces entries <16 and all other entires are released) |   Initial condition: # of entries <=16 (may have been signalled as original or ext list)   |  |  | | --- | --- | |  | Option C | | Legacy list without extension list | Replaces the entire list | | Extension list without legacy list | N/A (unless legacy list size == 16, in which case new entries can be added) | | Ext list with release | N/A (legacy list is not affected by the ext list) | | Original+ext list (ext list configures new elements) | Replaces both lists (conf of ext list allowed if signalled original list is more than 16) | | Original+ext list (ext list set to release) | N/A (since the extension list was not present!) |   As with other options, there are a few unclarities also here that need more discussion. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Rapporteur’s summary:** Several companies expressed the view that option C has not been fully described. Based on the tables of cases provided by two respondents, rapporteur interpretation is that the intended behaviour is as follows:

* Legacy list signalled without extension list: Replaces the entire list (only valid if the stored list has less than 16 entries)
* Extension list (set to *setup*) signalled without legacy list: Some discussion of this case may be needed. One company understood that this case is not applicable and the network would be required to include the fields together, while one company considered that this could be a valid case to modify any list entry or when the stored list size is exactly 16 and the network intends to add new entries. (Rapporteur considers that the suggested “modify any list entry” behaviour is a bit unclear since this is not a ToAddMod list.)
* Extension list (set to *release*) signalled without legacy list: This is a critical case where there was some divergence in the responses. One company understood that this case could be used to release entries >16, while one company felt that the case either is not applicable *or* results in release of entries >16. Rapporteur view is that using this case would result in ambiguities similar to option A, so it would need to be discussed further.
* Legacy list + extension list set to *setup*: Replaces the entire list (only valid if the legacy list has 16 entries)
* Legacy list + extension list set to *release*: This case also had some divergence. For the situation that the stored list has >16 entries, there is agreement that this case replaces the entire stored list. However, when the stored list has <=16 entries, one company considered that this case replaces the entire stored list, while one company viewed the case as not applicable since the extension list was not present. Rapporteur understands that if this case is allowed, the UE behaviour is not ambiguous, but some discussion may be needed on whether the case is actually needed when the stored list has <=16 entries, as in this situation replacement of the list can also be performed by sending the legacy list without the extension list.

**Question 3.2:** Do companies understand that option C can be specified/implemented without UE impact?

|  |  |  |
| --- | --- | --- |
| **Company** | **Y/N** | **Comments** |
| MediaTek | Y |  |
| Intel | Most likely | It depends on exact definition of C. Based on our understanding of C as captured above, it is quite likely that it is aligned with any possible UE implementation. |
| Nokia, Nokia Shanghai Bell | Depends | We had slightly different interpretation on all of the options, so would like to hear the UE vendor views. What matters the most is that 3GPP makes it clear how the extension works and ensures there are no inter-operability issues. |
| Ericsson | Y | We also do not see any case (possible UE impl) that is not covered by Option C |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Rapporteur’s summary:** Of the four responses, two companies felt that option C can be specified/implemented without UE impact, while one indicated that it is “likely” but depends on the details of the option, and one company was unsure pending clarification of the details. Rapporteur interprets from this outcome that some discussion is needed to agree on the details of the option.

**Question 3.3:** Any other comment on the details of option C?

|  |  |
| --- | --- |
| **Company** | **Comments** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## 4.4 Preferred option

Companies are invited to indicate their preferred option (A/B/C).

