正常发送：

--->SyS\_write

--->vfs\_write

--->tty\_write

--->n\_tty\_write

--->uart\_write

--->uart\_start

--->\_\_uart\_start

--->msm\_hs\_start\_tx\_locked

--->msm\_serial\_hs\_tx\_work

--->msm\_hs\_submit\_tx\_locked(uport);

--->sps\_transfer\_one(sps\_pipe\_handle, src\_addr, tx\_count,

产生中断：

--->bam\_isr

--->sps\_bam\_check\_irq

--->msm\_hs\_sps\_tx\_callback

--->queue\_kthread\_work(&msm\_uport->tx.kworker, &msm\_uport->tx.kwork);

--->init\_kthread\_work(&tx->kwork, msm\_serial\_hs\_tx\_work);

正常接收：

bam\_isr

--->sps\_bam\_check\_irq

---->pipe\_handler

---->pipe\_handler\_eot.isra.2

---->trigger\_event.isra.1

----->msm\_hs\_sps\_rx\_callback

----->msm\_serial\_hs\_rx\_work

---->msm\_hs\_queue\_rx\_desc(msm\_uport);

---->msm\_hs\_mark\_next(msm\_uport, msm\_uport->rx.rx\_inx+1);

---->msm\_hs\_write(uport,UART\_DM\_CR,START\_RX\_BAM\_IFC);

--->wake\_up(&msm\_uport->rx.wait);

---->msm\_serial\_hs\_rx\_work

----->tty\_flip\_buffer\_push(tty->port);

------>flush\_to\_ldisc

----->n\_tty\_receive\_buf

----->n\_tty\_receive\_char(tty, \*p);

----->process\_echoes(tty);

----->process\_echoes

----->tty->ops->flush\_chars(tty);

----->uart\_start

----->\_\_uart\_start

---->msm\_hs\_start\_tx\_locked

上层接受数据：

1：执行msm\_hs\_runtime\_resume

2：打开时钟：msm\_hs\_clk\_bus\_vote

3：与sps建立连接：msm\_hs\_spsconnect\_rx

4：msm\_hs\_start\_rx\_locked

5：设置波特率：msm\_hs\_set\_termios

6：msm\_serial\_hs\_rx\_work

=======

接受数据

1：msm\_hs\_runtime\_resume

--->处理中断，打开时钟。