LoRaWAN Remote Multicast Setup Specification v1.0.0

Copyright © 2018 LoRa Alliance, Inc. All rights reserved.

3

4

1

NOTICE OF USE AND DISCLOSURE

Copyright © LoRa Alliance, Inc. (2018). All Rights Reserved.

The information within this document is the property of the LoRa Alliance ("The Alliance") and its use and disclosure are subject to LoRa Alliance Corporate Bylaws, Intellectual Property Rights (IPR) Policy and Membership Agreements.

Elements of LoRa Alliance specifications may be subject to third party intellectual property rights, including without limitation, patent, copyright or trademark rights (such a third party may or may not be a member of LoRa Alliance). The Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

This document and the information contained herein are provided on an "AS IS" basis and THE ALLIANCE DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING BUT NOTLIMITED TO (A) ANY WARRANTY THAT THE USE OF THE INFORMATION HEREINWILL NOT INFRINGE ANY RIGHTS OF THIRD PARTIES (INCLUDING WITHOUTLIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING PATENT, COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NONINFRINGEMENT.

IN NO EVENT WILL THE ALLIANCE BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OFBUSINESS, OR FOR ANY OTHER DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTIAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR IN TORT, IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The above notice and this paragraph must be included on all copies of this document that are made.

18 WARRANTY THAT
19 PARTIES (INCLUDI
20 PATENT, COPYRIG
21 MERCHANTABILIT
22
23 IN NO EVENT WILL
24 OF USE OF DATA,
25 EXEMPLARY, INCI
26 IN TORT, IN CONN
27 ADVISED OF THE I
28
29 The above notice ar
30
31 LoRa Alliance™
31 5177 Brandin Court
33 Fremont, CA 94538
34 United States

Note: All Company, brand and product names may be trademarks that are the sole property of their respective owners.





38 39 **LoRaWAN Remote Multicast Setup** 40 **Specification** 41 42 43 **Authored by the FUOTA Working Group of the LoRa Alliance Technical Committee** 44 45 **Technical Committee Chairs:** N.SORNIN (Semtech), A.YEGIN (Actility) 46 47 48 **Working Group Chairs:** 49 J.CATALANO (Kerlink), N.SORNIN (Semtech) 50 51 52 J.CATALANO (Kerlink) 53 54 Contributors: J.CATALANO (Kerlink), J-P.COUPIGNY (STMicroelectronics), J.DELCLEF 55 56 (STMicroelectronics), A.GRIGORE (Flashnet), J.SCHLARB (Comcast), N.SORNIN (Semtech), J.STOKKING (The Things Network Foundation), A.YEGIN (Actility) 57 58

59

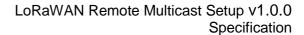
60 61

62 63

Date: September 10, 2018 Status: Final release



64	Contents	
65	1 Conventions	
66	2 Introduction	
67	3 Multicast group context definition	
68	4 Multicast Control Message Package	
69	4.1 PackageVersionReq & Ans	
70	4.2 McGroupStatusReq & Ans	
71	4.3 McGroupSetupReq & Ans	
72	4.4 McGroupDeleteReq & Ans	
73	4.5 McClassCSessionReq & Ans	
74 75	4.6 McClassBSessionReq & Ans	
75 70	5 Glossary	
76	6 Bibliography	
77 78	6.1 References 7 NOTICE OF USE AND DISCLOSURE	۱
76 79	NOTICE OF USE AND DISCLOSURE	10
70		
80	Tables	
81	Table 1: Multicast Control messages summary	7
82	Table 2: PackageVersionAns	
83	Table 3: McGroupStatusReq	
84	Table 4: McGroupStatusReq CmdMask field	8
85	Table 5: McGroupStatusAns	
86	Table 6: McGroupStatusAns Status field	
87	Table 7: McGroupSetupReq	
88	Table 8: McGroupSetupReq McGroupIDHeader field	
89	Table 9: McGroupSetupAns	
90	Table 10: McGroupSetupAns McGroupIDHeader field	
91	Table 11: McGroupDeleteReq	11
92	Table 12: McGroupDeleteReq McGroupIDHeader field	
93	Table 13: McGroupDeleteAns	
94	Table 14: McGroupDeleteAns McGroupIDHeader field	
95	Table 15: McClassCSessionReq	
96	Table 16: McClassCSessionReq McGroupIDHeader field	
97	Table 17: McClassCSessionReq SessionTimeOut field	
98	Table 18: McClassCSessionAns	13
99	Table 19: McClassCSessionAns Status&McGroupID field	
100	Table 20: McClassBSessionReq	
101	Table 21: McClassBSessionReq McGroupIDHeader field	
102	Table 22: McClassBSessionReq TimeOutPeriodicity field	
103	Table 23: McClassBSessionAns	
104	Table 24: McClassBSessionAns Status&McGroupID field	15
105		





Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.
The octet order over the air for all multi-octet fields is little endian (Least significant byte is sent first).



2 Introduction

115116117

This document defines an application layer messaging package running over LoRaWAN to perform the following operations on a fleet of end-devices:

118 119

Program a multicast distribution window into a group of end-devices

120 121 Having all end-devices of the group switch to ClassB or ClassC temporarily at the beginning of the slot

122 123 Close the distribution window and revert to normal operation (e.g. return to Class A, or change to a different periodicity in Class B)

124

All messages described in this document are transported as application layer messages. As such, all unicast messages (uplink or downlink) are encrypted by the LoRaWAN MAC layer using the end-device's AppSKey. Downlink multicast messages are encrypted using a multicast group McAppSKey common to all end-devices of the group. The setup of the group as well as means to convey the MCAppSKey are described in the document.

125 126

127 128

129

130 The "multicast control" package can be used to:

131

Remotely create a multicast group security context inside a group of end-devices

132

Report the list of multicast context existing in the end-device

133

Remotely delete a multicast security context.

134

Program a classC multicast session

Program a classB multicast session

135136

This package uses a dedicated port to separate its traffic from the rest of the applicative traffic.



3 Multicast group context definition

142 This package makes the following assumptions.

Inside a given end-device a multicast group is defined by the following parameters (the multicast group context):

- 1. A McGroupID: an integer in [0:3], the index of the multicast group. This index is used as an end-device specific shortcut to reference one of the multicast groups defined inside the end-device. An end-device supports a maximum of 4 simultaneous multicast groups, and a minimum of 0.
- 2. Multicast address: the 4 bytes network address of the multicast group, common to all end-devices of the group.
- 3. A multicast group key (McKey) from which are derived a McAppSKey and a McNwkSKey. The McKey is multicast group specific (different for every multicast group), but all end-devices of a given multicast group have the same McKey associated to this group
- 4. A frame counter.

Because the end-device can be part of up to 4 multicast groups, every multicast control command MUST first define which multicast group is concerned by the command. To minimize the protocol overhead, a 2-bit McGroupID shortcut is used instead of the full 4 bytes multicast group network address in most of the commands defined in this package. An end-device MAY support up to 4 multicast groups contexts defined simultaneously. If an end-device supports N simultaneous multicast group contexts where 1<=N<=4 then the McGroupID can only be in the range [0:N-1].

For example, if an end-device is designed to support only a single multicast group, then this group can only have McGroupID=0.



171 172

173

174 175

176 177

178179

180

181 182 183

184

4 Multicast Control Message Package

The identifier of the multicast control package is 2. The version of this package is version 1.

The following messages are sent to each end-device individually using Unicast downlink on a port specifically used for the multicast package. The default port value is 200. These messages MUST NOT be sent using multicast. If these messages are received on a multicast address the end-device MUST drop them silently.