**Question 4.1:** Which option do companies prefer among options A/B/C?

|  |  |  |
| --- | --- | --- |
| **Company** | **Preferred Option** | **Comments** |
| MediaTek | A or C | We have some concern about option B, as it effectively changes the need code of *candidateBeamRSListExt-v1610* from Need M to Need R. We are fine with the other two options. |
| ZTE | prefer A.1, then B | We prefer option A.1. But if companies think option A.1 has NBC issue, option B is ok to us.  @ MediaTek: For option B, we think it is unnecessary to change the need code of *candidateBeamRSListExt-v1610* from Need M to Need R. We can use release branch also. Please see our comments in 4.2. |
| Intel | B, C or combination of B and C (see comments) | We particularly don’t like option A1 as it seems quite complex for the UE to remember which list an entry was signalled in. We also have several open points with regard to option A1 which is not clear to us as identified in the table in Q1.2.  We support UE has a single list containing entries signalled by both lists without any additional differentiation.  Option B and C are similar in our understanding in terms of how the list is maintained by the UE. Option B does not use Release branch, while option C restricts signalling just the ext. A combination of B and C could also be considered.  A2 is not entirely clear to us (as indicated in the table on A2 behaviour) – if it is clarified, it may also be acceptable. |
| CATT | A.1 | From our point of view A.1 is reasonable and simpler. The ext list introduced in R16 was so that if it released the UE only keeps the R15 list. |
| Nokia, Nokia Shanghai Bell | B or C, but depends on exact details | We don't like option A1/A2 as they seem to have several open questions. B seemed preferable initially, but it's true that C could have some benefits in simplicity.  But no matter what, we think it's important to make this topic clear. |
| Ericsson | B/C | Conceptually, we agree with Intel comments above on B/C above, on single list containing entries signalled by both lists. And aligns to what we have agreed earlier. Option A depends on UE impl. |
| Huawei, HiSilicon | A.1 | Sorry for the late input. After checking both HiSilicon and Huawei prefers option A.1. Difficult for us to accommodate other options. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Rapporteur’s summary:** Companies’ preferences showed some divergence. With six companies responding and most expressing two preferences, the results were as follows:

A.1: 4

B: 4

C: 4

Option A.1 produced something of a hard split, with two companies indicating a preference not to go this direction (along with one company observing that option A depends on UE implementation). One company expressed concern with option B based on a perceived change of the need code of the extension field. No strong objections to option C were expressed. Rapporteur thinks further discussion is needed in phase 2, with the involvement of draft CRs to clarify the exact proposals.

## 4.5 Text for the general case

The difficulty of this example seems to suggest that we should have some general guidance in the spec for extending a list without ToAddMod, e.g. in a new section A.4.3.7 or by expanding on the existing example in section A.3.10. Any general text to be captured will depend on what approach we take to solving the specific example, but companies are invited to provide candidate text or general guidance for discussion.

**Question 5.1:** What guidance should we provide for the general case of extending lists without ToAddMod?

|  |  |
| --- | --- |
| **Company** | **Comments** |
| MediaTek | This ASN.1 idiom is definitely unfortunate, but it may not always be avoidable. Basically, we think the agreed-upon option should be documented as a general practice for the case that such an extension is necessary. A couple of principles that should be captured:   * If option A is selected, we should document that the UE needs to remember which entries were configured by the extension field, as this is a departure from the usual extension practice of considering the fields as a single combined list. * If option B is selected, we should make the extension field Need R instead of Need M. Some clarification is still needed about which entries are released, but the combination of Need M with “release when absent” behaviour is confusing. * If option C is selected, the field description should clarify the network behaviour—we shouldn’t rely only on the general guidance to specify what is expected, as it’s easy for general principles like this to be overlooked in specific cases. * In general, we don’t see that any of these solutions are really in conflict with the principle that lists without ToAddMod are always replaced when signalled. The base and extension lists are separate fields with separate need codes, and we understand the existing guideline (section A.3.10) as applying to a single field. It might be good to change section A.3.10 to state this unambiguously (in the first sentence, “the contents of the field are always replaced”).   We think clarification of how the extension works should go in a new subsection of A.4.3 (the non-critical extension section), with a reference from section A.3.10 (the “lists without ToAddMod” section). |
| Intel | I think we should differentiate the general case going forward from what is agreed for this particular case.  We have already captured the following:  Upon reception of a list not using ToAddModList and ToReleaseList structure, the UE shall delete all entries of the list currently in the UE configuration before applying the received list and shall consider each entry as newly created. This applies also to lists whose size is extended (i.e. with a second list structure in the ASN.1 comprising additional entries). This implies that Need M should not be used for fields in the entries of these lists; if used, UE will handle such fields equivalent to a Need R.  Non-AddMod lists will always be fully replaced and there is no mechanism for partial release or update of the entries. The issue for this specific field came about because we didn’t have a release for the original list and had a release for the ext. That won’t happen for the future releases.  In the future, we won’t have a scenario where we have an original list that cannot be released. And we won’t have the scenario where there is ambiguity on release of the extension list. The current text also implies the full list (org and ext) always has to be signalled. |
| Nokia, Nokia Shanghai Bell | In hindsight, it really seems like we should simply NOT use AddMod to extend plain lists: This discussion shows that brings some troubles (despite all the good intentions when we agreed to do so, when we also supported doing that).  As Intel says, to some extent following the "legacy list" rule (i.e. always re-signal everything) fits the AddMod-list paradigm as well (where everything is just "one list"). |
| MediaTek (2) | We think Intel’s statement that “we won’t have a scenario where we have an original list that cannot be released” may be a little bit optimistic. It seems difficult to exclude the possibility that a field with no release mechanism is extended in some future release, and even more difficult to guarantee that no one ever again introduces a list without a release mechanism (that could then be extended in the future). So it seems necessary to have some guidance in case this problem arises again (hopefully we can forestall it by having a clear guideline for extending lists without ToAddMod).  The existing guideline in 6.1.3 is in conflict with the use of Need M for the extension list, so we think it’s not enough to rely on this guideline to disambiguate the expected UE behaviour. |
| Ericsson | We agree with Intel that what we agree for this particular case need not mean we have to use the same approach in future. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Rapporteur’s summary:** Based on the comments, there may be consensus on the following points:

