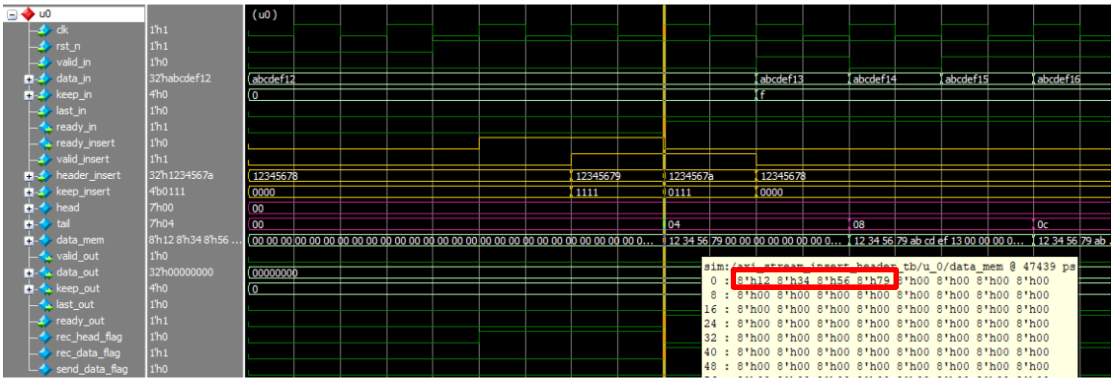
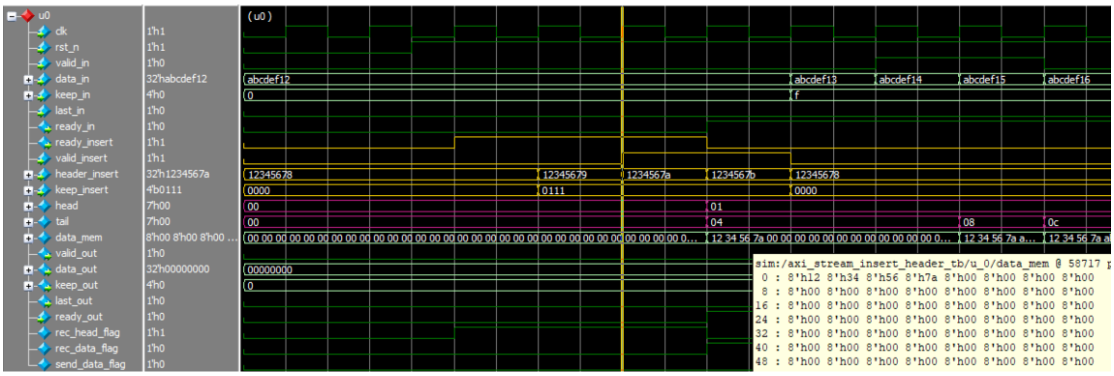
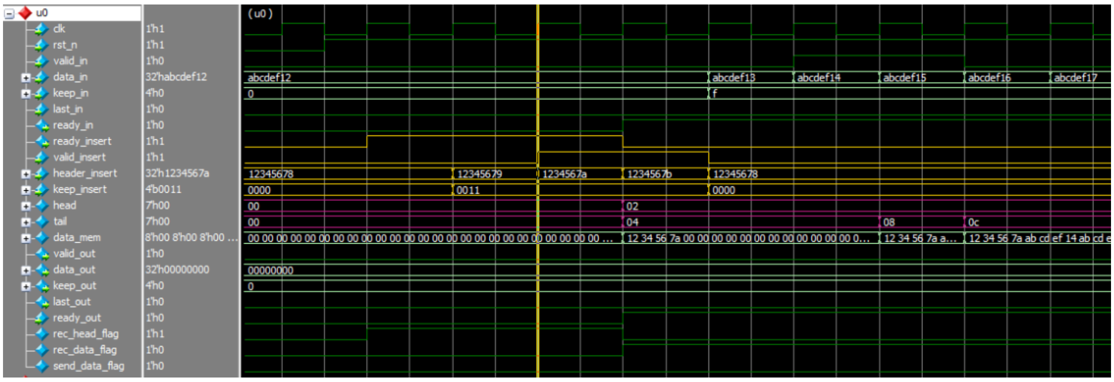
仿真结果