All unicast control messages use the same format:

Command1 Command1		Command2	Command2	
	Payload		payload	

A message MAY carry more than one command. The length of each command's payload is fixed and a function of the command. Commands are executed from first to last. Each command MUST be individually acknowledged by the end-device.

The following table summarizes the list of multicast control messages

CID	Command name	Transmitted by		Short Description
		End- device	server	
0x00	PackageVersionReq		Х	Used by the AS to request the package version implemented by the end-device
0x00	PackageVersionAns	Х		Conveys the answer to PackageVersionReq
0x01	McGroupStatusReq		Х	Asks an end-device to list the multicast groups currently configured
0x01	McGoupStatusAns	Х		Conveys answer to the McGroupStatus request
0x02	McGroupSetupReq		х	Provides an end-device will all necessary information to join a multicast group
0x02	McGroupSetupAns	Х		
0x03	McGroupDeleteReq		Х	Used to delete a multicast group from an end-device
0x03	McGroupDeleteAns	Х		
0x04	McClassCSessionReq		Х	Conveys information about the next classC multicast session the end-device shall join
0x04	McClassCSessionAns	Х		
0x05	McClassBSessionReq		Х	Creates a class B multicast session
0x05	McClassBSessionAns	Х		

Table 1: Multicast Control messages summary



4.1 PackageVersionReq & Ans

187 188 189

The *PackageVersionReq* command has no payload.

The end-device answers with a *PackageVersionAns* command with the following payload.

190 191

Field	Packageldentifier	PackageVersion
Size (bytes)	1	1

192

195

197

Table 2: PackageVersionAns

193 *PackageIdentifier* uniquely identifies the package. For the "multicast control package" this 194 identifier is 2.

Package Version corresponds to the version of the package specification implemented by the

196 end-device.

4.2 McGroupStatusReq & Ans

198 199

The McGroupStatusReq command has a single byte payload.

200

Field	CmdMask
Size (bytes)	1
	01 1 0

201

Table 3: McGroupStatusReq

202 203

Where:

CmdMask Fields	RFU	ReqGroupMask
Size (bits)	4bits	4bits

204

Table 4: McGroupStatusReq CmdMask field

The ReqGroupMask bit mask defines the multicast groups whose status should be reported

205 206 207

by the end-device. ReqGroupMask[n] = 1 means that the n^{th} multicast group status SHOULD be included in the answer. ReqGroupMask[n] = 0 means that this group SHALL NOT be included in the answer.

208 209 210

The end-device responds to the McGroupStatusReq command with a McGroupStatusAns with the following payload:

211212213

Field	status	Optional list of [McGroupID+McAddr]
Size (bytes)	1	5xNbltems

214215

Table 5: McGroupStatusAns

216

The status field encodes the following information:

217

Status Fields	RFU	NbTotalGroups	AnsGroupMask
Size (bits)	1bit	3bits	4bits

218

Table 6: McGroupStatusAns Status field

219220221

222

AnsGroupMask is a bit mask describing which groups are listed in the report. If the end-device cannot report the status of the multicast groups specified by the ReqGroupMask field of the request, the end-device SHALL discard the nth last groups (starting with the highest GroupID) until the answer fits. In that case, the AnsGroupMask mask is different from the



223 ReqGroupMask. In that case the server can get the status of the groups not listed by issuing 224 a new McGroupStatusReq command with another RegGroupMask field. If all groups 225 requested can be listed, AnsGroupMask == RegGroupMask.

226 227

NbTotalGroups is the number of multicast groups currently defined in the end-device. The valid range is [0:4].

228229230

Each record consists of 5 bytes [McGroupID + McAddr].

