**文件编号：LTE\_HeNB\_MAC\_RA\_XXSJ\_V1.0**

**TD-LTE HeNB协议栈软件系统**

**MAC随机接入模块**

**详细设计说明书**

拟制：李亚楠

时间：2011-03-28

**中国科学院计算技术研究所**

**无线通信技术研究中心**

**软件组**

**LTE协议栈研发项目组**

**修改记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 文件编号 | 版本号 | 拟制人/  修改人 | 拟制/修改日期 | 更改理由 | 主要更改内容  （写要点即可） |
|  | 1.0 | 李亚楠 | 2011-03-28 | 建立 | 无 |
|  | 1.1 | 布朋生 | 2011-12-01 | 修改 | 1~5 |

修改列表：

1. 修改模块综述
2. 修改模块接口
3. 修改关键算法
4. 修改数据描述
5. 修改函数定义



本文档的程序或内容受版权法的保护，未经中科院计算所的书面许可，不得擅自泄漏、拷贝或复制本文档资料的全部或部

目录

[**TD-LTE HeNB协议栈软件系统** ………………………… …. …………………………......…………….1](#_Toc311447218)

[1 引言 ………………………………………………………………………………………………………..1](#_Toc311447219)

[1.1编写目的 1](#_Toc311447220)

[1.2背景 1](#_Toc311447221)

[1.3定义 1](#_Toc311447222)

[1.4参考资料 1](#_Toc311447223)

[2 模块描述…………………………………………………………………………………………………..2](#_Toc311447224)

[2.1 模块综述 2](#_Toc311447225)

[2.2 模块接口 4](#_Toc311447226)

[3 模块设计…………………………………………………………………………………………………..5](#_Toc311447227)

[3.1 模块结构 5](#_Toc311447228)

[3.2 关键算法 5](#_Toc311447229)

[3.2.1前导接收流程 7](#_Toc311447230)

[3.2.2 RAR准备流程 8](#_Toc311447231)

[3.2.3 Msg3接收及处理流程 9](#_Toc311447232)

[3.3 函数描述 10](#_Toc311447233)

[3.4 函数调用关系 10](#_Toc311447234)

[4 数据描述 11](#_Toc311447235)

[4.1数据结构说明 11](#_Toc311447236)

[4.2全局变量说明 12](#_Toc311447237)

[4.3数据库说明 12](#_Toc311447238)

[5 函数定义…………………………………………………………………………………………………13](#_Toc311447239)

[5.1引用函数描述 13](#_Toc311447240)

[5.2内部函数定义 13](#_Toc311447241)

[6 接口设计…………………………………………………………………………………………………14](#_Toc311447242)

[6.1用户接口 14](#_Toc311447243)

[6.2硬件接口 14](#_Toc311447244)

[6.3软件接口 15](#_Toc311447245)

[7功能测试设计…………………………………………………………………………………………….19](#_Toc311447246)

[7.1 RA功能测试 19](#_Toc311447247)

[7.2 RA功能测试 19](#_Toc311447248)

[8 其他............................................................................................................................................................19](#_Toc311447249)

**详细设计说明书**

# 1 引言

## 1.1编写目的

本文档将作为MAC随机接入模块的代码编写的依据，详细说明了本模块功能、结构、函数定义，以及与其它模块的接口。本说明书的读者为LTE HeNB端MAC随机接入模块设计、编码人员、测试人员、项目组负责人员、实验室主任及相关项目管理人员。

编写本说明书的目的在于

* 为开发人员提供依据。
* 为修改和维护本系统提供条件。
* 项目负责人将根据本文档计划和控制系统设计、开发的全过程。

## 1.2背景

软件系统名称：TD-LTE HeNB协议栈软件系统

软件开发者：无线通信技术研究中心LTE协议栈研发组

开发语言：C语言

硬件平台：暂无

软件平台：Linux

## 1.3定义

1. RA：random access
2. CE：control element
3. TC-RNTI：temporary C-RNTI
4. RAR：random access response

## 1.4参考资料

1. 《3GPP TS 36.321 v8.9.0: "E-UTRA Medium Access Control (MAC) protocol specification "》
2. 《3GPP TS 36.300 v8.12.0: " E-UTRA and E-UTRAN Overall description "》
3. 《3GPP TS 36.331 v8.11.0: " E-UTRA Radio Resource Control protocol specification "》
4. 《LTE\_HeNB协议栈软件系统MAC随机接入模块理解报告\_v1.0》
5. 《LTE\_HeNB协议栈软件系统MAC随机接入模块概要设计说明书\_v1.0》

