**文件编号：LTE-HeNB\_(MAC)\_(SCHEDULED)\_DYCS\_V0.2**

**TD-LTE HeNB协议栈软件系统**

**MAC调度模块**

**分析报告**

拟制：杜红艳

时间：2012.2.10

**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） |
|  | 0.2 |  | 2010-2-12 | 建立 |  |
|  |  |  |  |  |  |
| 注1：每次更改归档文件（指归档到组内及研究室的文件）时，需填写此表。  注2：文件第一次归档时，“更改理由”、“主要更改内容”栏写“无”。 | | | | | |

修改列表：



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。

**目录**

[1 引言 1](#_Toc316637865)

[1.1 编写目的 1](#_Toc316637866)

[1.2 背景 1](#_Toc316637867)

[1.3 定义 1](#_Toc316637868)

[1.4 参考资料 1](#_Toc316637869)

[2 模块概述 2](#_Toc316637870)

[2.1 特性列表 2](#_Toc316637871)

[2.2 模块功能描述 2](#_Toc316637872)

[2.3 与其他模块的关系 2](#_Toc316637873)

[3 Mac下行调度相关过程 3](#_Toc316637874)

[3.1 MAC下行处理流程 3](#_Toc316637875)

[3.2 上行相关调度 (上行授权/上行数据的ACK响应) 3](#_Toc316637876)

[3.2.1 DCI0 调度(UE发送上行数据的授权) 3](#_Toc316637877)

[3.2.2 上行数据的ACK 4](#_Toc316637878)

[3.3 下行数据调度 5](#_Toc316637879)

[3.3.1 BCH 调度 5](#_Toc316637880)

[3.3.2 PCH 调度 5](#_Toc316637881)

[3.3.3 MCH调度 6](#_Toc316637882)

[3.3.4 DL-SCH调度 6](#_Toc316637883)

[3.4 处理调度结果 9](#_Toc316637884)

[4 其他说明 9](#_Toc316637885)

[4.1 L1api 下行发送的控制信息数据结构 9](#_Toc316637886)

[4.1.1 PDU 节点 9](#_Toc316637887)

[4.1.2 DCI DL PDU 9](#_Toc316637888)

[4.1.3 BCH PDU 10](#_Toc316637889)

[4.1.4 MCH PDU 11](#_Toc316637890)

[4.1.5 DLSCH PDU 11](#_Toc316637891)

[4.1.6 PCH PDU 12](#_Toc316637892)

[4.2 L1api 下行发送数据对应数据结构 13](#_Toc316637893)

[4.2.1 Tx pdu 13](#_Toc316637894)

[4.2.2 Tx pdu node 13](#_Toc316637895)

[4.3 目前调度的实现与问题 13](#_Toc316637896)

**模块概要设计说明书**

# 引言

## 编写目的

本文档将作为MAC发送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发送模块设计、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。

## 背景

软件系统名称：TD-LTE HeNB协议栈软件系统

软件开发者：中科院计算所无线通信技术研究中心-LTE协议栈研发项目组

开发语言：C语言

硬件平台：暂无

软件平台：Linux

## 定义

1. **MAC Medium Access Control**

## 参考资料

1. 3GPP TR 36.213: "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer Procedures".
2. 3GPP TR 36.321: "Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol ".
3. PICOCHIP LTE eNB L1 API Definition.