231 McGroupID and McAddr are provided to the end-device by McGroupSetupReq.

4.3 McGroupSetupReq & Ans

232233234

This command is used to create or modify the parameters of a multicast group. The payload of the message is:

235 236

Field	McGroupIDHeader	McAddr	McKey_encrypted	minMcFCount	maxMcFCount	
Size	1	4	16	4	4	
(bytes)						
Table 7: McGroupSetupReg						

237238

Where:

239 240

McGroupIDHeader Fields	RFU	McGroupID
Size (bits)	6bits	2bits

241

Table 8: McGroupSetupReq McGroupIDHeader field

242243244

245

246

McGroupID is the multicast group ID of the multicast context. An end-device MAY support being part of several multicast group simultaneously. Therefore, all multicast related command MUST always contain an identifier (the McGroupID) of the multicast group being affected.

247248249

250

251

252

Note: The McAddr could be used as a multicast group identifier but this would add a systematic 4 bytes overhead, so a more compact McGroupID is used. Additionally, if MultiCast keys are kept in a Hardware Secure Element that can only keep a few keys, the MCU needs to indicate which key memory slot should be used. Therefore, the Multicast group ID concept is required.

253 254 An end-device implementing this package SHALL support at least one multicast group. An end-device MAY support up to a maximum of 4 simultaneous multicast contexts.

255256257

McKey_encrypted is the encrypted multicast group key from which McAppSKey and McNetSKey will be derived. The McKey_encrypted key can be decrypted using the following operation to give the multicast group's McKey.

259260

258

McKey = aes128_encrypt(McKEKey, McKey_encrypted)

261262263

The McKEKey is a <u>lifetime end-device specific</u> key used to encrypt Multicast key transported over the air (it is a Key Encryption Key), and may be either:

• Derived from a new root key (GenAppKey) provisioned in the end-device at any

264 265 time before the deployment of the end-device in the field. LoRaWAN 1.0.x end-devices SHALL use this scheme.

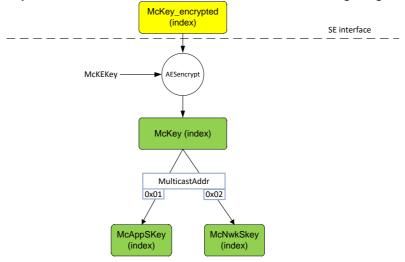


- McRootKey = aes128_encrypt(GenAppKey, 0x00 | pad₁₆)
 - McKEKey = aes128_encrypt(McRootKey, 0x00 | pad₁₆)
- Derived from the AppKey. LoRaWAN 1.1+ end-devices SHALL use this scheme.
 - McRootKey = aes128_encrypt(AppKey, 0x20 | pad₁₆)
 - McKEKey = aes128_encrypt(McRootKey, 0x00 | pad₁₆)

The McAppSKey and the McNetSKey are then derived from the group's McKey as follow:

McAppSKey = aes128_encrypt(McKey, 0x01 | McAddr | pad₁₆)
McNetSKey = aes128_encrypt(McKey, 0x02 | McAddr | pad₁₆)

The multicast key derivation scheme is summarized in the following diagram.



Note: using a Key Encryption Key to transport the multicast group McKey allows for a completely secure multicast scheme when using a hardware secure element, when the secure element does not export the McKey, McAppSKey, and McNwkSKey to the outside. It does not increase the security if a full software implementation is used in the end-device. However, for compatibility reason it is recommended to systematically use this scheme.

The *minMcFCount* field is the next frame counter value of the multicast downlink to be sent by the server for this group. This information is required in case an end-device is added to a group that already exists. The end-device MUST reject any downlink multicast frame using this group multicast address if the frame counter is < minMcFCount.

McAddr is the multicast group network address. McAddr is negotiated off-band by the application server with the network server.

maxMcFCount specifies the life time of this multicast group expressed as a maximum number of frames. The end-device will only accept a multicast downlink frame if the 32bits frame counter value minMcFCount ≤ McFCount < maxMcFCount.