# 2 模块描述

## 2.1 模块综述

随机接入过程的目的有三个方面：一是使UE获得上行时间同步；二是在基站与UE之间建立唯一的UE标识C-RNTI；三是为数据传输分配资源。

随机接入模块位于MAC层，与调度模块、发送模块、接收模块以及LOWMAC模块有接口关系。随机接入过程的应用场景如下：

1. 请求初始接入：当一个UE在LTE-IDLE状态时，表明网络并不精确知道UE处于哪个小区，该UE也没有任何小区范围内特有的C-RNTI。为了能够和基站进行通信，UE必须发起初始接入建立RRC连接，即从空闲状态转入连接状态并获得C-RNTI。这一步包含了初始接入和相关的信令流程。在该场景下，Msg3的内容是CCCH-SDU，即RRC CONNECTION REQUEST；（竞争模式）
2. 建立/恢复上行同步：当UE和HeNB尚未进行同步或者失去同步时，需要进行上行同步。这个过程可以由UE发起（MAC层触发）或者由网络侧发起（PDCCH order触发）。RRC-CONNECTION连接模式下，UE进行连接重建立过程时，Msg3的内容是CCCH-SDU，即RRC Connection re-establishment request，该消息中携带的是无线链路失败前UE所使用的C-RNTI；（竞争模式）
3. 小区切换接入：支持在HeNB之间切换后接入到新的小区。该过程是通过RRC连接重配置消息由基站发送给UE，UE的MAC子层进行触发。Msg3中发送的RRC 连接重配置完成消息和C-RNTI MAC控制单元，用于向HeNB确认切换完成。（竞争或非竞争）
4. UE处于RRC-CONNECTION状态，UE要发送新的上行数据，但是上行链路未同步或者没有用于调度请求的PUCCH资源时，则需要随机接入过程。（注：当UE满足上行同步并且没有其他上行资源分配给SR的情况下，UE使用RACH发送SR信息）。在RRC连接状态下，数据传输上行失步处理会由PDCCH order触发，如果UE接收到的PDCCH里面包含该UE的C-RNTI对应的PDCCH order，那么UE将初始一个随机接入过程。UE将在Msg3中发送上行传输数据和C-RNTI MAC控制单元。（竞争）
5. UE处于RRC-CONNECTION状态，UE要接收新的下行数据，但是上行链路未同步，则需要随机接入过程。UE将在Msg3中发送上行传输数据和C-RNTI MAC控制单元。（竞争或非竞争）



图1 UE主动发起的竞争随机接入过程 图2 HeNB主动发起的竞争随机接入过程

该模块能够接收并处理来自Lowmac层的前导信息和MAC接收模块的Msg3信息，能够根据MAC发送模块的rnti获取相应的RAR信息。当接收到Lowmac层的前导信息时，该模块能够根据rnti完成实体建立；当收到接收模块的Msg3时，该模块能够根据Msg3的内容进行相应实体的处理工作；当收到MAC发送模块获取的RAR消息时，该模块能够根据rnti类型来执行RAR相关信息的获取，并将RAR信息递交给MAC发送模块。

随机接入的一般流程如下：

对于UE主动发起的RA随机接入过程（都是基于竞争的），该模块主要完成以下功能：

1. 需要接收并处理来自lomac模块的Preambles，此时RA模块接收到的Preamble是一组Preambles（见API3.3.3.5），保存Preambles，建立RaRntiMsgNode实体（建立该实体的目的是为了下行调度模块对RARNODE进行调度）；
2. 为下行调度模块构造RARNODE数据，配置RAR信息中的RAPID、TA（11BITS）、UL-GRANT、TC-RNTI以及BI值；
3. 接收Msg3信息，如果Msg3中有Control Element，则解析Msg3中的Control Element，并将MAD-SDU递交给上行调度模块，同时保存一份Msg3 Control Element的拷贝，为发送Msg4做准备；
4. 构造Msg4消息，为下行调度模块提供接口；（不在RA模块完成）
5. 接收Msg4的ACK消息（不在RA模块完成），如果接收到Msg4的ACK消息看，则此次竞争随机接入成功。