# 模块概述

## 特性列表

本文档旨在理清mac下行调度处理的流程，为mac调度优化做好准备。

## 模块功能描述

结合l1api分析了mac下行调度处理过程；

给出l1api 下行数据发送 对应控制、数据信息数据结构；

对目前mac调度进行简要分析

## 与其他模块的关系

# Mac下行调度过程

## MAC下行处理流程

Mac 下行处理流程

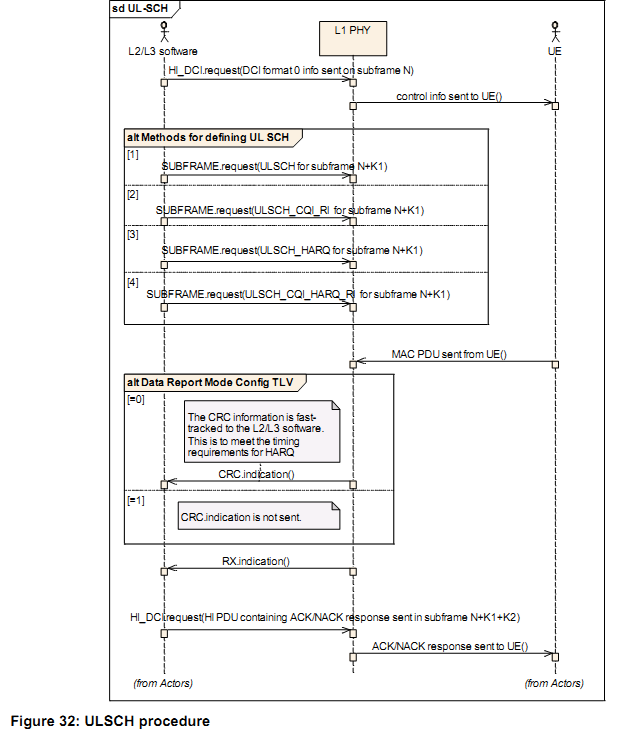
调度次序

* 寻呼(Paging)消息(PCH数据，P-RNTI标志)
* SI消息(BCH数据，SI-RNTI标志)
* 随机接入竞争性解决数据( TC\_RNTI标志)
* 随机接入响应 (RA-RNTI标志) （于上行相关调度中处理）
* 先重传后新传数据(DL-SCH， CRTIN标志)

## 上行相关调度 (上行授权/上行数据的ACK响应)

### DCI0 调度(UE发送上行数据的授权)

确定给UE的上行授权相关参数，（即pico API HI\_DCI.request 消息中DCI UL PDU）。



说明：DCI信息为物理层用于发送数据使用的控制信息，最终由PDCCH 物理信道发送到空中。

### 上行数据的ACK

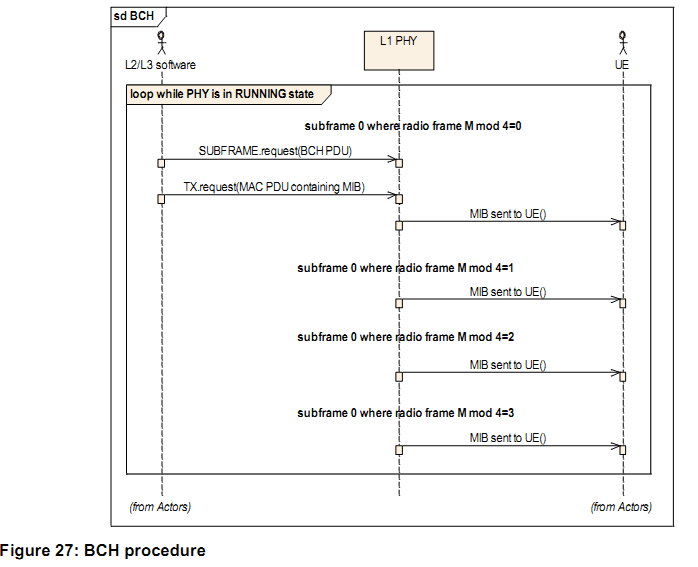
确定对UE上行数据的ACK/NACK反馈信息，（即pico API HI\_DCI.request 消息中HI PDU）。

## 下行数据调度

### BCH 调度

BCCH2BCH (对应 逻辑信道BCCH2BCH->传输信道PCH->物理信道PDSCH 的信息)

确定发送BCH PDU相关的信息(包括确定发BCH 数据相关的参数、待发BCH数据)