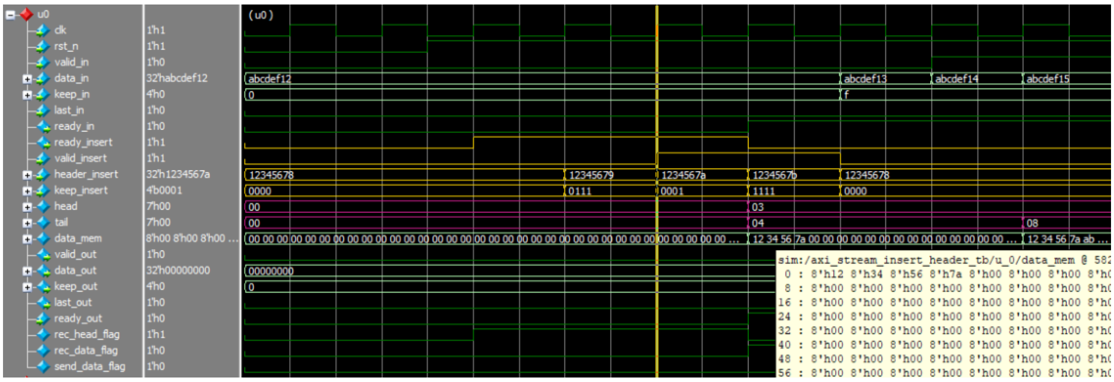
1 仿真说明

1.1 工况 1



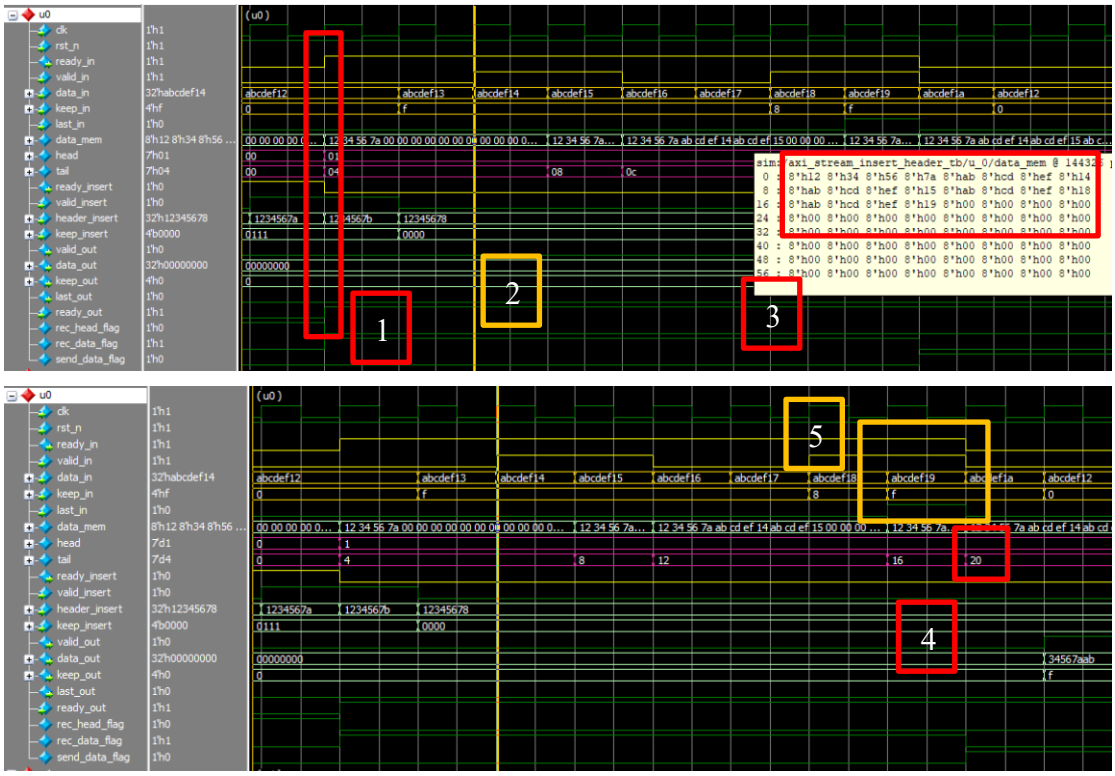






如图所示，当 ready_insert 和 valid_insert 都为‘1’时，将 header_insert 存入 data_mem 中，并且根据 keep_insert 更新 head 和 tail，keep_insert 为“1111”时，head 为 0，发送数据是从地址 0 开始发；keep_insert 为“0111”时，head 为 1，发送数据是从地址 1 开始发；keep_insert 为“0011”时，head 为 2，发送数据是从地址 2 开始发；keep_insert 为“0001”时，head 为 3，发送数据是从地址 3 开始发。