对于HeNB主动发起的基于竞争的随机接入过程与情况1相同；

对于HeNB主动发起的基于非竞争的随机接入过程，该模块主要完成以下功能：

1. 需要接收并处理来自lomac模块的Preambles，保存preambles，建立RaRntiMsgNode实体；
2. 为下行调度模块构造RARNODE数据，配置RAR信息中的RAPID、TA（11BITS）、UL-GRANT、TC-RNTI以及BI值；此时认为此次非竞争随机接入过程成功；

## 2.2 模块接口



图3 RA模块与其它模块接口关系图

RA模块与其它模块的输入输出关系如图3所示。

* RA模块与lowmac模块的接口

1. RA模块向lowmac模块提供接口**rcv\_ra\_preamble()**用于接收lomac模块传送的Preambles信息；
2. RA模块向lomac每模块提供接口**send\_pdcchorder()**用于发送PDCCH ORDER来触发一个随机接入过程；

* RA模块与MAC接收模块的接口

1. RA模块向MAC接收模块提供接口**rcv\_crnti\_ce()**来解析C-RNTI MAC Control Element控制单元；
2. RA模块向MAC接收模块提供接口**rcv\_msg3()**来解析CCCH-SDU；

* RA模块与MAC发送模块的接口

1. RA模块向MAC发送模块提供接口**get\_rar()**，为发送模块准备好发送的RARNODE数据；

* RA模块与核心数据管理模块的接口

RA模块不提供接口给核心数据管理模块；核心数据管理模块向RA模块提供如下接口：

1. get\_rnti\_type()，根据UE-ID获取CRNTI-TYPE；
2. get\_available\_c\_rnti()，获取一个可用的C-RNTI或TC-RNTI；
3. create\_ue\_entity()，根据CRNTI-TYPE建议一个相应的实体，如RA-RNTI实体、C-RNTI实体以及TC-RNTI实体等；
4. get\_rnti\_entity()，根据UE-ID获取相应的实体信息；
5. delete\_ue\_entity（），根据UE-ID删除相应的实体信息；

* RA模块与调度模块的接口

RA模块不提供接口给调度模块；调度模块向随机接入模块提供接口inform\_rcv\_crnti\_ce（）。

# 3 模块设计

## 3.1 模块结构



图4 随机接入模块子模块划分图

随机接入模块子模块划分如图4所示，子模块划分如下。

* 前导处理子模块

供给lowmac模块接口，得到UE发送的preambles及RA-RNTI。负责对接收前导的识别，相应实体的建立、删除，数据结构的内存分配与释放。

* RAR准备子模块

供给MAC发送模块接口，负责RAR信息的传递。

* Msg3处理子模块

供给MAC接收模块接口，得到UE发送的Msg3。负责对接收msg3的判断及相应tc-rnti实体的处理，对收到C-RNTI CE的随机接入情况进行c-rnti实体的查询及对调度模块进行C-RNTI情况的通知。

## 3.2 关键算法

**随机接入处理详细流程如下：**

1. UE发送preamble的过程

Preambles的选择有两种方式，第一种情况是由HeNB明确指示，一是可以由RRC Connection re-establishment发送给UE这种情况适用于场景3；二是通过PDCCH order发送给UE，这种情况适用于情况4、5。第二种情况是UE自己选择。

UE 发送preamble时，会从Group A或Group B中随机选择一个Preamble进行发送，如果UE检测到发送的MSG3大于messageSizeGroupA（该参数由SIB指示），则选择Group B，如果UE检测到发送的MSG3小于messageSizeGroupA，则选择Group A。Group A中的Preamble的具体数目由SIB2中的sizeOfRA-PreamblesGroupA参数决定，preambles的具体数据由SIB2中的参数numberofRA-Preambles决定，Group B中的Preamble的具体数目由（numberofRA-Preambles-sizeOfRA-PreamblesGroupA）决定。另外，需要注意的是由UE发起的随机接入都是基于竞争的。