* The extension of *candidateBeamRSList* was not done in an ideal way, and something different should be used for future cases (which should be rare if they exist at all).
* The general case should not be guided by how we resolve this specific field.
* The current text on list handling (section 6.1.3) suggests that the full (extended) list must always be signalled, but does not entirely clarify how the ASN.1 should be coded to achieve this.

There is some disagreement on whether the existing text in sections 6.1.3 and A.3.10 is adequate to describe the UE behaviour when a list without ToAddMod is extended.

The general issue is less time-critical than the handling of the specific Rel-16 field. Rapporteur suggests that text for the general case be proposed in phase 2 separately from the draft CRs for *candidateBeamRSList*, with the latter given higher priority to resolve in the scope of this discussion.

For phase 2, rapporteur proposes to discuss draft CRs for options A.1, B, and C according to the interpretations given in the summary above, and at lower priority a text proposal for the general case.

# 5 Discussion (Phase 2)

## 5.1 Draft CRs

For phase 2 of the discussion, draft CRs have been provided covering options A.1, B, and C. Detailed comments on those CRs can be taken in the CR files themselves, but this discussion document is provided as a container for larger issues that may require some explanation.

### 5.1.1 Option A.1

The CR for option A.1 clarifies that the UE maintains knowledge of which list entries were configured by which field, and applies updates to the extension field only to the entries that were configured using the extension field. In the table format used in phase 1, the intended behaviour is as follows:

Initial condition: Number of stored entries <= 16

|  |  |
| --- | --- |
| Legacy list without extension list | Replaces entries originally configured by the legacy list (entries originally configured by the extension list are preserved) |
| Extension list set to *setup* without legacy list | Replaces entries originally configured by the extension list (entries originally configured by the legacy list are preserved) |
| Extension list set to *release* without legacy list | Releases entries originally configured by the extension list (entries originally configured by the legacy list are preserved) |
| Legacy list + extension list set to *setup* | Replaces entire list (no restriction on whether the legacy list must be fully populated) |
| Legacy list + extension list set to *release* | Replaces entire list with the entries in the legacy list |

Initial condition: Number of stored entries > 16

|  |  |
| --- | --- |
| Legacy list without extension list | Replaces entries originally configured by the legacy list (entries originally configured by the extension list are preserved) |
| Extension list set to *setup* without legacy list | Replaces entries originally configured by the extension list (entries originally configured by the legacy list are preserved) |
| Extension list set to *release* without legacy list | Releases entries originally configured by the extension list (entries originally configured by the legacy list are preserved) |
| Legacy list + extension list set to *setup* | Replaces entire list (note: no restriction on whether the legacy list must be fully populated) |
| Legacy list + extension list set to *release* | Replaces entire list with the entries in the legacy list |

