TSG-RAN Working Group 1 meeting No. 7 *TSGR1#7(99)b37*
August 30-September 3, Hanover, Germany

Agenda Item: **Ad hoc 14**

Source: **Philips**

Title: **Enhanced CPCH with status monitoring and code assignment**

Document for: **Discussion**

Summary

This document proposes some modifications which significantly improve the performance of CPCH in terms of throughput, delay and flexibility.

Background

The current proposal for CPCH [1,2,3] is illustrated in Figure 1. It includes an initial access phase with power ramping of RACH-like preamble signatures and acknowledgment via an AICH (Acquisition Indicator Channel). A maximum of one access attempt per access slot is given a positive acknowledgement. The initial access is followed by a contention resolution phase where the UE randomly selects from another set of preamble signatures with a different scrambling code. The network would normally respond (on an AICH-like channel) to the transmission received with greatest power, thus granting permission to send the packet. The acknowledgements for the contention and resolution phases can be distinguished by different channelization codes. Thus if more than one UE selected the same initial preamble, the probability of selecting the same signatures in the contention resolution phase is reduced in proportion to the number of available signatures.

In the current CPCH proposal the access phase preambles are each mapped to one of a limited number of specific downlink spreading codes (for DPCCH) and uplink scrambling codes (with associated data rates). Therefore it is likely that UE's will spend significant time waiting for a given resource to become available, particularly with high traffic loading. Not only will this lead to transmission delays, but also significant numbers of failed access attempts on the uplink. This problem is made worse because the UE selects a specific signature for the access phase, which corresponds to a specific downlink DPCCH. If this is not available, the UE cannot be allocated another DPCCH, even if one is free, and the UE will need to make another access attempt.

Proposals have already be made to provide some means of selecting the DPCCH after the contention resolution phase [4,5], and a similar principle to that described in [5] is included in our proposed modified CPCH scheme shown in Figure 2, together with some other changes.