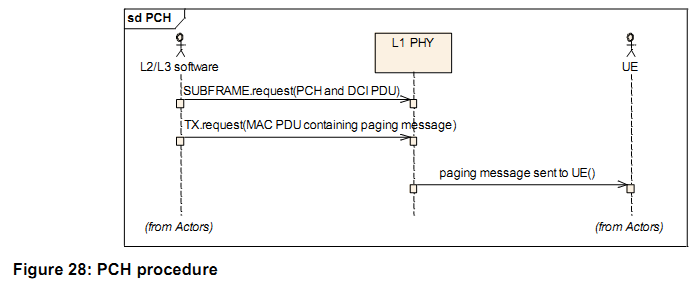
说明 ： 发送 BCH MIB。

### PCH 调度

PCH 调度(对应 逻辑信道PCCH->传输信道PCH->物理信道PDSCH 的信息)

确定发送PCH数据相关的信息，包括确定发PCH 数据相关的参数（即pico API SUBFRAME.request 消息中PCH PDU、DCI PDU）、待发PCH数据 (最后要包含到pico API TX.request 消息中)。

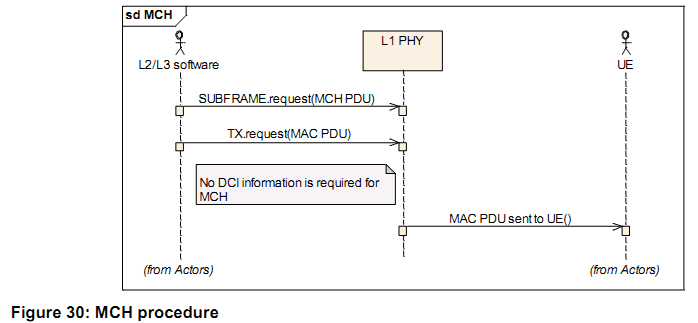
说明：DCI信息为物理层用于发送数据对应使用的控制信息，最终由PDCCH 物理信道发送到空中。



### MCH调度(暂时不考虑对MCH上的数据调度)

MCH 调度(对应 逻辑信道MTCH/MCCH->传输信道MCH->物理信道PMCH 的信息)

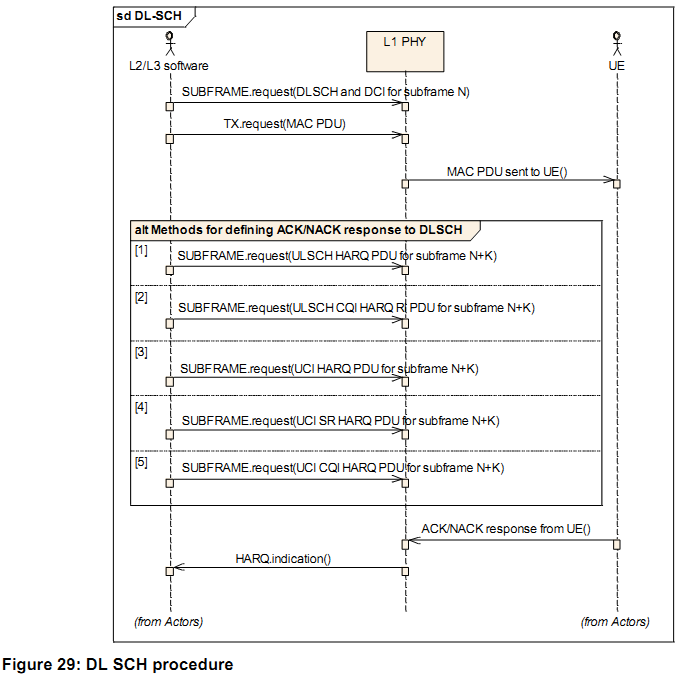
确定发送MCH数据相关的信息，包括确定发MCH 数据相关的参数（即pico API SUBFRAME.request 消息中MCH PDU、DCI PDU）、待发MCH数据 (最后要包含到pico API TX.request 消息中)。