302 303 304	The end-device acknowledges th McGroupSetupAns message with the	•		messag	le by sen	nding an
305	_	:ala MaC	man mal Di la a d			
			roupIDHead	er		
200	Size (by					
306	Table	e 9: McGrou	pSetupAns			
307 308	Where:					
	McGroupIDHeader Field	s RFU	IDerro	or N	/lcGroupID	7
309	Size (bits Table 10: McGrou		1 McGroupIDHe	ader field		
310 311 312 313	When set the <i>IDerror</i> bit indicates that indexed by the McGroupID requested support a single multicast group (Mcmulticast group with McGroupID = 1, the multicast gro	by the se cGroupID=	rver. For exa :0). If the se	imple, an erver tries	end-device l to create	MAY only a second
314 315	4.4 McGroupDeleteReq & Ans	5				
316 317	This message is used to delete a multi payload is:	icast group	from an end	d-device. T	he comman	ıd
318	Fi	eld McGi	oupIDHeade	\r		
319			1	,1		
320	Size (byte	es)	l			
321	Table	11: McGrou	pDeleteReq			
322 323	Where:					
	McGroupIDHeade	er Fields	RFU	McGroup	OID	
20.4		ze (bits)	6bits	2bits		
324	Table 12: McGrou	pDeleteReq	McGroupIDHe	eader field		
325 326		ield McC	ns with paylo roupIDHead			
327	Size (by: Table		ipDeleteAns			
328 329	Where:					
	McGroupIDHeader Fields	RFU	MCGroup	Undefined	McGroup	ID
330	Size (bits) Table 14: McGrou	5bits pDeleteAns	1bit McGroupIDHe	eader field	2bits	
331 332	MCGroupUndefined is set 1 if the Mother the end-device (was not created before	•	•		and is not o	defined in



4.5 McClassCSessionReq & Ans

·

This message is only used to setup a temporary classC multicast session associated with a multicast context.

The payload of the message is:

Field	McGroupIDHeader	Session	SessionTimeOut	DLFrequ	DR
		Time			
Size	1	4	1	3	1
(bytes)					

Table 15: McClassCSessionReg

341 Where:

McGroupIDHeader Fields	RFU	McGroupID
Size (bits)	6bits	2bits

Table 16: McClassCSessionReq McGroupIDHeader field

344 And where:

SessionTimeOut Fields RFU TimeOut
Size (bits) 4bits 4bits

Table 17: McClassCSessionReg SessionTimeOut field

McGroupID is the identifier of the multicast group being used.

SessionTime is the start of the Class C window, and is expressed as the time in seconds since 00:00:00, Sunday 6th of January 1980 (start of the GPS epoch) modulo 2^32. Note that this is the same format as the Time field in the beacon frame.

TimeOut encodes the maximum length in seconds of the multicast session (max time the end-device stays in classC before reverting to class A to save battery)

The maximum duration in second is 2^{TimeOut} (Example: TimeOut=8 means 256 seconds)

This is a maximum duration because the end-device's application might decide to revert to class A before the end of the session, this decision is application specific.

For example, the multicast session might be used to broadcast a firmware upgrade file. In that case the end-device might end the multicast session has soon as the full file is received without waiting for TimeOut.

DIFrequ: Encodes the frequency used for the multicast. This field is a 24 bits unsigned integer. The actual channel frequency in Hz is 100 x DIFrequ whereby values representing frequencies below 100 MHz are reserved for future use. This allows setting the frequency of a channel anywhere between 100 MHz to 1.67 GHz in 100 Hz steps.

This field has the same meaning and coding as LoRaWAN NewChannelReq MAC command 'Freq' field.

DR: index of the data rate used for the multicast. Uses the same look-up table than the one used by the LinkAdrReq MAC command of the LoRaWAN protocol.



The end-device acknowledges the reception of this message by sending a McClassCSessionAns message on the same port with the following payload:

376 377

375

Field	Status&McGroupID	(cond)TimeToStart
Size (bytes)	1	3

Table 18: McClassCSessionAns

378

379 Where:

380

Status&McGroupID	RFU	McGroupUndefined	FreqError	DRError	McGroupID
Fields					
Size (bits)	3bits	1bit	1bit	1bit	2bits

Table 19: McClassCSessionAns Status&McGroupID field

382 383

381

FreqError bit is set to 1 if the DLFrequ frequency set by the network is not usable for the end-device.

