**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 |

修改列表：

无。



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分。

[1 引言 2](#_Toc291495108)

[1.1编写目的 2](#_Toc291495109)

[1.2背景 2](#_Toc291495110)

[1.3定义 2](#_Toc291495111)

[1.4参考资料 2](#_Toc291495112)

[2 模块描述 3](#_Toc291495113)

[2.1 模块综述 3](#_Toc291495114)

[2.2 模块接口 3](#_Toc291495115)

[3 模块设计 4](#_Toc291495116)

[3.1 模块结构 4](#_Toc291495117)

[3.2 关键算法 5](#_Toc291495118)

[3.2.1前导接收流程 5](#_Toc291495119)

[3.2.2 RAR准备流程 5](#_Toc291495120)

[3.2.3 Msg3接收及处理流程 6](#_Toc291495121)

[3.3 函数描述 6](#_Toc291495122)

[3.4 函数调用关系 7](#_Toc291495123)

[前导接收及RAR准备： 7](#_Toc291495124)

[Msg3接收及处理： 7](#_Toc291495125)

[4 数据描述 8](#_Toc291495126)

[4.1数据结构说明 8](#_Toc291495127)

[4.2全局变量说明 9](#_Toc291495128)

[4.3数据库说明 10](#_Toc291495129)

[5 函数定义 11](#_Toc291495130)

[5.1引用函数描述 11](#_Toc291495131)

[5.2内部函数定义 11](#_Toc291495132)

[INT32 init\_ra () 12](#_Toc291495133)

[INT32 cleanup\_ra () 12](#_Toc291495134)

[6 接口设计 12](#_Toc291495135)

[6.1用户接口 12](#_Toc291495136)

[6.2硬件接口 12](#_Toc291495137)

[6.3软件接口 13](#_Toc291495138)

[INT32 rcv\_ra\_preamble (UINT16 ra\_rnti, UINT8 preamble\_index) 13](#_Toc291495139)

[RaRntiMsgNode\*get\_rar (UINT16 rnti) 15](#_Toc291495140)

[INT32 rcv\_msg3 (UINT16 tc\_rnti, UINT16 msg\_len, UINT8 \*msg\_p) 16](#_Toc291495141)

[INT32 rcv\_crnti\_ce (UINT16 tc\_rnti, UINT8\* msg\_p) 17](#_Toc291495142)

[7功能测试设计 18](#_Toc291495143)

[7.2 RA功能测试 18](#_Toc291495144)

[7.3 RA功能测试 18](#_Toc291495145)

[8 其他 18](#_Toc291495146)

**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

MAC随机接入模块处于MAC层。

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发送模块的获取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。

## 2.2 模块接口



图1随机接入模块与其它模块接口关系图

随机接入模块与其它模块的输入输出关系如图1所示。

* 随机接入模块与lowmac模块的接口

随机接入模块向lowmac模块提供接口rcv\_ra\_preamble（）；

* 随机接入模块与MAC接收模块的接口

随机接入模块向MAC接收模块提供接口rcv\_msg3（）、rcv\_crnti\_ce（）；

* 随机接入模块与MAC发送模块的接口

随机接入模块向MAC发送模块提供接口get\_rar（）；

* 随机接入模块与核心数据管理模块的接口

随机接入模块向核心数据管理模块不提供接口；核心数据管理模块向随机接入模块提供接口create\_ue\_entity（）、get\_available\_c\_rnti（）、delete\_ue\_entity（）、get\_rnti\_type（）、get\_rnti\_entity（）；

* 随机接入模块与调度模块的接口

随机接入模块向调度模块不提供接口；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2 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2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发送的前导及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

### 3.2.1前导接收流程



图3 前导接收及处理流程图

### 3.2.2 RAR准备流程



图4 RAR准备流程图

### 3.2.3 Msg3接收及处理流程



图5 Msg3接收处理流程图

## 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(UINT16 ra\_rnti, UINT8 preamble\_index) | 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 {

UINT16 fixed\_riv;

UINT8 hopping\_flag;

UINT8 truncated\_mcs;

UINT8 tpc\_pusch;

