IEEE P802.15
Wireless Personal Area Networks

Project	IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)
Title	**Annex F Changes to match 04/610r2**
Date Submitted	[January 17, 2005]
Source	[John C. Sarallo] [Appairent Technologies] [6378 Ryeworth Drive Frisco, TX 75035] Voice: [585-727-2014] Fax: [585-214-2461] E-mail: [sarallo@appairent.com]
Re:	[Ammendment Draft F8]
Abstract	[Proposed Annex F text to match material presented in 04/610r2]
Purpose	[Annex F of draft F8 was based on material presented in 04/610r0. Because there is now a r2 version of 04/610, Annex F needs updates as well. This document contains new Annex F text.]
Notice	
Release	

Annex F

(informative)

Implementation considerations

F.1 Channel time requests

F.1.1 Types of CTAs

Various types of CTAs are defined in the standard depending on the type of access method, destination and the ability to be changed by the PNC. The types of CTAs used in the standard are listed in Table F.1.

Table F.1—Types of CTAs in the standard

CTA type	SrcID/DestID	Stream Index	Access method(s)
CAP	N/A	N/A	Uses CSMA/CA, not a real CTA, but it is assigned time in the superframe.
Regular CTA	Any valid single DEVID[a]	A regular stream index	TDMA with transmit control transfer
Regular MCTA	PNCID and any valid single DEVID	MCTA stream index	TDMA with transmit control transfer. This is the same functionality as a regular CTA.
Association MCTA	UnassocID/PNCID	MCTA stream index	Slotted aloha or CSMA/CA as determined by the PHY.
Private CTA	Both IDs are any valid single DEVID	A regular stream index	Not defined by PNC, handled by DEV that has control of the CTA.
Open MCTA	BcstID/BcstID	MCTA stream index	Slotted aloha or CSMA/CA as determined by the PHY.
Open CTA	BcstID/BcstID	Asynchronous stream index	Slotted aloha or CSMA/CA as determined by PHY.

[a]A single DEVID is a DEVID that corresponds to a single physical device

In addition, CTAs are either dynamic or pseudo-static, as described in 8.4.3.1. All CTAs, with the exception of regular CTAs and private CTAs, are dynamic CTAs. All private CTAs are pseudo-static CTAs. Regular CTAs can be either dynamic or pseudo-static, depending on the CTA Type field in the Channel Time Request command when the CTA was originally requested.

A DEV is able to request the creation or a change in a CTA using the Channel Time Request command, 7.5.6.1. The PNC interprets the command based on the target ID and stream index. The various interpretations of these requests are listed in Table F.2.

Table F.2—Interpetation of Parameters in a Channel Time Request command

DestID	Stream Index	CTA Type	Description
Any DEVID	Unassigned	Regular CTA	New CTA
Any DEVID	Assigned stream index	Regular CTA	Modify or terminate existing CTA
Any DEVID	Asynchronous stream index	Regular CTA	Create or modify asynchronous CTA
Same as SrcID	Unassigned	Private CTA	New private CTA
Same as SrcID	Assigned stream index	Private CTA	Modify or terminate existing private CTA
DEVID different than SrcID	Stream index previously assigned to private CTA	Private CTA	Handover control of the private CTA to the DEV indicated in the DestID field
UnassocID, Reserved DEVID	Any	N/A	Not allowed in a request, only the PNC assigns association CTAs
BcstID	MCTA stream index	Open MCTA	Modify request for an open MCTA, PNC takes this as a suggestion
BcstID	Channel forward (CF) stream index (=FF)	Open CTA	Modify request for open CTAs, PNC takes this as a suggestion.
PNCID	MCTA stream index	Regular MCTA	Modify request for DEV to PNC CTAs, PNC takes this as a suggestion

F.1.2 Interpretation of channel time requests

The channel time request is based on the TU, which indicates the smallest unit that the DEV needs in the allocation. The TU is specified because it allows the PNC to allocate time in useful amounts and also to split up an allocation so that the latency requirements of other allocations can be met. Consider the following example:

— CTRq 1: Channel time required one-half of the superframe duration, latency required < twice the superframe duration.
— CTRq 2: Channel time required is one-tenth of the superframe duration, latency required < one quarter of the superframe duration.