UE发送完Preamble后，需要在PDCCH信道上监听是否有HeNB的回复消息，UE侦听RAR的时间窗是从UE发送了preambles的子帧后+3个子帧开始，长度为Ra-ResponseWindowSize个子帧，如果在此时间内UE没有检测到HeNB给自己的回复消息，则认为此次随机接入过程失败，UE需要重新进行随机接入过程。

1. HeNB接收Preamble的过程

HeNB接收preambles时，可能是一个RA-RNTI对应一个Preamble，也可能是多个RA-RNTI对应一个Preamble，HeNB需要将接收到的Preamble以RarNode的形式保存起来。

HeNB需要在有机会发送RAR消息时，组装RAR消息，并将为调度模块组织RAR消息。RAR消息的格式如图5、图6、图7所示：

 

图5 E/T/RAPID MAC subheader 图6 E/T/R/R/BI MAC subheader



图7 MAC RAR

其中E->用于扩展或用于指示MAC头部是否还有其他域，E=1表示随后至少还有一个域，否则指示随后是RAR消息或者填充部分；

**T->表示用于指示MAC子头部包含的是随机接入RAPID还是BACKOFF指示；T=0表示这个子头部包含的是BI值，如果T=1表示这个子头部包含的是RAPID。**

R->预留比特位，置为0；

BI->backoff指示，通常是在小区过载的情况下，指示UE延后发起随机接入过程。

RAPID->随机接入preamble。

TA->时间调整值，11bits。

UL-GRANT->HeNB分配的用于传送上行MSG3的资源；

TC-RNTI->HeNB分配给UE的临时TC-RNTI。

HeNB填写RAR消息时，其中RAPID值为UE发送的preamble；TA值由物理层反馈给MAC层，通过调用LOMAC层的接口获取该值；UL-GRANT值由上行调度模块决定；TC-RNTI由RA模块内部维护一个RA-RANTI->Preambles->TC-RNTI的一张表，由RA模块分配；BI值暂时写成固定值，在本版本的实现中不考虑BI值。

1. UE接收RAR

UE物理层根据接收RAR消息的资源能够唯一确定RA-RNTI，然后UE进行上行同步，具体的上行同步过程由物理层完成。UE会保存RAR中的TC-RNTI，待随机接入成功后将该TC-RNTI变成C-RNTI；UE会按照RAR中UL-GRANT指定的资源发送MSG3；如果RAR中包含BI值，则此次随机接入不成功，UE会等待BI时间后重新进行随机接入过程。

1. UE准备、发送Msg3

UE发送Msg3后开始启动竞争消除定时器mac-ContentionResolutionTimer，UE需要在该时间内监听HeNB返回给自己的竞争冲突解决消息。为了区分不同的UE，在Msg3中会携带UE-id，在初始接入的情况下，这个ID可以是UE的S-TMSI或者随机生成一个40位的值。

1. HeNB接收Msg3消息

HeNB首先解析MAC CE，然后保存一份Msg3，向RLC层传递Msg3消息；



图8 C-RNTI MAC control element

1. HeNB发送Msg4消息

HeNB会将Msg3消息包含在Msg4中一起发送给UE，这样UE就可以根据Msg3消息来确认该Msg4是否是发送给自己的。

1. UE接收Msg4消息

如果UE在mac-ContentionResolutionTimer时间内接收到HeNB返回的ContentionResolution消息并且其中携带的UE-ID与自己在Msg3中上报给HeNB的UE-ID相符，UE就认为自己赢得了此次的随机接入机会，随机接入成功，并把在RAR消息中得到的TC-RNTI置为自己的C-RNTI。否则UE认为此次接入失败并按照上述的规则进行随机接入的重传过程。竞争冲突解决消息Msg4，也是基于HARQ的，只有赢得冲突的UE才能发送ACK值，失去冲突或无法解码Msg4的UE不发送任何反馈消息。