384 385 386

DRError bit is set to 1 if the classC downlink Data Rate set by the network is not defined.

387 388

McGroupUndefined is set 1 if the McGroupID specified by the command is not defined in the end-device (was not created before calling this command).

389 390 391

392

393 394

395

396

If no errors are present, the TimeToStart field encodes the number of seconds from the McClassCSessionAns uplink to the beginning of the multicast session. This allows the server to check that the end-device clock is well synchronized and that the end-device will effectively switch to classC exactly at the right moment (with second accuracy). This is possible because all uplinks are accurately time stamped by the network gateways (at least with an accuracy better than the second).

397

4.6 McClassBSessionReg & Ans

398 399

This message is only used to setup a temporary ClassB multicast session associated with a multicast context.

400 401 402

403

The payload of the message is:

Field	McGroupIDHeader	Session Time	TimeOutPeriodicity	DLFrequ	DR
Size (bytes)	1	4	1	3	1

Table 20: McClassBSessionReg

404 Where:

McGroupIDHeader Fields	RFU	McGroupID
Size (bits)	6bits	2bits

Table 21: McClassBSessionReq McGroupIDHeader field

406 407

408

405

TimeOutPeriodicity Fields	RFU	Periodicity	TimeOut
Size (bits)	1bits	3bits	4bits

Table 22: McClassBSessionReg TimeOutPeriodicity field

And where:



McGroupID is the identifier of the multicast group being used.

411 412

413 414

SessionTime is the start of the Class B window, and is expressed as the time in seconds since 00:00:00, Sunday 6th of January 1980 (start of the GPS epoch) modulo 2^32. Note that this is the same format as the Time field in the beacon frame. SessionTime MUST be an integer multiple of 128.

415 416 417

418

419

TimeOut encodes the maximum length in BeaconPeriods (128seconds) of the multicast fragmentation session (max time the end-device stays in classB before eventually reverting to class A to save battery)

The maximum duration in second is 128*2^{TimeOut} (Example: TimeOut=8 corresponds roughly 420 421 to 9.1hours).

422 423

Attention: For classB TimeOut is expressed in BeaconPeriod (128sec), whereas it is expressed in seconds for classC. This is because a classB multicast session is heavily duty-cycled and is likely to last a lot longer than a classC session.

424 425

426

427

This is a maximum duration because the end-device's application might decide to revert to class A before the end of the session, this decision is application specific.

428 429 430 Periodicity encodes the classB ping slot periodicity for the multicast group. The encoding format is the same than for the Periodicity field of the PingSlotInfoReg classB MAC command defined in LoRaWAN.

431 432

433

434

440

441

442

DIFrequ: Encodes the frequency used for the multicast. This field is a 24 bits unsigned integer. The actual channel frequency in Hz is 100 x DIFrequ whereby values representing frequencies below 100 MHz are reserved for future use. This allows setting the frequency of a channel anywhere between 100 MHz to 1.67 GHz in 100 Hz steps.

435 436

This field has the same meaning and coding as LoRaWAN NewChannelReg MAC command 437 'Freq' field.

438 439

In regions where the classB beacon is transmitted following a frequency hopping pattern. DIFrequ=0 signals the end-device to use the default classB default frequency hopping scheme. That scheme is defined in the classB section of the LoRaWAN specification. In that case, Class B downlinks use a channel which is a function of the Time field of the last beacon (see Beacon Frame content) and the multicast address McAddr.