To meet the needs of both requests, the PNC will have to split CTRq 1 into multiple allocations so that CTRq 2 will have at least four separate allocations spread throughout the superframe.

The PNC interprets the channel time request based on the CTRq TU, Miniumum Number of TUs, CTA Rate Factor and the CTA Rate Type fields. The CTRq TU is simply used to change the other numbers into time as well as to specify the smallest useable time for the request. Examples of how the PNC interprets these CTRq parameters are shown in Table F.3.

If a DEV sets the Desired Number of TUs field equal to the Minimum Number of TUs field, then it is indicating that it has a constant bit rate stream that needs a specific number of TUs. On the other hand, if the DEV has a variable bit rate stream, it should set the Minimum Number of TUs to the average time required and the Desired Number of TUs to help manage the latency of the high-bandwidth portions of the steam. Likewise, asynchronous traffic is also able to use the isochronous method for channel time allocation by setting the Minimum Number of TUs to zero and the Desired Number of TUs such that it specifies the entire superframe. In this last case the PNC will understand that the DEV needs as much time as possible, but not at the expense of other time-critical streams.

Table F.3—Possible allocations based on CTRq parameters

Minimum number of TUs	CTA rate factor	CTA rate type	Allocation by the PNC
11	4	0 (super-rate)	11 TUs per superframe in every superframe, spread out evenly. One possible allocation is 4 allocations, three with 4 TUs and one with 3 TUs.
11	4	1 (sub-rate)	11 TUs in a superframe, but the allocation occurs every 4th superframe and doesn't occur in the three intervening superframes. This is an average allocation of 3.75 TUs per superframe.
5	0	0 (super-rate)	Not allowed, the CTA Rate Factor and CTA Rate Type fields cannot simultaneously be zero
10	1	0 (super-rate)	10 TUs per superframe in every superframe, possibly in one allocation. However, the PNC is free to split this allocation into multiple CTAs to support the latency requirements of other CTRqs.

F.1.3 Determining CTA Rate Factor from latency requirements

In order to provide timely delivery of data, applications need periodic communication opportunities in such a way that the time between opportunities is bounded. In some cases, a single channel time allocation in each superframe is sufficient to provides this level of service. However, some applications have latency requirements that require more than one channel time allocation in each superframe. For example, if an application needs to keep its latency below 10 ms and the superframe duration is 65 ms, then more than one channel time allocation per superframe, a super-rate allocation, is required. This subclause discusses a method that can be used to determine the correct CTA Rate Factor to request for a channel time allocation based on an upper bound on the latency.

The steps involved are:

a) Determine the maximum allowed spacing between CTAs (MaxCTASpacing). This is roughly equivalent to the latency requirement.
b) Determine the superframe duration (SuperframeDuration) and time required in each superframe (TimerRequiredPerSuperframe) to support the bit rate.
c) Calculate AllocationCriteria = 2 * (SuperframeDuration - TimerRequiredPerSuperframe)
d) If the AllocationCriteria is greater than the MaxCTASpacing, then a super-rate allocation is required and the following calculations need to be made:
    1) CTA Rate Factor = (SuperframeDuration - TimeRequiredPerSuperframe) / MaxCTASpacing
    2) Round this value up to the next highest integer
    3) The TimeToRequest = TimeRequiredPerSuperframe
e) If theAllocatioCriteria is less than the MaxCTASpacing, then a sub-rate allocation is acceptable and the following calculations need to be made:
    1) CTA Rate Factor = MaxCTASpacing / (SuperframeDuration - TimeRequiredPerSuperframe)
    2) Round this value down to the next power of 2 (because sub-rates are required to be powers of 2).
    3) The TimeToRequest = (CTA Rate Factor * TimeRequiredPerSuperframe)
    4) If this value is a significant fraction of a superframe, the request may be denied and so a lower CTA Rate Factor and different TimeToRequest could be selected.
f) Use the CTA Rate Factor and TimeToRequest that was calculated above in the channel time request command.

