**联调问题与总结**

在九月份的联调版本中，我们一共有四个版本：信令测试、下行分级业务、下行峰值及上行峰值。在这四个测试场景的搭建过程中，我们遇到了很多问题，有一些问题已经成功解决，但是还有一些问题仍然存在，只是用一些其他方法掩盖而达到测试效果。下面将分别对这四个场景进行详细说明。

#### 1 信令场景

此次搭建的信令场景的运行环境：MME运行在Linux PC机（141）上、HENB运行的picochip板卡上、UE运行在终端板卡上、wireshark运行在Windows PC机上。在此场景中没有同步机制；两套协议栈的TTI设置为20ms；没有DCI的发送与接收，只是单纯的上下行数据的发送与接收。

通过修改基站侧/source/defs/henb\_config.c文件中的server\_addr和server\_port可以控制接收信令的wireshark的配置；

通过修改基站侧/source/henb\_driver/henb\_driver.c文件中的init\_tcp\_socket函数中的open\_kernel\_socket("10.21.1.141", 17000, 28000)的第三个参数来设置接收VLC数据的端口，并且通过查看/debug/test\_all/init\_henb.c中的construct\_data\_func函数可以找到socket与RB的对应关系；

通过修改终端侧/source/common/lte\_socket.c中的全局变量ip\_addr1、ip\_port1…..来设置发送VLC数据的套接字；并且可以通过查看/include/main\_include/pre\_ue\_test.h文件中的rx\_ip\_data函数查看socket与RB的对应关系。

* 1. 协议栈已解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 解决方法 |
| 两个物理板连接后，一旦有数据交互就会出现段错误。 | 由于picochip使用大端进行编码，而终端板卡使用小端，在TCP实现中，第一个接收的是数据的长度，此处直接使用强制转换，所以此处会出现大小端不一致导致的错误； | 只需要将两侧协议栈中接收到的length进行一下32位的大小端转换即可，在实现中是写了一个转换大小端的宏，然后在处理length的地方调用。 |
| 所有的信令完成后，发起一个RB的视频业务，RLC及PDCP出现错误 | 错误一直定位在解包过程中，由于RLC及PDCP实体都是通过信令创建，所以初步认为两端关键参数不一致，通过printf打印发现，当在UM模式下时，基站侧RLC的DL\_SN=10,PDCP的SN=12，但是终端侧RLC的DL\_SN=5， PDCP的SN=5. | 通过查看RRCReConfig信令发现UE的配置与信令一致，表明基站存在错误，通过修改基站端相应的信令配置文件，使得两端的实体配置一致，从而正常解析。 |
| 信令完成并且配置一致后，发起一个RB的视频业务，但是VLC客户端无法显示。 | 发起视频业务后，通过打印发现协议栈可以正常解析并且往外递交，并且外部通过wireshark也能够抓取此端口的数据包，但是VLC客户端就是没有任何反应。通过show\_memory基站侧收到VLC数据和终端侧发出去的VLC数据发现两个数据不一致。通过查看代码发现两侧加解密不统一。 | 在PDCP侧，终端侧下行接收过程将解密操作注释，所以为了方便及提高协议栈处理性能决定将基站侧的下行加密过程也注释掉；据此操作后，视频能够正常播放。 |

* 1. 协议栈未根本解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 暂时解决方法 |
| 两套协议栈没有进行同步 | 由于以前写的外部同步定时器不准时并且间隔时间太久，无法满足信令完成后视频业务的传输，所以无法使用此同步机制。但是如果不同步则不能进行上行授权及ACK等的交互。 | 两端没有进行同步，基站不发送DCI及ACK等信息，并且中断在判断有无上行授权到达时，默认有授权接收并且发送数据。 |
| 信令偶尔无法全部完成 | 当测试信令时，信令经常会停在终端发送preamble前、接收msg4后或者进行安全能力之后 | 此问题的原因初步认为是终端的非同步机制导致，并不是协议栈代码不完善，并且失败率在接收范围内，所以暂时没有理会，只是在基站启动后等一段时间再启动终端，这样成功率会较高。 |
| 基站侧运行一段时间后会发生段错误，并且断在帧定时器超时函数中。 | 段错误的位置为帧超时函数，但是此地出现段错误的可能性不大，有可能是其他地方内存越界导致。 | 由于时间紧迫，并且没有使用harq的各种作用，所以将harq部分暂时注释掉，等待联调完成后再进行此段错误的定位及修正。 |
| 基站和终端在进行视频业务后，过一段时间分别有可能会发生段错误 | 此时发生的段错误位置不固定并且发生时间也不固定，有时视频业务开始就有，有时是较长时间才会出现。 | 暂时搁置 |
| 信令建立时，当全部建立完全后偶尔会发送RRC Release消息 | 当信令全部运行完后，偶尔会发送RRC Release消息，导致基站的RNTI实体删掉 | 将MME发送RRC Release消息的这条信令暂时删掉，后续检查发送此消息的具体原因。 |
| 信令完成后，视频业务发起时，基站会发生counter\_check的状态机错误 | 视频业务时，PDCP在发送DRB时，满足一定条件后会给RRM状态机一个消息，使状态转到counter\_check,但是目前好像条件判断不准确，导致每次都会发这条指令，状态机发生错误 | 暂时屏蔽PDCP向RRM递交的这条信息 |
| 上行视频业务能够显示但是相当不流畅 | 上行的帧配置等基本是改成与下行一致，但是仍然无法使上行视频流畅的显示 | 此路视频改为下行视频，效果良好 |