表1：与随机接入过程相关的参数以及物理意义

|  |  |  |  |
| --- | --- | --- | --- |
| **参数** | **所属信令** | **物理意义** | **取值范围** |
| numberofRA-Preambles | SIB2 | 用于竞争随机接入的preambles个数 | 4、8、12、16….64 |
| messageSizeGroupA | SIB2 | Group A中Preambles指示的MSG3的大小 | 56\144\208\256bits |
| sizeOfRA-PreamblesGroupA | SIB2 | 用于竞争随机接入的Group A中preambles个数 | 4、8、12、16….60 |
| Ra-ResponseWindowSize | SIB2 | UE等待RAR消息的时间窗 | 2、3、4、5、6、7、8、10个子帧 |
| mac-ContentionResolutionTimer | SIB3 | UE等待接收竞争冲突解决消息的时间窗 | 8、16、24、32、48、56、64个子帧 |

### 3.2.1前导接收流程

Lomac模块为RA模块提供RA-RNTI与Preamble，RA模块需要维护一个RA-RNTI链表，用于保存RA-RNTI、Preambles等信息，RA模块维护的RA-RNTI链表的节点中，多个RA-RNTI可以对应一个Preamble，一个RA-RNTI可以对应一个Preamble，不能重复出现RA-RNTI与Preamble完全相等的节点。该接收流程只是创建RAR节点，RAR节点中各参数的配置需要在有调度机会时才进行配置。



图9 前导接收及处理流程图

### 3.2.2 RAR准备流程

RA模块需要为MAC发送模块准备好将要发送的RAR信息，在RA模块中，需要判断该RAR节点是否获得调度机会，如果获得调度机会，则创建TC-RNTI实体、配置RAR信息、将RAR信息添加到RAR维护的调度链表中，为MAC发送模块准备好要发送的数据。如果是非竞争接入模式，则基站发送完RAR信息后，认为随机接入过程完成。



图10 RAR准备流程图

### 3.2.3 Msg3接收及处理流程

Msg3在UE初始接入或重新连接的时候包含的是CCCH-SDU，在其他场景Msg3会包含C-RNTI CE字段，两种情况的处理流程如图11、图12所示：



图11 Msg3包含CE的接收处理流程 图12 Msg3不包含CE的接收处理流程

## 3.3 函数描述

|  |  |  |
| --- | --- | --- |
| **函数名** | **功能** | **子模块：部分** |
| INT32 init\_ra() | 负责随机接入模块的初始化 | **RA管理** |
| INT32 rcv\_ra\_preamble(PhyRaInfoLst \*phy\_rainfo\_lst\_p) | 从lowmac层接收RA-RNTI、preamble\_index，负责RA-RNTI链表的管理 | **前导处理** |
| RaRntiMsgNode\*get\_rar(UINT16 rnti) | 为调度到的RA\_RNTI构造RAR消息内容 | **RAR准备** |
| INT32 rcv\_crnti\_ce(UINT16 tc\_rnti, UINT8\* msg\_p) | 从MAC 接收模块接收C-RNTI控制单元,进行C-RNTI实体的查找及TC-RNTI实体的处理 | **Msg3处理** |
| INT32 rcv\_ msg3 (UINT16 tc\_rnti, UINT16 msg\_len, UINT8 \*msg\_p) | 接收CCCH SDU，将CCCH SDU递交CE list。 | **Msg3处理** |
| INT32 cleanup\_ra() | 负责RA-RNTI实体及未处理TC-RNTI实体的删除。 | **RA管理** |

## 3.4 函数调用关系

前导接收及RAR准备：





Msg3接收及处理：





# 4 数据描述

## 4.1数据结构说明

**常量：**

#define RA\_RNTI\_TBL\_SIZE (MAX\_RA\_RNTI - MIN\_RA\_RNTI + 1)

#define MIN\_RA\_RNTI 0x0001 /\* Minimal RA RNTI \*/

#define MAX\_RA\_RNTI 0x0001 + MAX\_UE\_NUM -1 /\* Maximum RA RNTI \*/

**结构定义：**

/\* this should be defined in UL Grant management \*/

**typedef struct {**

UINT32 msg\_size;

UINT16 fixed\_riv;

UINT16 rb\_start;

UINT16 rb\_len;

UINT8 hopping\_flag;

UINT8 truncated\_mcs;

UINT8 tpc\_pusch;

UINT8 ul\_delay;