### DL-SCH调度

DLSCH 调度(对应 逻辑信道CCCH/DCCH/DTCH/ BCCH2SCH ->传输信道DL\_SCH->物理信道PDSCH 的信息)

确定发送DLSCH数据相关的信息，包括确定发DLSCH数据相关的参数（即pico API SUBFRAME.request 消息中DLSCH PDU、DCI PDU）、待发DLSCH数据 (最后要包含到pico API TX.request 消息中)。



DLSCH调度流程

调度算法设计：

1. 轮询调度

根据基站配置估计可用的总调度资源 R

R\_temp = 0 ;// 当前剩下的总资源

For I = 1 to n //从第1个UE到 第 n个UE 分配资源

UE 逻辑信道信息按优先级高到低排序；

For j=1 to lc //从第1个逻辑信道 to 第 lc 个逻辑信道

R\_temp = R – lch(j) ;// lch(j)为逻辑信道j分配的数据大小

End for

If j> lc

调用rlc接口获取各lc数据，构造UE i的 DL-SCH mac PDU

决策发送 mac pdu 对应的 控制参数

Endif

End for

逻辑信息数据中SRB信息的调度优先级 应高于 DRB，若某UE DRB 数据量较大，后面的UE将无法得到服务。故先为每个UE的DRB分配 资源，再为剩下DRB按优先级高到低分配资源。

1’)

R\_temp = 0 ;// 当前剩下的总资源

For I = 1 to n //从第1个UE到 第 n个UE 分配SRB资源

For j=3 to lc //从第1个逻辑信道 to 第 lc 个逻辑信道

R\_temp = R – lch(j) ;// lch(j)为逻辑信道j分配的数据大小

End for

End for

For I = 1 to n //从第1个UE到 第 n个UE 分配DRB资源

UE 逻辑信道信息按优先级高到低排序；

For j=3 to lc //从第1个逻辑信道 to 第 lc 个逻辑信道

R\_temp = R – lch(j) ;// lch(j)为逻辑信道j分配的数据大小

End for

If j> lc

调用rlc接口获取各lc数据，构造UE i的 DL-SCH mac PDU

决策发送 mac pdu 对应的 控制参数

Endif

END for

1. PF 正比公平算法调度

根据基站配置估计可用的总调度资源 R (可能为不连续的资源块);

R\_temp = 0 ;// 当前剩下的总资源

// UE / 逻辑信道按优先级排序 形成 待调度队列

For I = 1 to n //从第1个UE到 第 n个UE 分配资源

//为用户分配一个优先级P 按优先级P高到低分配资源 ；

P = 用户在t时刻的载干比(信道质量)/用户的平均数据传输率(吞吐量)；//UE实体中应//保存UE反馈的最近CQI，最近k帧调度数据大小

按UE优先级 P高到低排序；//插入排序形成UE序列

For j=1 to lc //从第1个逻辑信道 to 第 lc 个逻辑信道

按逻辑信道优先级排序 //lc 优先级由高层配置

End for

End for

//为各UE 分配资源

For I = 1 to n //从第1个UE到 第 n个UE 分配资源

For j=1 to lc //从第1个逻辑信道 to 第 lc 个逻辑信道

R\_temp = R – lch(j) ;//?????

End for

If j> lc

构造UE mac PDU

决策发送 mac pdu 对应的 控制参数

Endif

End for

说明 ：逻辑信道优先级也可以 考虑采用**令牌桶算法.**

处理调度结果

将调度结果(发数据使用的控制信息、调度数据)封装成 lowmac用于向phy物理层发送数据的格式，调用l1api 接口发往物理层。

说明：需要按phy api的要求，按规定时序发送lowmac（phy api）消息，下行数据发送与上行授权的发送要发送不同l1api消息 。

# 其他说明

## L1api 下行发送的控制信息数据结构

### PDU 节点

typedef enum{

*L1API\_DCI\_DL\_PDU* = 0,

*L1API\_BCH\_PDU*,

*L1API\_MCH\_PDU*,

*L1API\_DLSCH\_PDU*,

*L1API\_PCH\_PDU*,

}DlPduType;

typedef struct L1ApiMsgPduNode\_s{

NodeType ln;