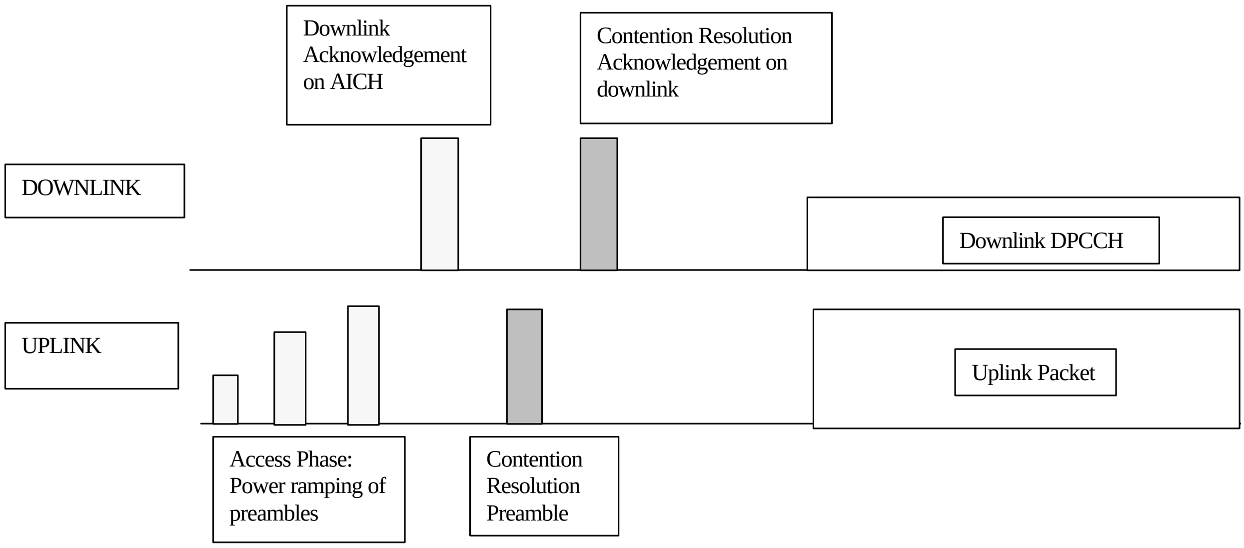

Figure 1: Basic CPCH scheme

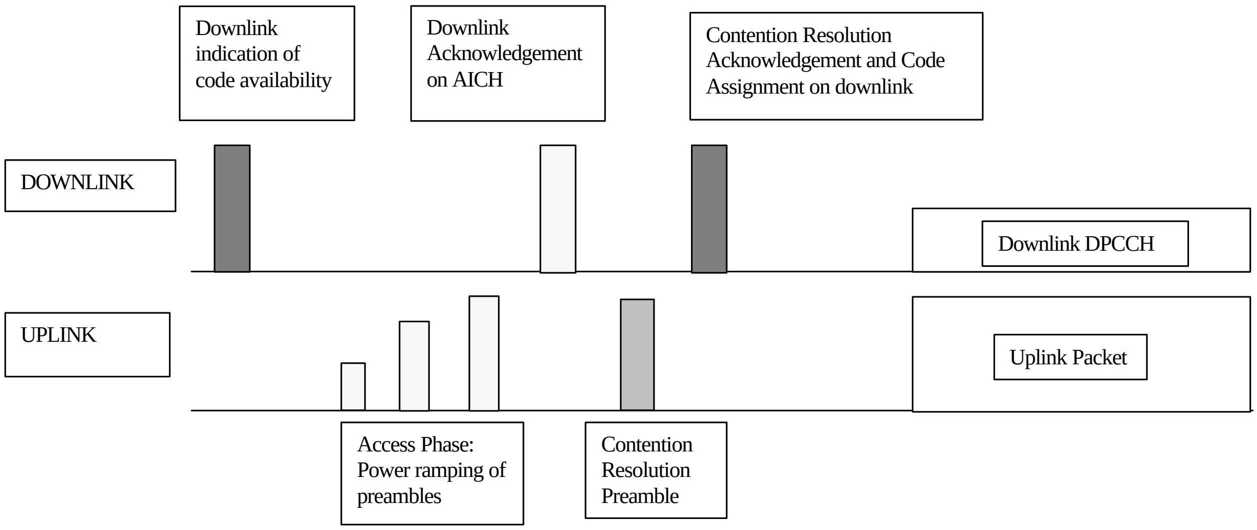

Figure 2: Modified CPCH scheme

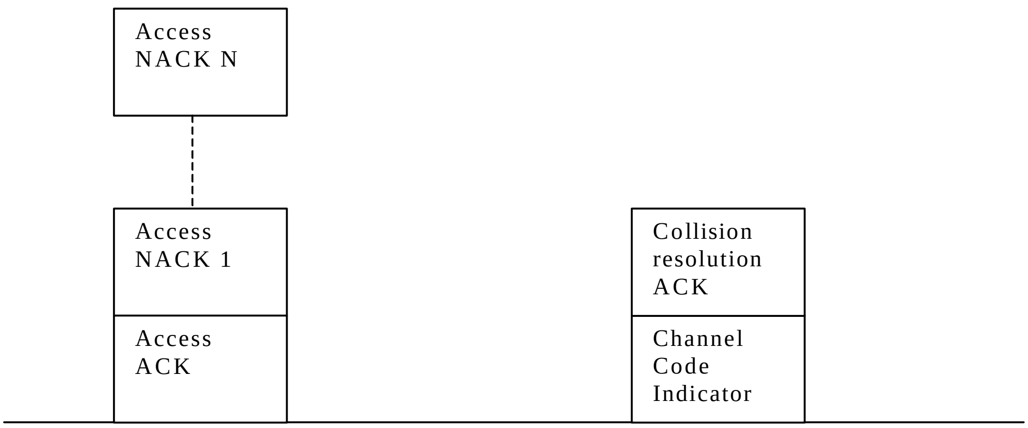

Figure 3 Information on AICH

New Proposal

The modified CPCH access procedure is outlined as follows:

1. A UE wishing to send a packet reads CPCH status information sent on the downlink which indicates the maximum bit rate available for use on the CPCH. This value would be one selected from the following (in kbps) {0, 60, 120, 240, 480, 960, 1920}. This could be represented by a 3 bit word, and since this quantity is likely to vary dynamically it should be sent at a suitable rate (e.g. every 10ms frame). A possible mechanism for signalling this information is described in another contribution [6], but others could be used. If the required bit rate is indicated to be available, then we propose that UE applies a short random back-off period (maximum duration of the order of 1 frame) before moving to step 2. This will reduce the probability of excessive collisions in the case that a number of UE's suddenly find the same resource is free. If the available bit rate is zero, or the UE determines that the available bit rate is not sufficient, then the CPCH procedure is re-started.

2. The UE chooses a bit rate (equal to or less than the maximum currently available), and then transmits one of the corresponding signatures in the access preamble. Each of the signatures available is mapped to a single uplink bit rate. Assuming there is more than one signature mapped to the required bit rate, then the UE makes a random choice. The signatures to be used for access and mappings to bit rates could be indicated on the BCH.

3. The UE starts power ramping with the selected preamble signature, until a matching acknowledgement is received on the AICH. If the acknowledgement is positive, then the UE continues with the CPCH procedure. If the acknowledgement is negative, a random back-off period is applied and the UE returns to Step 1. This ends the access phase.

4. The UE now randomly selects a signature for transmission as the contention resolution preamble (which may have a different scrambling code to the one used in the access phase). At this point there may be more than one UE in the contention resolution phase. Note that the set of signatures to be used for contention resolution could be indicated on the BCH.

5. The network acknowledges at most one of the contention resolution preambles, and at the same time indicates which DPCCH channelisation code should be used on the downlink. For each DPCCH code there is a corresponding scrambling code for the uplink. This information is sent on an AICH-like channel (see Figure 3). With a choice of 16 signatures and their inverses, it is possible to acknowledge up to 16 different preamble signatures with one code word, and at the same time send another code word indicating one of up to 16 different channelisation codes. In order to avoid the case where a signature and its inverse must be transmitted at the same time, the signatures are divided in two sets. The first set and its inverses are used for acknowledgements, and the second set and its inverses are used for code assignment (see Annex for an example). If a UE fails to receive an acknowledgement, we propose that a random back off period is applied and the UE returns to Step 1.

6. The downlink DPCCH transmission starts and the UE sends the uplink packet, possibly with a preamble phase for power control convergence, as described in the current CPCH proposal [1,2,3].

In the above description, a number of detailed points have been omitted for clarity, and some of these may need to be included in a detailed text proposal. For example some time-out periods may need to be specified to prevent repeated attempts to send a packet when the network is busy. Also the number of power ramping steps may be limited.

Advantages

The proposed changes have the following benefits:

- The UE does not attempt to transmit unless it receives an indication that suitable CPCH resource is available. This minimises interference on both uplink and downlink and saves power in the UE. It also reduces possible congestion of the CPCH, increasing throughput under high load conditions.

- It is suggested that only 3 bits per frame are required on the downlink for sending the CPCH status. This is a low overhead considering the saving in downlink signalling on the AICH under high CPCH load. An update rate of once per frame is considered a reasonable compromise between downlink overhead and delay, since the overall transmission delay will be dominated by the packet duration (typically a few frames).

- The ability to partly resolve collisions during the access phase it at least as good as the current CPCH proposal (e.g. in the case of 16 signatures and 16 CPCH channels of 60kbps).

- The allocation of uplink and downlink codes after the collision resolution phases increases the probability that a UE can obtain access to uplink resources, even if many of the CPCH channelisation codes are already in use.