**Question 6.1:** Do companies agree that the draft CR accurately captures the behaviour in the tables above?

|  |  |  |
| --- | --- | --- |
| **Company** | **Y/N** | **Comment** |
| MediaTek | Yes, but | The “e.g.” in the field description probably should be “i.e.”, since the text that follows describes all cases.  For this option to work, the network needs to know which entries belong to which list, which means that they have to be signalled separately in the INMs. This doesn’t seem to have INM signalling impact, but we are open to consider if anything is needed to clarify this for the network implementation. |
| Huawei, HiSilicon | Yes | We are open to look into Mediatek suggestion “i.e.” vs “e.g.”. |

**Question 6.2:** Any comment on the behaviour captured in the draft CR?

|  |  |
| --- | --- |
| **Company** | **Comment** |
| MediaTek | The only open issue here (besides the INM issue mentioned above) that we’re aware of is whether the legacy list must be fully populated to include the extension list set to *setup*. We don’t have a strong view on this (there is no ambiguity in case of sending both lists, whether or not we have such a restriction). |

### 5.1.2 Option B

The CR for option B clarifies that the *release* branch is not used, and signalling replacement entries replaces the entire list. In the table format, the intended behaviour is as follows:

Initial condition: Number of stored entries <= 16

|  |  |
| --- | --- |
| Legacy list without extension list | Replaces the entire list |
| Extension list set to *setup* without legacy list | Not allowed |
| Extension list set to *release* without legacy list | Not allowed |
| Legacy list + extension list set to *setup* | Replaces the entire list |
| Legacy list + extension list set to *release* | Not allowed |

Initial condition: Number of stored entries > 16

|  |  |
| --- | --- |
| Legacy list without extension list | Replaces the entire list |
| Extension list set to *setup* without legacy list | Replaces entries >16 |
| Extension list set to *release* without legacy list | Not allowed |
| Legacy list + extension list set to *setup* | Replaces the entire list |
| Legacy list + extension list set to *release* | Not allowed |

**Question 7.1:** Do companies agree that the draft CR accurately captures the behaviour in the tables above?

|  |  |  |
| --- | --- | --- |
| **Company** | **Y/N** | **Comment** |
| MediaTek | Almost | The draft CR does not mention the restriction that the extension list set to *setup* cannot be sent without the legacy list when the currently stored list has fewer than 16 entries. We aren’t sure that this is critical to have (there seems no ambiguity if we allow it), but it seems to be the general view and would be OK to include. |

**Question 7.2:** Any comment on the behaviour captured in the draft CR?

|  |  |
| --- | --- |
| **Company** | **Comment** |
| MediaTek | In discussion, some companies have suggested a variation of option B in which the *release* branch is still used. We understand that this would require some more detailed analysis, to make sure we have a clear understanding of which entries are released in specific scenarios. The simplest such approach might be to allow the use of *release* only when the legacy list is also included and the stored list has >16 entries, but we think this is basically the same as option C.  Considering the length of the discussion so far, we would prefer not to introduce more proposed solutions unless something is broken with the existing ones. |

### 5.1.3 Option C

The CR for option C gives a set of principles for the usage of the fields, intended to disallow the ambiguous signalling cases. In the table format, the intended behaviour is as follows:

Initial condition: Number of stored entries <= 16

|  |  |
| --- | --- |
| Legacy list without extension list | Replaces the entire list |
| Extension list set to *setup* without legacy list | Not allowed (note: flagged for potential discussion in the draft CR) |
| Extension list set to *release* without legacy list | Not allowed (note: flagged for potential discussion in the draft CR) |
| Legacy list + extension list set to *setup* | Replaces the entire list |
| Legacy list + extension list set to *release* | Not allowed (note: flagged for potential discussion in the draft CR) |

Initial condition: Number of stored entries > 16

|  |  |
| --- | --- |
| Legacy list without extension list | Not allowed |
| Extension list set to *setup* without legacy list | Not allowed (note: flagged for potential discussion in the draft CR) |
| Extension list set to *release* without legacy list | Not allowed (note: flagged for potential discussion in the draft CR) |
| Legacy list + extension list set to *setup* | Replaces the entire list |
| Legacy list + extension list set to *release* | Replaces the entire list |

**Question 8.1:** Do companies agree that the draft CR accurately captures the behaviour in the tables above?

|  |  |  |
| --- | --- | --- |
| **Company** | **Y/N** | **Comment** |
| MediaTek | Yes, but | We acknowledge that the draft CR may not be entirely clear in terms of UE implementation. This is somewhat inevitable, because it intentionally allows different UE implementations. |