UINT8 pdu\_type;

UINT8 pdu\_size;

UINT16 reserved;

void \*pdu\_cfg\_p; //类型为L1ApiMsgDciDlPdu\_t/ L1ApiMsgBchPdu\_t/ //L1ApiMsgMchPdu\_t等下行PDU中一种

} L1ApiMsgPduNode\_t;

### DCI DL PDU

/\* DCI DL PDU \*/

typedef struct L1ApiMsgDciDlPdu\_s

{

UINT8 dci\_format;

UINT8 cce\_index; /\*CCE index used to send the DCI.\*/

UINT8 aggregation\_level;

UINT8 reserved;

UINT16 rnti; /\* The RNTI used for the receiving UE \*/

UINT8 resource\_alloc\_type; /\* Valid for DCI formats: 1,2,2A \*/

UINT8 vrb\_assignment\_flag;

UINT32 rb\_coding;

UINT8 mcs\_1st\_tb;

UINT8 redundancy\_ver\_1st\_tb;

UINT8 new\_data\_ind\_1st\_tb;

UINT8 tb2codeword\_swap\_flag;

UINT8 mcs\_2nd\_tb;

UINT8 redundancy\_ver\_2nd\_tb;

UINT8 new\_data\_ind\_2nd\_tb;

UINT8 harq\_process\_num;

UINT8 tpmi; /\* The codebook index to be used for precoding \*/

UINT8 pmi; /\* Confirmation for precoding \*/

UINT8 precoding\_info;

UINT8 tpc; /\* Tx power control command for PUCCH. \*/

UINT8 dl\_assignment\_index;

/\* Indicates which gap value to use for distributed virtual resource blocks \*/

UINT8 ngap;

UINT8 tb\_size\_index;

/\* Indicates the DL power offset type for multi-user MIMO transmission \*/

UINT8 dl\_power\_offset;

/\* Indicates that PRACH procedure is initiated \*/

UINT8 alloc\_prach\_flag;

/\*The preamble index to be used on the PRACH \*/

UINT8 preamble\_index;

UINT8 prach\_mask\_index;

UINT8 scramble\_type;

}L1ApiMsgDciDlPdu\_t;

### BCH PDU

/\* The format of the BCH PDU \*/

typedef struct L1ApiMsgBchPdu\_s

{

UINT16 length;

UINT16 pdu\_index;

UINT16 tx\_power;

UINT16 reserved;

}L1ApiMsgBchPdu\_t;

### MCH PDU

/\* The format of the MCH PDU \*/

typedef struct L1ApiMsgMchPdu\_s

{

UINT16 length;

UINT16 pdu\_index;

UINT16 rnti; /\*he RNTI associated with the MCH \*/

UINT8 resource\_alloc\_type;

UINT8 reserved1;

/\*The encoding for the resource blocks.\*/

UINT32 rb\_coding;

UINT8 modulation\_type;

UINT8 reserved2;

UINT16 tx\_power;

}L1ApiMsgMchPdu\_t;

### DLSCH PDU

typedef struct L1ApiMsgDlSchPdu\_s

{

UINT16 length;

UINT16 pdu\_index;

UINT16 rnti; /\*The RNTI associated with the MCH \*/

UINT8 resource\_alloc\_type;

UINT8 vrb\_assignment\_flag;

UINT32 rb\_coding;

/\* Indicates which gap value to use for distributed virtual resource blocks \*/

UINT8 ngap;

UINT8 mcs;

UINT8 redundancy\_ver;

UINT8 tb\_num;

UINT8 reserved1;

UINT8 tx\_scheme;

UINT8 layers\_num;