* 1. 其他问题总结

1. 基站板出现段错误时，可以将core文件copy到可执行文件的源PC目录目录下，用mips64-octeon-linux-gnu-gdb进行调试，或者使用gdbserver进行调试。终端类似。
2. 终端使用scp拷贝时，要使用非root用户
3. 信令过程是，要尽量等到基站初始化完毕后再启动终端，这样成功率较高。
4. 执行之前一定要检查终端侧的config\_lte.txt文件中的配置与基站一致。
5. 等信令全部完成后再发起视频业务。

#### 2 下行分级业务

此次搭建的下行分级业务的运行环境：HENB运行在PC(32机)、UE运行在PC(141)、在PC(141)上运行自己写的ping可执行文件;此场景没有信令的交互，RB在系统初始化时已经建立，其中HENB建立了3、4、5三个RB，UE建立的所有的RB；为了提高性能将无用的控制面的线程关闭；由于1ms中断不准确，在运行时使用2ms的中断；使用配置2.

通过修改基站侧/source/henb\_driver/henb\_driver.c文件中的init\_tcp\_socket函数中的open\_kernel\_socket("10.21.1.141", 17000, 28000)的第三个参数来设置接收外部数据的端口，并且通过查看/debug/test\_all/init\_henb.c中的construct\_data\_func函数可以找到socket与RB的对应关系；目前是28000的RB为3且对应ping，28001的RB为4且对应iperf，28002的RB为5且对应VLC

通过修改终端侧/source/common/lte\_socket.c中的全局变量ip\_addr1、ip\_port1…..来设置发送外部数据的套接字；并且可以通过查看/include/main\_include/pre\_ue\_test.h文件中的rx\_ip\_data函数查看socket与RB的对应关系。

* 1. 协议栈已解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 解决方法 |
| 下行速率测试TTI不准时 | 由于协议栈的TTI实现的是软中断，准确定较低，该线程必须得到CPU的调度实际才会进行定时器的超时处理，所以初步认为是线程太多，轮转调度使得空闲进程占用太多不必要的CPU时间 | 1．去掉所有的控制面相关的线程；  2. 以前分三个线程接收外部数据，目前使用文件标识符从一个线程里面对三个接收socket进行监听，然后非阻塞的接收三个不同地方的数据；  3. PDCP侧发送DRB时拷贝次数太多，但是都是函数内部拷贝然后释放，完全没有意义，将这些拷贝去掉。 |
| 发送大数据时，内存下降很快 | 当发送50M+的数据时，内存很有明显的下降，基站执行一段时间后就会自动停掉，通过查看RLC buffer发现缓存里面的数据并不是太多，每次基本上都可以全部完成调度，于是怀疑为内存泄露。 | 通过使用valgrind查看，发现有多处存在线程的内存泄露，主要是调度及lowmac处，修正后效果明显，内存基本上保持不变。 |
| 终端运行一段时间后会出现PDCP状态报告的错误 | 终端运行一段时间后，会出现PDCP状态报告的问题，但是目前RLC的模式为UM，不应该存在PDCP状态报告，通过对RLC及PDCP的运行进行跟踪，发现并没有异常，但是查看缓存的接收数据时发现list中的节点过多，进一步调查发现接收数据使用的是一块共享内存，最终原因就是上层处理太慢，导致共享内存数据覆盖，导致错误。 | 1. 去掉PDCP的加密，提高PDCP的处理能力 2. 减少终端侧不必要的线程，提高CPU的利用率 3. 增大共享内存，使共享内存不至于较容易的发送内存覆盖，修改的为sim\_phy.h里面的宏 TX\_BUF\_NUMBER |