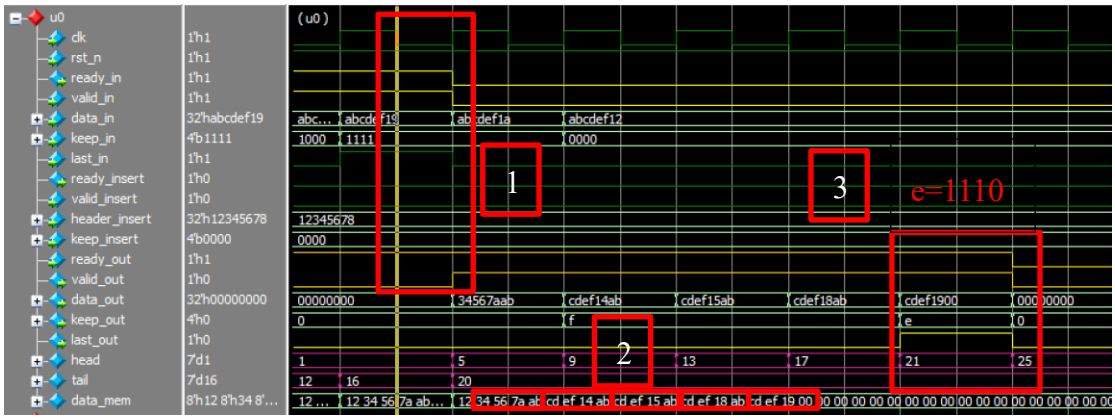
1.2 工况 2



如图所示，由 Mark1 可以看出，ready_insert 拉低时，ready_in 拉高，开始接收数据部分；由 Mark2、Mark3、Mark4 可以看出，当 ready_in 和 valid_in 都为高时，将对应的 data_in 存入 data_mem，并且修改计算寄存器长度的 tail 地址信号，一共存了 4 个数据，加上帧头，共 5 个数据，所以 tail 为 20；通过 Mark3 可以看出，data_mem 里存储的数据与 ready_in 和 valid_in 都为高时的 data_in 数据是一一对应的；由 Mark5 可以看出，当 last_in 拉高时，代表该数据为本次传输最后一个数据，将 ready_in 拉低，停止接收数据。

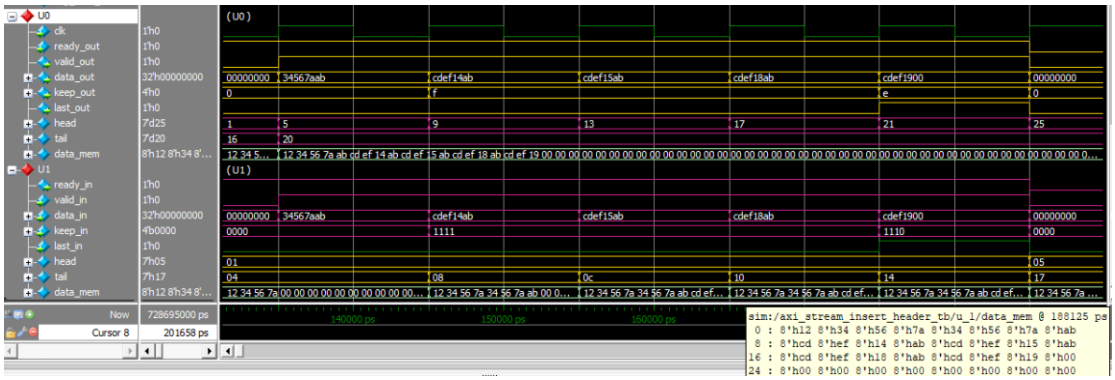
如图所示，当 last_in 拉高时，需要通过 keep_in 判断有效位，当 keep_in 为 1110 时，最后一字节无效，tail 为 19；当 keep_in 为 1100 时，最后两字节无效，tail 为 18；当 keep_in 为 1000 时，最后三字节无效，tail 为 17；当 keep_in 为 0000 时，所有字节无效，tail 为 16。

1.3 工况 3

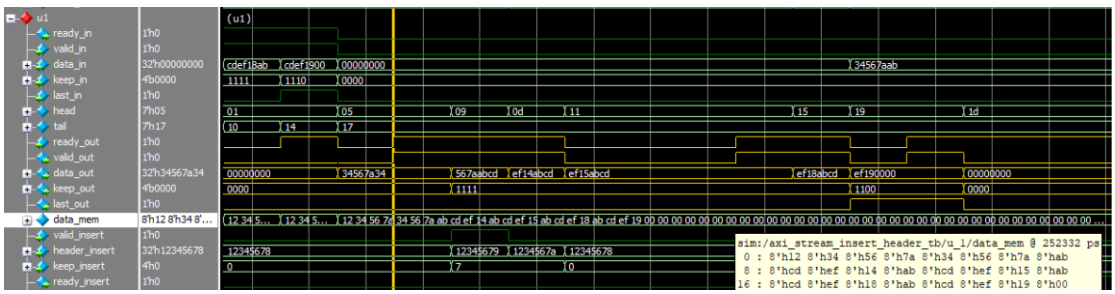


如图所示，由 Mark1 可以看出，valid_in 拉低之后，valid_out 开始拉高，发送给数据，由于提前将数据放在寄存器中，可以直接输出，没有气泡产生；由 Mark2 可以看出，由于第一字节数据无效，从第二字节开始输出，data_out 与 data_mem 中的数据能够一一对应；由 Mark3 可以看出，由于最后一个数据最后一字节无效，keep_out 为 1110，并且 valid_out 发送完最后一个数据后拉低。

1.4 工况 4



如图所示，将 U0 与 U1 级联，U0 的输出作为 U1 的数据输入，可以得到如上图所示的 data_mem，下面主要关注 U1 的输出。



如图所示，valid_out 为高时的数据与 data_mem 中的数据一致，并且在 ready_out 拉低后再拉高时，不会丢失数据，也不会重复传输数据。