UINT8 ul\_delay;

UINT8 cqi\_request;

} RARUlGrant;

typedef struct {

INT32 ta;

RARUlGrant \*ul\_grant\_p;

UINT16 tc\_rnti; /\* tc-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 \*/

} RarNode;

typedef struct { NodeType ln;

UINT16 bytes\_size; /\* total size of all rar node \*/ UINT16 ra\_rnti; /\* ra-rnti value \*/

ListType rar\_node\_lst; UINT8 backoff\_flag; /\* 0：no backoff\_ind；1：exist backoff\_ind \*/ UINT8 backoff\_ind; /\* backoff indication value \*/

}RaRntiMsgNode;



图6 g\_ra\_rnti\_msg\_lst结构图



图7 g\_ra\_rnti\_table[]结构图

## 4.2全局变量说明

/\* RA-RNTI message list table \*/

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

/\* 供给调度模块的ra-rnti链表 \*/

ListType 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 | 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 |
| 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 ()

**函数描述**

初始化g\_ra\_rnti\_table []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 ()

**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)

**函数描述**

接收到PRACH上发送的前导，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；

preamble\_index：接收到的前导序列码；

**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，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。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试1 | 测试说明： | 测试参数配置函数错误处理 |
| 测试输入： | 输入参数不在协议规定范围 |
| 预期输出： | 能够检查出参数配置错误 |
| 测试2 | 测试说明： | 输入已有的ra\_rnti，则该前导对应rar节点可成功添加到已有ra\_rnti链表中以及供给调度模块的g\_ra\_rnti\_msg\_lst；输入新ra\_rnti，则新建ra\_rnti链表并将该前导对应rar节点添加到新建ra\_rnti链表中，以及将该节点添加到供给调度模块的g\_ra\_rnti\_msg\_lst中。 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 添加了新rar节点的ra\_rnti链表及g\_ra\_rnti\_msg\_lst。 |

### 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。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试3 | 测试说明： | 测试参数配置函数错误处理 |
| 测试输入： | 输入参数不在协议规定范围 |
| 预期输出： | 能够检查出参数配置错误 |
| 测试4 | 测试说明： | 获得调度结果的RAR节点，从ra\_rnti链表及g\_ra\_rnti\_msg\_lst中成功删除并成功添加到scheduled\_rar 链表中。 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 显示已经删除了获得调度节点的ra\_rnti链表及g\_ra\_rnti\_msg\_lst，显示添加了获得调度结果节点的scheduled\_rar 链表内容 |
| 测试5 | 测试说明： | 若输入ra\_rnti对应的rar节点都未获得调度结果则返回空指针。 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 调度链表中该没有该rar节点。该节点未从ra\_rnti链表及g\_ra\_rnti\_msg\_lst链表中删除。 |

### 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。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试6 | 测试说明： | 测试参数配置函数错误处理 |
| 测试输入： | 输入参数不在协议规定范围 |
| 预期输出： | 能够检查出参数配置错误 |
| 测试7 | 测试说明： | 不能找到TC-RNTI对应实体，则向上层报错 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 报错处理 |
| 测试8 | 测试说明： | 成功接收ccch sdu，则将ccch sdu放入TC\_RNTI实体的dl\_ce\_list。 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 显示TC\_RNTI实体的dl\_ce\_list。 |

### 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。

**测试设计**

|  |  |  |
| --- | --- | --- |
| **测试编号** | **测试内容** | |
| 测试9 | 测试说明： | 测试参数配置函数错误处理 |
| 测试输入： | 输入指针为空，参数超出协议规定范围 |
| 预期输出： | 能够检查出参数配置错误 |
| 测试10 | 测试说明： | 不能找到TC-RNTI对应实体，则向上层报错 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 配置成功 |
| 测试11 | 测试说明： | 能找到C-RNTI对应实体，成功报告给调度模块并成功删除TC-RNTI实体 |
| 测试输入： | 正常参数配置 |
| 预期输出： | 删除后无法找到该TC-RNTI实体 |

# 7功能测试设计

## 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。 |

## 7.3 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；