An example of this calculation is as follows:

a) The stream requires a MaxCTASpacing of less than 2 ms.
b) The stream requires 2 ms of channel time per superframe (TimeRequiredPerSuperframe = 2 ms) and the SuperframeDuration = 10 ms
c) AllocationCriteria = 2 * (10 ms - 2 ms) = 16 ms
d) AllocationCriteria is greater than MaxCTASpacing, so a super-rate allocation is required.
e) CTA Rate Factor = (10 ms - 2 ms)/(2 ms) = 8/2 = 4
f) TimeToRequest = 2 ms
g) Request 2 ms channel time with a CTA Rate Factor = 4 and CTA Rate Type = 0 (super-rate)

Another example is as follows:

a) The stream requires a MaxCTASpacing of less than 50 ms.
b) The stream requires 1 ms of channel time per superframe (TimeRequiredPerSuperframe = 1 ms) with and the SuperframeDuration = 10 ms
c) AllocationCriteria = 2 * (10 ms - 1 ms) = 18 ms
d) AllocationCriteria is less than MaxCTASpacing, so a sub-rate allocation will suffice.
e) CTA Rate Factor = 50 ms / (10 ms - 1 ms) = 51/9 = 5.6, round down to 4
f) TimeToRequest = (4 * 1 ms) = 4 ms, which is 40% of the superframe.
g) If the CTRq TU is 1 ms, the DEV has 3 options for a channel time request based on the loading in the piconet:
    1) CTA Rate Factor = 4, CTA Rate Type = 1 (sub-rate), Minimum Number of TUs = 4
    2) CTA Rate Factor = 2, CTA Rate Type = 1 (sub-rate), Minimum Number of TUs = 2
    3) CTA Rate Factor = 1, CTA Rate Type = 0 (super-rate), Minimum Number of TUs = 1
h) On the other hand, if the CTRq TU is 2 ms, the options are slightly different:
    1) CTA Rate Factor = 4, CTA Rate Type = 1 (sub-rate), Minimum Number of TUs = 2
    2) CTA Rate Factor = 2, CTA Rate Type = 1 (sub-rate), Minimum Number of TUs = 1
    3) CTA Rate Factor = 1, CTA Rate Type = 0 (super-rate), Minimum Number of TUs = 1

Note that the last case results in a waste of the resources in the piconet because the DEV is allocated twice the time require to support the stream.

The PNC interprets the CTA Rate Factor field as follows:

a) If the CTA Rate Type = 0 (super-rate), the PNC calculates the MaxCTASpacing using:
    MaxCTASpacing = (SuperframeDuration - TimeRequested)/(CTA Rate Factor)
b) If the CTA Rate Type = 1 (sub-rate), the PNC uses the CTA Rate Factor for a sub-rate allocation. The MaxCTASpacing provided by the PNC can be calculated using:
    MaxCTASpacing = (CTA Rate Factor * SuperframeDuration) - TimeRequested

F.1.4 Calculating channel time requests based on MLME parameters

The DEV is allowed to convert from the MLME-CREATE-STREAM.request parameters to the Channel Time Request parameters in any manner it chooses. This section provides an example of one method to arrive at those parameters.

As an example, consider the following requirements for a stream:

— 8 Mb/s, constant bit rate stream.
— Buffer restricts the latency to less than 5 ms.
— High priority stream
— High reliability desired, so retry limit = 4.

— Typical frame size = 1000 octets

The MAC of the source DEV has the current parameters of the piconet as well as an estimate of the channel quality between it and the destination DEV. Assume the following parameters for the channel and piconet:

— The FER is less than 5%
— Data rates available on the link between the two DEVs is 22, 33, 44
— Superframe duration = 10 ms
— Policy is Imm-ACK.
— Overhead is approximately 30 μs per frame, an ACK duration is 30 μs and the SIFS is 10 μs.