UINT8 codebook\_index;

UINT8 ue\_capacity;

/\*The ratio of PDSCH EPRE to cell-specific RS EPRE among PDSCH REs

in all the OFDM symbols not containing cell-specific RS in dB. \*/

UINT8 p\_a;

UINT8 delta\_power\_offset\_index;

UINT8 reserved2;

}L1ApiMsgDlSchPdu\_t;

### PCH PDU

/\* The format of the PCH PDU \*/

typedef struct L1ApiMsgPchPdu\_s

{

UINT16 length;

UINT16 pdu\_index;

UINT16 p\_rnti; /\*The P-RNTI associated with the paging \*/

UINT8 resource\_alloc\_type;

UINT8 vrb\_assignment\_flag;

UINT32 rb\_coding;

/\* Indicates which gap value to use for distributed virtual resource blocks \*/

UINT8 ngap;

UINT8 mcs;

UINT8 redundancy\_ver;

UINT8 tb\_num;

UINT8 reserved1;

UINT8 tx\_scheme;

UINT8 layers\_num; /\* The number of layers used in transmission \*/

UINT8 reserved2[2];

/\*The ratio of PDSCH EPRE to cell-specific RS EPRE among PDSCH REs

in all the OFDM symbols not containing cell-specific RS in dB. \*/

UINT8 p\_a;

UINT8 reserved3[2];

}L1ApiMsgPchPdu\_t;

## L1api 下行发送数据对应数据结构

### Tx pdu

**typedef** **struct** L1ApiMsgTxReq\_s

{

UINT16 sfn\_sf; /\* [15:4] SFN, [3:0] SF \*/

UINT16 pdus\_num;

L1ApiMsgPduList\_t tx\_req\_pdu\_list; /\* L1ApiMsgTxPduNode\_t \*/

}L1ApiMsgTxReq\_t;

### Tx pdu node

/\* TX.request message body \*/

**typedef** **struct** L1ApiMsgTxPduNode\_s

{

NodeType ln;

UINT16 pdu\_len;

UINT16 pdu\_index;

UINT8 \*pdu\_data\_p;

}L1ApiMsgTxPduNode\_t;

## 目前调度的实现与问题

**henb\_dl\_sched**

{//进行下行调度取得 调度结果

BCH 调度; //待修改,还未与rrc层的BCH处理关联

PCH 调度; //未实现, 未与rrc层的PCH处理关联

MCH 调度; //未实现

// DL-SCH 调度 ： 如下

build\_dl\_sched\_queue //创建下行调度队列 (包含待调度大小)，其中对UE与逻辑信道大小根据 物理信道 最大字发送节数进行限制

do\_dl\_sched //根据下行调度队列结点，取得下行调度结果结点。根据物理资源块分配资源时只发送 大小<已有资源块大小 的待发数据。

}

目前 DL-SCH 调度的问题：

1. 部分数据无法调度

根据物理资源块分配资源时只发送 大小<已有资源块大小 的待发数据。

1. 调度参数不全，不一致

调度结果参数与l1api中数据发送对应控制信息不一致，具体见下面。

未区分不同类型 pdu数据的不同控制信息参数，见4.1节。

typedef struct {

NodeType ln;

LchResultLst lch\_result\_lst;

UINT32 size;

UINT16 rb\_start;

UINT16 rb\_len;

UINT16 rnti;

UINT8 mcs;

UINT8 ndi;

/\*retx schedule process\*/

UINT16 proc\_num;

} DlSchedSchResultNode;

1. 待确定 l1api 与mac 处理的交互实现。

直接调度出发送数据时 l1api 消息对应的 控制、数据 信息格式(直接使用 l1api定义的数据结构，见4.1节)；

增加目前数据结构中调度变量，l1api中增加中间转换过程(增加调度参数、中间转换易出错，转换耗时比前一种长)。

1. 待理解l1api发送各类pdu对应的 控制信息参数意义