* 1. 协议栈未根本解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 暂时解决方法 |
| PDCP加解密及头压缩 | 由于PDCP层的加解密及头压缩有大量的内存申请及释放，严重影响下行速率。 | 由于加解密在软件层实现的可能不大，并且目前也没有完善的头压缩机制，所以现在不做那些内存申请及拷贝。但是在真正实现中仍然有改动的空间，可以将那些在函数中申请并释放的堆空间改为栈空间，这样会节省很大的开销。 |
| 下行时，基站运行一段时间后会在harq的地方存在段错误 | 基站在pc上运行一段时间后，会在帧超时的地方出现段错误，但是查看harq和帧定时器均没有发现错误，初步估计是某个地方出现数组越界访问导致。 | 为了节省时间，并且基站与终端之间没有harq的交互，将基站侧harq及帧定时器部分暂时屏蔽。 |
| 最终版本中偶尔两侧均会出现段错误 | 下行运行一段时间后，基站或者终端就有可能出现段错误，应该是协议栈设计不完善，还存在内存越界访问的地方 | 暂时搁置 |
| Picochip发生段错误 | 当使用picochip发送iper的大数据时，发送开始后很快就会出现段错误，并且段错误每次都断在pdcp中的send\_drb中的malloc处，但是通过检查代码发现那个地方应该不会存在导致段错误的代码。 | 由于每次都是这样，并且在PC上没有问题，所以估计与picochip中的glibc有关，所以此次演示决定运行在PC机上。 |

* 1. 其他问题总结

1. 基站协议栈中目前存在大量的无用全局变量和函数，需要有时间时进行集中清理。
2. 协议栈中不必要的内存拷贝仍然很多，在优化时需要进行集中处理。
3. 目前的线程设计所有的线程优先级一样，这样会导致一些关键的操作得不到及时处理，以后可以改进线程设计思路。
4. 基站侧的帧定时器存在缺陷，需要进行修正。
5. 基站侧mac padding有可能存在与协议相悖的地方，需要进行查询及修正。
6. 目前只是测试UM，AM也应该拿出时间长测试下。

#### 3 下行峰值

下行峰值的场景与下行分级场景的代码为一套，只是在一些参数配置上存在差异，主要是调度大小和下行帧数量上。问题与上面的一致。

#### 4 上行峰值

此次搭建的上行峰值的运行环境：HENB运行在PC(32机)、UE运行在PC(141);此场景没有信令的交互，RB在系统初始化时已经建立，其中HENB建立了3、4、5三个RB，UE建立的所有的RB，但上行中只是用RB=3的相关实体；由于没有信令的交互，所以终端无法获取基站给的上行授权，所以终端在每次检查上行授权时，都会默认的收到一个固定大小的授权，这样就会将数据及时的进行下发，不会因为等待DCI导致性能较低；为了提高性能将无用的控制面的线程关闭；由于1ms中断不准确，在运行时使用2ms的中断；并且由于配置2的上行帧较少，所以使用配置0

通过修改基站侧/source/henb\_driver/henb\_driver.c文件中的init\_tcp\_socket函数中的open\_kernel\_socket("10.21.1.141", 17000, 28000)的第一、二个参数来设置发送到外部数据的套接字，并且通过查看/source/pdcp/pdcp.c中的ip\_receive函数可以找到通过socket（调用的函数的第一个参数i的含义为创建的第i-1个kernel socket）将数据发送出去；

通过修改终端侧/source/common/lte\_socket.c中的open\_socket\_kernal函数中的rec\_to\_kernal[socket\_index]绑定的端口号来设置接收外部数据的套接字；并且可以通过查看/include/main\_include/pre\_ue\_test.h文件中的tx\_msg\_from\_kernal\_task函数查看socket与RB的对应关系。

4.1协议栈已解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 解决方法 |
| 没有上行授权进行上行发送 | 由于没有下行数据的发送与接收，所以终端不会收到来自基站的上行授权。 | 所以终端侧在处理上行授权时，假设已经收到，并且进行mac包的构造 |

4.2协议栈未根本解决问题总结

|  |  |  |
| --- | --- | --- |
| 问题 | 问题描述 | 暂时解决方法 |
| 基站侧出现mac unpack错误 | 数据通路通后，基站侧接收有可能会出现unpack mac pdu的错误，主要是包里面的length与接收到的length不匹配，导致错误 | 由于为偶而出现，暂时没有解决 |
| 数据跑上一定时间后会变得很不稳定 | 数据运行几分钟后，发现速率会变得很小，并且很不稳定，但是iperf的数据照常发送、ue照常接收。目前还不确定是由于UE上行调度没有进行及时处理还是HENB上行接收没有进行及时解析 | 由于此问题在最后的演示时才发现，所以暂时进行了搁置。 |

4.3 其他问题总结

1.使用iper向终端灌包的长度必须限定在2000字节之内。