Based on the preceeding information, the source MAC will perform the following calculations:

— Need 8 Mb/s / 8000 bits/frame = 1000 frames/second.
— With retries (and retries of the retries), and additional 5.05% of frames are required, which translates to 1051 frames/second.
— Time per frame is = 8000 bits/44 Mbs + 90 μs overhead = 271 μs/frame, so TU = 271 μs.
— In each superframe, the DEV needs to send 1051 frames/second on average. Thus, the DEV needs to send 1051 frames/second * 10 ms/superframe = 11 frames/superframe
— The total time per superframe is 11 frames/superframe * 0.271 ms = 2.99 ms, or roughly 3 ms.
— Using F.1.3, calculate AllocationCriteria = 2 * (10 ms - 3 ms) = 14 ms > 5 ms latency, so a super-rate is required.
— Using F.1.3, super-rate CTA Rate Factor = (10 ms - 3 ms) / 5 ms = 1.4, round up to 2.
— Using F.1.3, the TimeToRequest = 3 ms
— The Minimum Number of TUs to request is 3 ms / 0.271 ms = 11

Thus the DEV should use the following parameters in its request:

— CTA Rate Type = 0 (super-rate)
— CTA Rate Factor = 2
— CTRq TU = 271 μs
— Minimum Number of TUs = 11
— Desired Number of TUs = 11

The PNC upon receiving the above parameters in the DEVs request would calculate the maximum space allowed between allocated CTAs:

— Amount of time requested = 11 * 0.271 ms = 3 ms
— Using F.1.3, calculate MaxCTASpacing = (10 ms - 3 ms) / 2 = 3.5 ms

Based on this calculation the PNC can satisfy the request by providing a minimum of 2 CTAs per superframe with the maximum space between CTAs of 3.5 ms and a minimum CTA size of 1 TU (271 μs). It is not possible for the PNC to know that the actual inter-CTA spacing required was 5 ms.

As a second example, consider the example above modified such that the maximum latency was 20 ms instead of 5 ms. Because 20 ms is greater than the calculated AllocationCriteria of 14 ms, a sub-rate allocation would suffice.

— Using F.1.3, sub-rate CTA Rate Factor = 20 ms / (10 ms - 3 ms) = 20 / 7 = 2.86, round down to 2.
— The DEV has 2 options for a channel time request based on the loading in the piconet:
    1) CTA Rate Factor = 2, CTA Rate Type = 1 (sub-rate), TimeToRequest = (3 ms * 2) = 6 ms, Minimum Number of TUs = 6 ms / 0.271 ms = 22
    2) CTA Rate Factor = 1, CTA Rate Type = 0 (super-rate), TimeToRequest = 3 ms , Minimum Number of TUs = 3 ms / 0.271 ms = 11

Thus the DEV may use the following parameters in its request:

- CTA Rate Type = 1 (sub-rate)
- CTA Rate Factor = 2
- CTRq TU = 271 µs
- Minimum Number of TUs = 22
- Desired Number of TUs = 22

The PNC upon receiving the above parameters in the DEVs request can satisfy the request by providing 22 TUs every other superframe (sub-rate factor 2). The maximum inter-CTA spacing provided by the PNC using a sub-rate factor of 2 is:

- Amount of time requested = 22 * 0.271 ms = 6 ms
- Using F.1.3, MaxCTASpacing = (2 * 10 ms) - 6 ms = 14 ms.

It is not possible for the PNC to know that the actual inter-CTA spacing required was 20 ms.

F.2 Sample frames

The subclause presents sample frames that provide examples of the HCS and FCS calculations as well as the scrambler from the 2.4 GHz PHY. Two data frames are presented with the following characteristics:

- Scrambler seed = 0b00
- Data rate = 55 Mb/s
- Payload length = 20 octets
- Protocol version = 0
- Frame type = data frame
- Security off (SEC = 0)
- ACK pokicy = Imm-ACK
- Retry = 0
- More data = 1
- PNID = 100
- DestID = 5
- SrcID = 3
- MSDU number = 64 (0x40)
- Fragment number = 3
- Last fragment = 4
- Stream index = 13

Two frame payloads are provided. The first has pseudo-random data in the frame payload. The second frame payload has octets that increase in value by one, i.e., octet 0 has value 0, octet 10 has value 10.

In the figures that follow, the bits are listed lsb on the right, msb on the left with four octets per line. The lowest numbered octets are the first line and higher number octets on subsequent lines, i.e. octets 3, 2, 1 and 0 are on line 1 (octet 3 on the left, octet 0 on the right) while octets 7, 6, 5 and 4 would be on line 2 (octet 7 on the left, octet 4 on the right).