Whereby Beacon_Time is the 32-bit Time field of the current beacon period

Class B downlink channel = $\left[McAddr + floor \left(\frac{Beacon_Time}{Beacon_period} \right) \right]$ modulo NbChannel 443

444 445

Beacon_period is the length of the beacon period (defined as 128sec in the specification)

446 447

Floor designates rounding to the immediately lower integer value McAddr is the 32-bit network address of the multicast group

448 449

NbChannel is the number of channel over which the beacon is frequency hopping

450 451 452

DR: index of the data rate used for the classB multicast. Uses the same look-up table than the one used by the LinkAdrReg MAC command of the LoRaWAN protocol.



The end-device acknowledges the reception of this message by sending a **McClassBSessionAns** message on the same port with the following payload:

456 457

455

Field	Status&McGroupID	(cond)TimeToStart
Size (bytes)	1	3

Table 23: McClassBSessionAns

458

459 Where:

460

Status&McGroupID	RFU	McGroupUndefined	FreqError	DRError	McGroupID
Fields					
Size (bits)	3bits	1bit	1bit	1bit	2bits

Table 24: McClassBSessionAns Status&McGroupID field

462 463

461

FreqError bit is set to 1 if the DLFrequ frequency set by the network is not usable for the end-device.

464 465 466

DRError bit is set to 1 if the classB downlink Data Rate set by the network is not defined.

467 468

McGroupUndefined is set 1 if the McGroupID specified by the command is not defined in the end-device (was not created before calling this command).

469 470 471

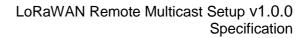
472

473

474

475

If no errors are present, the *TimeToStart* field encodes the number of seconds from the *McClassBSessionAns* uplink to the beginning of the multicast fragmentation session. This allows the server to check that the end-device clock is roughly synchronized and that it will effectively start acquiring the classB beacon at the right moment (before the beginning of the classB multicast session with some margin).





476	5 Glos	ssary
477		
478	AS	Application Server
479		
480	TBD	To Be Done
481		



482 6	Bibliography
--------------	--------------

483 6.1	References
----------------	------------

- 484 [LoRaWAN 1.0.2]: LoRaWAN™ 1.0.2 Specification, LoRa Alliance, July 2016
- 485 [LoRaWAN 1.1]: LoRaWAN™ 1.1 Specification, LoRa Alliance, October 11, 2017



487 7 NOTICE OF USE AND DISCLOSURE

- 488 Copyright © LoRa Alliance, Inc. (2018). All Rights Reserved.
- The information within this document is the property of the LoRa Alliance ("The Alliance")
- 490 and its use and disclosure are subject to LoRa Alliance Corporate Bylaws, Intellectual
- 491 Property Rights (IPR) Policy and Membership Agreements.
- 492 Elements of LoRa Alliance specifications may be subject to third party intellectual property
- 493 rights, including without limitation, patent, copyright or trademark rights (such a third party
- 494 may or may not be a member of LoRa Alliance). The Alliance is not responsible and shall
- 495 not be held responsible in any manner for identifying or failing to identify any or all such third
- 496 party intellectual property rights.
- 497 This document and the information contained herein are provided on an "AS IS" basis and
- 498 THE ALLIANCE DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING
- 499 BUT NOT LIMITED TO (A) ANY WARRANTY THAT THE USE OF THE INFORMATION
- 500 HEREIN WILL NOT INFRINGE ANY RIGHTS OF THIRD PARTIES (INCLUDING WITHOUT
- 501 LIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING PATENT
- 502 COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF
- 503 MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR
- 504 NONINFRINGEMENT.
- 505 IN NO EVENT WILL THE ALLIANCE BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF
- 506 BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OFBUSINESS, OR FOR ANY
- 507 OTHER DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTIAL, PUNITIVE OR
- 508 CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR IN TORT, IN
- 509 CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN,
- 510 EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.
- The above notice and this paragraph must be included on all copies of this document that
- 512 are made.
- 513 LoRa Alliance™
- 514 5177 Brandin Court
- 515 Fremont, CA 94538
- 516 United States
- Note: All Company, brand and product names may be trademarks that are the sole property
- 518 of their respective owners.