UINT8 cqi\_request;

**} RARUlGrant;**

**typedef struct {**

RARUlGrant \*ul\_grant\_p; /\* pointer to schedule result \*/

INT32 ta; /\* time advance Command \*/

UINT16 tc\_rnti; /\* temporary rnti sent to construct rar pdu \*/

**} RarInfo;**

**typedef struct {**

NodeType ln;

RarInfo \*rar\_msg\_p; /\* Pointer to RAR message of the RA-RNTI \*/

UINT8 rapid; /\* The RAPID of the RAR message,preambles \*/

**} RarNode;**

**typedef struct {**

NodeType ln;

UINT16 bytes\_size; /\* total size of all rar node \*/

UINT16 ra\_rnti; /\* ra-rnti value \*/

ListType rar\_node\_lst; /\* point to rarnode \*/

UINT8 backoff\_flag; /\* no backoff\_ind exist backoff\_ind \*/

UINT8 backoff\_ind; /\* backoff indication value \*/

**}RaRntiMsgNode;**

RA模块的数据结构组织如图所示：



## 4.2全局变量说明

存储RA-RNTI信息的指针数组：

RaRntiMsgNode \*g\_ra\_rnti\_table [RA\_RNTI\_TBL\_SIZE+1];

提供给下行调度模块的数据指针：

typedef ListType RaRntiUeLst;

RaRntiUeLst g\_ra\_rnti\_msg\_lst;

## 4.3数据库说明

无

# 5 函数定义

## 5.1引用函数描述

|  |  |  |
| --- | --- | --- |
| 模块名 | 引用函数名 | 函数描述 |
| 核心数据管理模块 | create\_ue\_entity（） | 根据给出的RNTI值创建一个普通的RNTI实体并初始化该RNTI实体，然后将这个RNTI放置在RNTI指针数组和RNTI链表之中。 |
| delete\_ue\_entity () | 删除该TC-RNTI实体并回收TC-RNTI |
| get\_available\_c\_rnti() | 从未使用RNTI值的队列数组中获取当前可用，且未用的TC-RNTI |
| get\_rnti\_entity() | 查找到RNTI值对应的RNTI实体。 |
| get\_rnti\_type() | 获得rnti类型 |
| lowmac模块 | get\_ta\_info() | 获得上行同步所需时间 |
| 上行调度模块 | inform\_rcv\_crnti\_ce（） | 对收到C-RNTI的随机接入情况，进行调度处理 |

## 5.2内部函数定义

INT32 init\_ra () 什么时候调用这个函数？在上层初始化MAC层时，或MAC接收模块接收到preambles后调用RA模块的第一个接口时；或RA模块内部自己初始化。

**函数描述**

初始化g\_ra\_rnti\_table []以及g\_ra\_rnti\_msg\_lst。

**输入**

**无**

**算法与处理流程**

**输出**

无

**返回**

成功返回0，无内存返回-1。

INT32 cleanup\_ra () 在什么时候调用这个函数？随机接入成功以后？

**函数描述**

清除ra-rnti实体。

**输入**

**无**

**算法与处理流程**

清除ra-rnti实体。

**输出**

无

**返回**

成功返回0，输入参数错误返回-2，系统参数错误返回-3。

# 6 接口设计

## 6.1用户接口

无。

## 6.2硬件接口

无

## 6.3软件接口

INT32 rcv\_ra\_preamble (UINT16 ra\_rnti, UINT8 preamble\_index)

INT32 rcv\_ra\_preamble (PhyRaInfoLst \*phy\_rainfo\_lst\_p)

**Typedef ListType** PhyRaInfoLst;

**Typedef struct {**

UINT16 ra\_rnti;

UINT8 preamble;

UINT8 [reserve](http://dict.bing.com.cn/?FORM=BNGCN#reserve)d;

UINT16 ta;

UINT8 rssi;