The PHY header, MAC header and HCS are common to both frames and are shown in Figure F.1.

```
# name: PHY header
# length: 16
 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0
# name: MAC header
# length: 80
 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0
 0 0 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1
 0 0 0 0 1 1 0 1 0 0 0 0 0 1 0 0
# name: HCS
# length: 16
 1 0 1 0 1 0 1 1 1 1 1 0 0 0 0 1
```

Figure F.1—PHY header, MAC header and HCS for sample frame

The frame body and FCS for pseudo-random data is shown in Figure F.2.

```
# name: Frame payload
# length: 160
 1 0 1 1 0 1 1 1 0 0 0 0 0 0 1 0 1 1 1 0 1 1 1 0 1 0 0 0 0 1 1 1
 0 1 0 1 0 0 1 1 1 1 1 0 1 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 0 1 1 0
 1 0 0 1 0 1 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 1
 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 1 1 1 0 1 0 0 0 0 1 0 0
 0 0 1 0 0 1 0 0 0 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 1 0 1 0 0 0
# name: FCS
# length: 32
 1 1 1 1 0 1 1 1 1 0 1 0 0 1 0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 0 0 1
```

Figure F.2—Frame payload and FCS for sample frame with pseudo-random data

The MAC frame body length is 160 + 32 = 192 bits.. Because the frame is encoded with 5 bits per symbol, three stuff bits, 11.4.6, are added to make it an integer number of symbols, bringing the length to 195 bits. After the stuff bits are added, the scrambler is applied, 11.4.4, to the MAC header, HCS and MAC frame body, a total of 291 bits, resulting in the bit stream, shown in Figure F.3. This is the bit stream will be modulated and sent with the PHY preamble and header over the air.

```
# name: Scrambled MAC header, HCS, frame payload and FCS
# length: 291
 1 0 0 1 0 0 0 0 0 1 1 0 0 1 0 1 1 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0
 0 0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 0 0 0 0 1 0 1
 1 0 1 0 0 0 1 0 0 1 1 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 0
 0 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1
 0 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1
 0 1 0 0 1 0 0 1 1 1 1 1 1 1 0 0 1 0 1 0 1 0 0 1 1 0 0 0 1 1 0 1
 0 1 0 1 1 0 1 0 1 0 1 0 1 1 0 0 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 1
 0 1 1 1 0 1 0 0 0 0 1 0 1 1 0 0 0 0 1 0 1 0 1 1 0 0 0 1 0 0 1 1
 1 0 1 1 0 1 1 0 1 0 1 0 0 1 0 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 0 0
 0 0 0
```

Figure F.3—MAC header, HCS and MAC frame body for sample frame with pseudo-random data after scrambler has been applied

The frame body and FCS for incremented data is shown in Figure F.4.

```
# name: Frame payload
# length: 160
 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0
 0 0 0 0 1 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0
 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 0
 0 0 0 1 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0
# name: FCS
# length: 32
 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 0
```

Figure F.4—Frame payload and FCS for sample frame with incremented data

The scrambled MAC header, HCS and MAC frame body for the incremented data frame is illustrated in Figure F.5.

```
# name: Scrambled MAC header, HCS, frame payload and FCS
# length: 291
 1 0 0 1 0 0 0 0 0 1 1 0 0 1 0 1 1 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0
 0 0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 0 0 0 0 1 0 1
 1 0 1 0 0 0 1 0 0 1 1 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 0
 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 1 1 1 1 0 1 0 1 1 0 0
 0 1 1 0 1 0 1 1 1 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 1 1 0 0 0 1 0 1
 1 1 0 1 0 1 1 0 1 1 1 0 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 0 0 1 0 0
 1 0 1 0 0 1 0 1 1 0 1 0 1 0 1 1 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 1
 0 1 0 0 0 0 1 1 0 0 0 1 0 0 0 1 0 0 1 0 1 1 1 0 1 1 1 0 1 0 1 1
 0 1 1 1 1 0 1 0 1 1 0 1 1 1 0 1 0 0 0 0 0 0 1 1 1 0 1 0 0 1 0 1
 0 0 0
```

Figure F.5—MAC header, HCS and MAC frame body for sample frame with incremented data after scrambler has been applied