- The proposed method of allocation of uplink and downlink codes improves flexibility, since any combination of uplink bit rates can be used, within the capability of the base-station or within the limit of the resources allocated to CPCH. In contrast, to achieve the same flexibility in the current CPCH scheme would require frequent updating of broadcast parameters.

- Sending the collision resolution acknowledgement at the same time as the code allocation minimises delay and allows signature assignment in a way which is compatible with RACH sub-channels.

- If only a limited set of signatures is available for collision resolution, the two step CR process described in [7] can be used. In this case the code allocation information can be split between the first and second collision resolution phases. Four signatures (and their inverses) will then give equivalent performance to a single phase with 16 signatures, including allocation of up to 16 codes.

- Even if the principle of providing explicit CPCH status on the downlink is not adopted, and some other method of monitoring CPCH activity is preferred, the proposed method for transmission of code allocation information at the same time as collision resolution on the AICH can still be used.

The performance of this scheme in terms of delay and access rate vs throughput is given in [8] where the results confirm the points discussed above.

Recommendation

It is proposed that the modifications described here are adopted as working assumptions for CPCH.

References

1. TSGR1#7(99)A72, “Proposed CPCH-related insertions into 25.213 (Resubmission)”, Adhoc 14
2. TSGR1#7(99)A73, “Proposed CPCH-related insertions into 25.214 (Resubmission)”, Adhoc 14
3. TSGR1#7(99)A74, “Proposed CPCH-related insertions into 25.211 (Resubmission)”, Adhoc 14
4. TSGR1#6(99)A24, “Channel assignment for CPCH”, InterDigital
5. TSGR1#6(99)906, “Enhanced CPCH procedure”, Samsung
6. TSGR1#7(99)b38, “Status information for CPCH”, Philips
7. TSGR1#6(99)820, “Improved performance and downlink code use for CPCH”, Philips
8. TSGR1#7(99)b34, “Performance of CPCH”, Philips

ANNEX: Example of signature assignment for contention resolution acknowledgement and code assignment.

In the current CPCH scheme, the signatures proposed for the contention resolution preamble and acknowledgement are the same as those given for the AICH (TS 25.211 5.3.3.6). Those currently described in the specification are given in Table 1.

	Preamble symbols															
Signature	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
1	**A**	**A**	**A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**	**A**	**-A**	**A**	**-A**	**A**	**A**
2	**-A**	**A**	**-A**	**-A**	**A**	**A**	**A**	**-A**	**A**	**A**	**A**	**-A**	**-A**	**A**	**-A**	**A**
3	**A**	**-A**	**A**	**A**	**A**	**-A**	**A**	**A**	**-A**	**A**	**A**	**A**	**-A**	**A**	**-A**	**A**
4	**-A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**	**-A**	**A**	**-A**	**A**	**A**	**A**
5	**A**	**-A**	**-A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**	**-A**	**A**
6	**-A**	**-A**	**A**	**-A**	**A**	**-A**	**A**	**-A**	**A**	**-A**	**-A**	**A**	**A**	**A**	**A**	**A**
7	**-A**	**A**	**A**	**A**	**-A**	**-A**	**A**	**A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**
8	**A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**	**A**	**-A**	**A**	**A**	**A**	**A**	**-A**	**A**
9	**A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**	**A**	**A**	**A**	**-A**	**-A**	**-A**	**A**	**A**	**A**
10	**-A**	**A**	**A**	**-A**	**A**	**A**	**-A**	**A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**	**A**	**A**
11	**A**	**A**	**A**	**A**	**A**	**A**	**-A**	**-A**	**A**	**A**	**-A**	**A**	**A**	**-A**	**-A**	**A**
12	**A**	**A**	**-A**	**A**	**A**	**A**	**A**	**A**	**-A**	**-A**	**-A**	**-A**	**A**	**A**	**A**	**A**
13	**A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**	**-A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**	**A**	**A**
14	**-A**	**-A**	**-A**	**A**	**-A**	**A**	**A**	**A**	**A**	**A**	**A**	**A**	**A**	**-A**	**A**	**A**
15	**-A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**
16	**-A**	**-A**	**A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**	**-A**	**A**	**-A**	**A**	**A**	**-A**	**A**

Table 1. Preamble signatures. A = 1+j.