**} PhyRaInfoNode**

**函数描述**

该函数RA模块提供给LOMAC模块的接口，该函数对接收到PRACH上发送的preambles进行保存，记录该用户随机接入信息，并添加到RA用户队列中。

**输入**

ra\_rnti： RA\_RNTI值；

**算法与处理流程**

/\* 检验输入参数\*/

/\* 建立RarNode \*/

/\* 判断是否已存在该ra-rnti对应RaRntiMsgNode \*/

if (存在该ra-rnti对应RaRntiMsgNode) {

/\* 判断该RaRntiMsgNode中的RarNode是否存在与preamble\_index相同的rapid \*/

If（不存在rapid == preamble\_index）{

/\* 添加RAR node到已有RaRntiMsgNode\*/

} else {

printf(“this ra\_rnti and preamble\_index have existed!\n”);

}

} else {

/\* 建立RaRntiMsgNode \*/

/\* 添加RarNode到RaRntiMsgNode\*/

/\* 添加RaRntiMsgNode到g\_ra\_rnti\_table[ra\_rnti] \*/

/\* 添加RaRntiMsgNode到g\_ra\_rnti\_msg\_lst \*/

}

**输出**

无

**返回**

成功返回0，错误返回-1；

Typedef struct {

UINT16 preamble\_index; /\* 指示非竞争接入的Preamble \*/

UINT8 prach\_index; /\* 指示非竞争接入的Prach资源 \*/

} PdcchOrderInfo

#define NON\_CONENTION\_PREAMBLE 4；

Typedef struct {

UINT8 bitmask\_use; /\* 用于指示非竞争Preamble是否可用 \*/

UINT8 preamble[NON\_CONENTION\_PREAMBLE]; /\* 用于非竞争的preamble \*/

UINT16 crnti[NON\_CONENTION\_PREAMBLE]; /\* 与Preamble对应的CRNTI值 \*/

}

INT32 prepare\_pdcchorder(PdcchOrderInfo \*pdcchorder\_info\_p)

函数描述：

为发送PDCCH Order准备preamble index以及PRACH MaskIndex，供Lomac模块调用。

输入：无

算法流程：

输出：

\*pdcchorder\_info\_p：携带配置好的preamble index以及PRACH MaskIndex。

返回值：

成功返回0；失败返回-1；

INT32 get\_ta\_value(UINT16 \*ta\_value\_p)

函数描述：

输入：

算法流程：

输出：

返回值：

成功返回0；不成功返回-1；

INT32 get\_backoff\_value(UINT16 \*backoff\_value\_p)

函数描述：

输入：

算法流程：

输出：

返回值：

成功返回0； 不成功返回-1；

RaRntiMsgNode\*get\_rar (UINT16 rnti)

**函数描述**

该函数根据发送模块递交的rnti通过g\_ra\_rnti\_table[rnti]找到对应的ra-rnti链表，然后对该链表中各rar节点进行调度结果查询。若获得调度，则对该rar节点进行RAR信息完善，然后将其从原ra-rnti链表中删除并添加到scheduled\_rar链表中。进行TC-RNTI值的获取及TC-RNTI实体建立后，将scheduled\_rar链表递交给发送模块。

**输入**

rnti：RA\_RNTI值

**算法与处理流程**

/\* 检验输入rnti是否为RA-RNTI\*/

if (rnti类型为RA\_RNTI\_TYPE) {

/\* 找到该ra-rnti对应链表 \*/

/\* 获得该ra-rnti链表的RAR节点 \*/

while（RAR节点不为空）{

rapid = rar\_node\_p-> rapid；

/\* 查询调度结果 \*/

if (获得调度) {

/\*将该节点从ra-rnti链表中删除 \*/

/\* 建立scheduled\_rar链表 \*/

/\* 获得可用tc-rnti 并建立tc-rnti实体\*/

/\* 完善RAR 内容\*/

/\* 将该节点添加到scheduled\_rar链表 \*/

add\_list ((ListType \*) rar\_scheduled\_lst\_p, (NodeType \*) rar\_node\_p);

}

/\* 获得该ra-rnti链表下一RAR节点

rar\_node\_p = (RarNode \*)first\_list((ListType \*) rar\_list\_p；

} /\* while \*/

/\* 计算该ra-rnti对应的bytes\_size \*/

/\* 返回获得调度的RA-RNTI对应实体 \*/

**输出**

RA\_RNTI对应UE的RAR内容；

**返回**

成功返回0，无内存错误返回-1，输入参数错误返回-2，系统参数错误返回-3。

INT32 rcv\_msg3 (UINT16 tc\_rnti, UINT16 msg\_len, UINT8 \*msg\_p)