**Question 8.2:** Any comment on the behaviour captured in the draft CR?

|  |  |
| --- | --- |
| **Company** | **Comment** |
| MediaTek | We are OK with this solution in principle, but as noted above, it seems that the draft CR may not yet be clear enough. We would be open to a longer but clearer explanation.  One approach might be to add a note to the field description explaining why we have the network restrictions, e.g.: “These restrictions ensure that the contents of the list are always signalled unambiguously. In particular, *candidateBeamRSListExt-v1610* set to *release* is only included in combination with *candidateBeamRSList* (without suffix), and in these cases the stored list is replaced with the contents of *candidateBeamRSList* (without suffix).” |

## 5.2 General case

As discussed above, the general case can be considered at lower priority. There was some variance of opinion on whether any text is needed to clarify how to use noncritical extension for lists without ToAddMod. One approach could be to add some commentary in section A.3.10, which already shows an extension of a list but gives no details on the mechanism. The language in section 6.1.3 may also need clarification.

The behaviour in certain cases is already clear. Rapporteur understands that there is consensus that if the original and legacy lists are signalled together, the entire list is replaced (this is aligned with the description in section 6.1.3). The ambiguous cases arise when one list is signalled without the other.

For the case that the legacy list is signalled without the extension list, rapporteur identifies three possibilities:

* + - 1. Behave as option A.1 above (the UE maintains knowledge of which entries came from which list);
      2. Replace the entire list (the absent extension list is treated as Need R);
      3. Limit the signalling so this case does not occur (the lists are always signalled together).

These are generally similar to options A.1/B/C for the specific case above. However, it was expressed by several companies in phase 1 that how we handle the specific example does not need to guide the general case.

**Question 9.1:** Do companies have a preferred option for handling the general case of list extension without ToAddMod?

|  |  |  |
| --- | --- | --- |
| **Company** | **Option** | **Comment** |
| MediaTek | 1 or 3 | We acknowledge the issues described below, but we tend to think the issue with option 2 is intractable without adding network restrictions that would bring it closer to option 3. Note that this should be a very rare case (it should occur only if, in a future release, we extend a Rel-15 field that has no release mechanism—or if we make a mistake and define a new field that has no release mechanism), so we think some special handling for it is acceptable. |

There are some specification issues to be resolved with all three options. Rapporteur identifies the following issues based on the earlier discussion:

* + - 1. The UE behaviour is somewhat divergent from what is indicated in section 6.1.3, and guidance may be needed in the field description for every instance where this extension mechanism is used. In addition, inter-node messaging may be impacted, e.g. so that the target gNB knows which entries in the UE’s configuration were configured by which field.
      2. The Need M code currently shown for the extension field in section A.3.10 may be read as confusing or inconsistent with the expected UE behaviour. If the code is changed to Need R, it needs to be clear which list entries are affected in each scenario (e.g., if the legacy list is omitted and the extension list is signalled, which entries are replaced?).
      3. A mechanism is needed to signal an empty extension list, so that the extension list can be included for cases where the intended configuration fits within the legacy list.

Comments on these options are invited.

**Question 9.2:** Any comment on how to capture the options in specification text?

|  |  |
| --- | --- |
| **Company** | **Comment** |
| MediaTek | For options 1 and 3, we think the draft CRs for the specific case can form a model for the general case. There should be some clarification in section 6.1.3.  If option 2 is pursued, we think the use of Need R should be documented, and we need to pin down what network restrictions are needed. As noted above, we tend to think this effort would evolve into something more like option 3. |

# 6 Conclusions

Based on the discussion in sections 4 and 5 above, we propose the following outcomes:

Phase 1

No proposal is offered for phase 1. The draft CRs and handling of the general extension case can be further discussed in phase 2.

Phase 2

[To be populated]

# 7 References

[1] R2-2106736: “Report of e-mail discussion [AT114-e][022][NR16] RRC II (MediaTek), MediaTek Inc., RAN2#114-e

[2] R2-2106115: “Extension of candidateBeamRSList set to ‘release’”, MediaTek Inc./Intel Corporation, RAN2#114-e