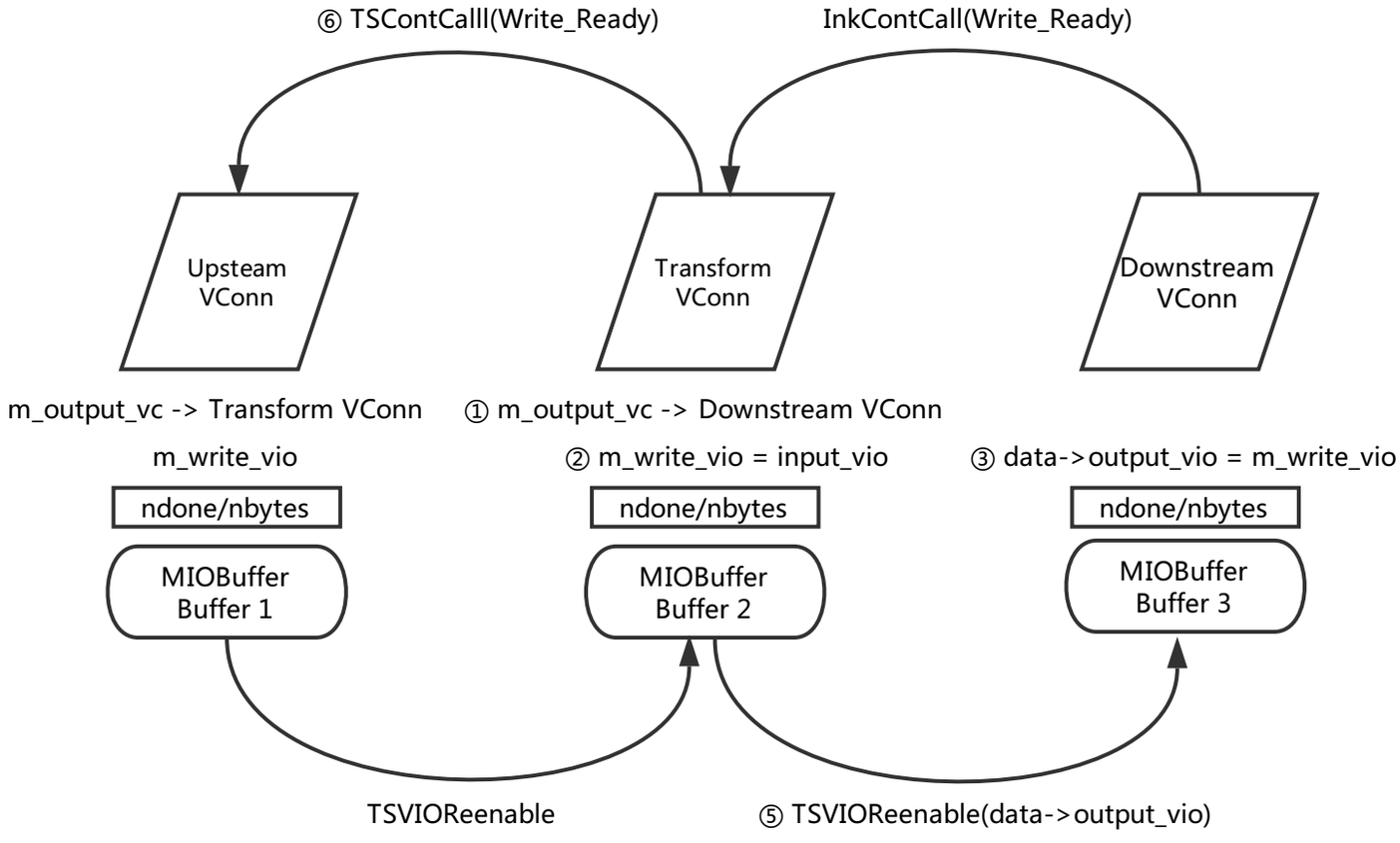

① 获得Downstream VConn(m_output_vc)

② 获得当前VConn的m_write_vio(input_vio)，注意这个VIO是由Upsteam VConn负责初始化
data->output_vio = TSVConnWrite(output_conn, contp, data->output_reader, INT64_MAX);
output_conn->do_io_write(contp, data->output_reader, INT64_MAX)

③ 创建MIOBuffer，作为参数初始化Downstream VConn的m_write_vio

④ 将input_vio对应MIOBuffer(Buffer 2)中的数据拷贝到Downstream VConn的m_write_vio(data->output_vio)，也就是MIOBuffer(Buffer 3)
TSIOBufferCopy(TSVIOBufferGet(data->output_vio), TSVIOReaderGet(input_vio), towrite, 0);

⑤ 通过Downstream VConn的m_write_vio(data->output_vio)向Downstream VConn发送event_immediate，告知其MIOBuffer(Buffer 3)已经准备好，可以继续操作了(譬如继续向自己的下游拷贝)
InkAPI.cc: data->output_vio->reenable

⑥ 通知Upsteam VConn本地MIOBuffer(Buffer 2)中的数据已经消费完毕，可以继续写了
TSContCall(TSVIOContGet(input_vio), TS_EVENT_VCONN_WRITE_READY, input_vio)
->
input_vio->_cout(TS_EVENT_VCONN_WRITE_READY, input_vio)
注意这个_cout是由Upstream VConn在初始化input_vio时赋值为Upstream VConn