**函数描述**

找到该tc-rnti对应的TC-RNTI实体，将接收的msg3即CCCH SDU递交给TC-RNTI实体中的CE list。

**输入**

tc\_rnti：TC\_RNTI值；

msg\_len：CCCH消息长度

msg\_p：ccch sdu所在消息指针；

**算法与处理流程**

/\* 检查输入参数 \*/

if (get\_rnti\_type (tc\_rnti) == TC\_RNTI\_TYPE) {

/\* 找到该tc-rnti对应实体 \*/

rnti\_p = get\_rnti\_entity (tc\_rnti)；

/\* 将CCCH SDU放入CE LIST \*/

}

**输出**

无

**返回**

成功返回0，无内存错误返回-1，输入参数错误返回-2，系统参数错误返回-3。

INT32 rcv\_crnti\_ce (UINT16 tc\_rnti, UINT8\* msg\_p)

**函数描述**

对具有C-RNTI终端发来的c-rnti控制单元，首先查找tc-rnti对应的TC-RNTI实体，根据接收的CRNTI控制单元，进行对应C-RNTI实体的查询。若查找到对应的C-RNTI实体则删除TC\_RNTI实体。若未找到C-RNTI实体则向上层进行报错处理。

**输入**

**tc\_rnti**：TC\_RNTI值；

**msg\_p**：C\_RNTI控制单元消息指针；

**算法与处理流程**

/\* 检验输入参数 \*/

if (rnti类型为TC\_RNTI\_TYPE) {

/\*检测是否存在该tc-rnti对应实体 \*/

/\*获取Msg3中的C-RNTI \*/

/\* 寻找该c-rnti对应实体 \*/

if (get\_rnti\_entity (c\_rnti)) {

inform\_rcv\_crnti\_ce(c\_rnti); /\* 通知调度模块，收到C-RNTI \*/

delete\_ue\_entity (tc\_rnti); /\* 回收tc-rnti，且删除该tc-rnti实体 \*/

} else {

/\* 向上层报错 \*/

}

}

**输出**

无

**返回**

成功返回0，无内存错误返回-1，输入参数错误返回-2，系统参数错误返回-3。

# 7功能测试设计

## 7.1 RA功能测试

**功能描述：**

收到PRACH上发送的前导后，记录该用户随机接入信息，并添加到RA用户队列中。根据发送模块递交的rnti通过g\_ra\_rnti\_table[rnti]找到对应的ra-rnti链表，对获得调度的rar节点进行RAR信息完善，然后将其从原ra-rnti链表中删除并添加到scheduled\_rar链表中。获取TC-RNTI值并建立TC-RNTI实体后，将scheduled\_rar链表递交给发送模块。从接收模块收到Msg3后，找到该tc-rnti对应的TC-RNTI实体，将接收的msg3即CCCH SDU递交给TC-RNTI实体中的CE list。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试12 | 测试说明： | 测试从前导接收到接收Msg3为CCCH SDU的完整随机接入过程。 |
| 测试输入： | 正确参数。 |
| 预期输出： | 正常执行RA流程。 |

## 7.2 RA功能测试

**功能描述：**

收到PRACH上发送的前导后，记录该用户随机接入信息，并添加到RA用户队列中。根据发送模块递交的rnti通过g\_ra\_rnti\_table[rnti]找到对应的ra-rnti链表，对获得调度的rar节点进行RAR信息完善，然后将其从原ra-rnti链表中删除并添加到scheduled\_rar链表中。获取TC-RNTI值并建立TC-RNTI实体后，将scheduled\_rar链表递交给发送模块。从接收模块收到c-rnti ce后，找到该c-rnti对应的C-RNTI实体，成功报告给调度模块并成功删除TC-RNTI实体。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试13 | 测试说明： | 测试从前导接收到接收Msg3为c-rnti ce的完整随机接入过程。 |
| 测试输入： | 正确参数。 |
| 预期输出： | 正常执行RA流程。 |

# 8 其他

暂未考虑内容：

1. 退避值的获取；