In Table 2 we show the inverses of these preambles (indentified by n*, where n is the identifier of the original signature).

	Preamble symbols															
Signature	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
1*	**-A**	**-A**	**-A**	**A**	**A**	**A**	**-A**	**A**	**A**	**-A**	**-A**	**A**	**-A**	**A**	**-A**	**-A**
2*	**A**	**-A**	**A**	**A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**	**-A**	**A**	**A**	**-A**	**A**	**-A**
3*	**-A**	**A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**	**-A**	**-A**	**A**	**-A**	**A**	**-A**
4*	**A**	**-A**	**A**	**-A**	**A**	**A**	**A**	**A**	**A**	**-A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**
5*	**-A**	**A**	**A**	**A**	**A**	**-A**	**-A**	**A**	**A**	**A**	**A**	**-A**	**A**	**A**	**A**	**-A**
6*	**A**	**A**	**-A**	**A**	**-A**	**A**	**-A**	**A**	**-A**	**A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**
7*	**A**	**-A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**	**-A**	**A**	**A**	**A**	**A**	**A**	**A**	**-A**
8*	**-A**	**-A**	**A**	**A**	**A**	**A**	**A**	**-A**	**-A**	**A**	**-A**	**-A**	**-A**	**-A**	**A**	**-A**
9*	**-A**	**A**	**-A**	**A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**	**A**	**A**	**A**	**-A**	**-A**	**-A**
10*	**A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**	**A**	**A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**
11*	**-A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**	**A**	**-A**	**-A**	**A**	**-A**	**-A**	**A**	**A**	**-A**
12*	**-A**	**-A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**	**A**	**A**	**A**	**-A**	**-A**	**-A**	**-A**
13*	**-A**	**A**	**A**	**-A**	**-A**	**A**	**A**	**A**	**-A**	**A**	**-A**	**A**	**A**	**A**	**-A**	**-A**
14*	**A**	**A**	**A**	**-A**	**A**	**-A**	**-A**	**-A**	**-A**	**-A**	**-A**	**-A**	**-A**	**A**	**-A**	**-A**
15*	**A**	**A**	**A**	**A**	**-A**	**A**	**A**	**-A**	**A**	**-A**	**A**	**A**	**-A**	**A**	**A**	**-A**
16*	**A**	**A**	**-A**	**-A**	**A**	**-A**	**A**	**A**	**A**	**A**	**-A**	**A**	**-A**	**-A**	**A**	**-A**

Table 2. Inverse preamble signatures. A = 1+j.

Transmission of an inverse signature rather than the original signature will not affect the orthogonality properties compared to the other signatures. Hence we can use these inverse signatures to give us 32 signalling possibilities - except that we must ensure that we never try to transmit a signature and its inverse at the same time (since the sum would be zero).
Therefore we propose that in order to transmit two signatures on downlink simultaneously - one to indicate contention resolution acknowledgement, and one to indicate channel assignment - we split these signatures into two sets. The exact split is not important, as long as the first set contains 8 of the original signatures, and their inverses, and the second set contains the remaining signatures and inverses. An obvious example of two possible sets would be
{1,2,3,4,5,6,7,8,1*,2*,3*,4*,5*,6*,7*,8*} and
{9,10,11,12,13,14,15,16,9*,10*,11*,12*,13*,14*,15*,16*}.
You could then acknowledge contention resolution signatures 1-8 with signatures 1-8 and contention resolution signatures 9-16 with 1*-8*. Signatures 9-16 and 9*-16* would then be assigned to indicate particular channel assignments.

The same approach could be adopted for any similar signatures (e.g. based on Hadamard sequences).