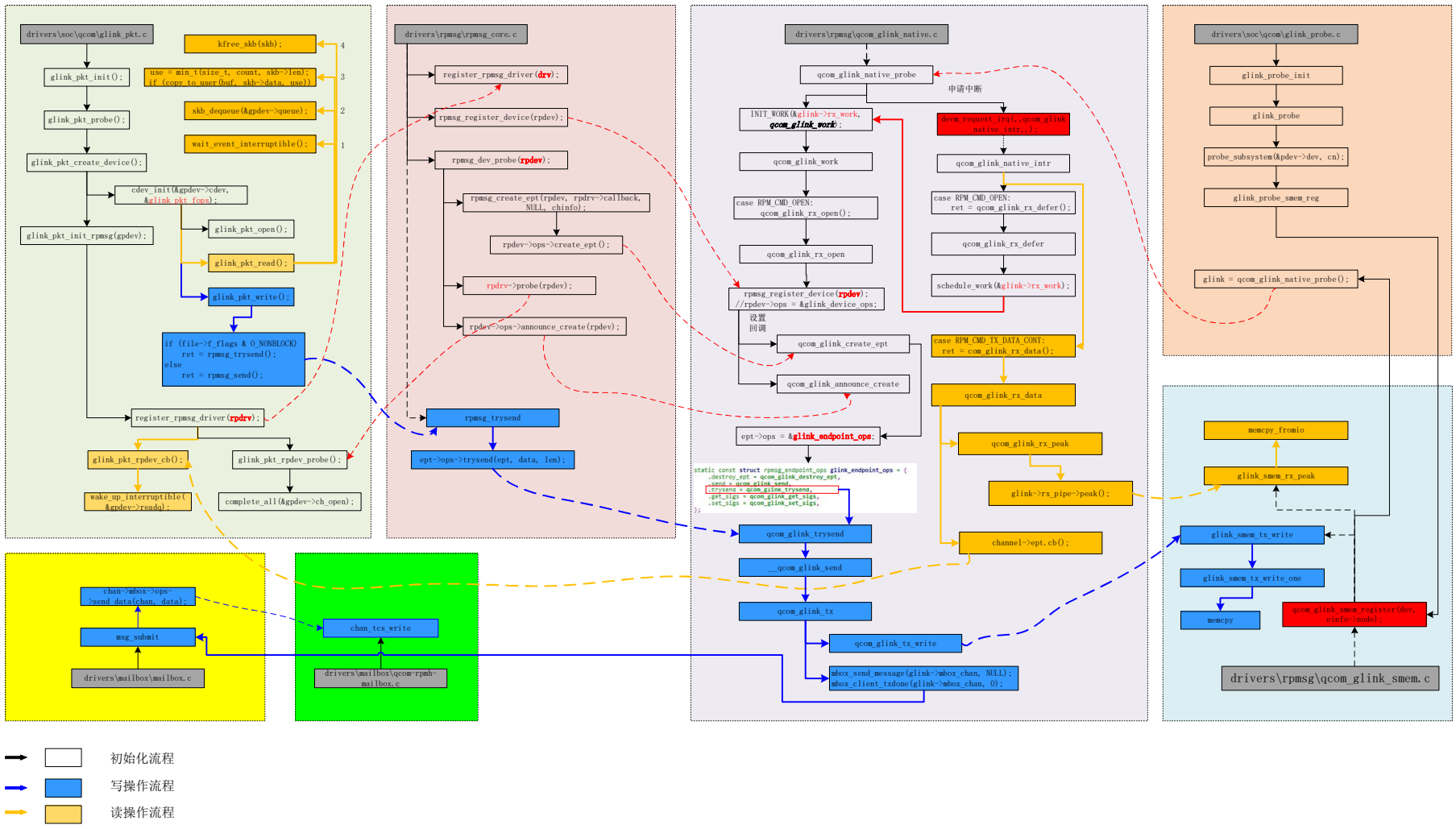
drivers\soc\qcom\glink_pkt.c
glink_pkt_init();
glink_pkt_probe();
glink_pkt_create_device();
cdev_init(&gpdev->cdev, &glink_pkt_fops);
glink_pkt_init_rpmsg(gpdev);
glink_pkt_open();
glink_pkt_read();
glink_pkt_write();
kfree_skb(skb);
use = min_t(size_t, count, skb->len);
if (copy_to_user(buf, skb->data, use))
skb_dequeue(&gpdev->queue);
wait_event_interruptible();
4
3
2
1
if (file->f_flags & O_NONBLOCK)
ret = rpmsg_trysend();
else
ret = rpmsg_send();
register_rpmsg_driver(rpdrv);
glink_pkt_rpdev_cb();
glink_pkt_rpdev_probe();
wake_up_interruptible(&gpdev->readq);
complete_all(&gpdev->ch_open);
drivers\rpmsg\rpmsg_core.c
register_rpmsg_driver(drv);
rpmsg_register_device(rpdev);
rpmsg_dev_probe(rpdev);
rpmsg_create_ept(rpdev, rpdrv->callback, NULL, chinfo);
rpdev->ops->create_ept();
rpdrv->probe(rpdev);
rpdev->ops->announce_create(rpdev);
rpmsg_trysend
ept->ops->trysend(ept, data, len);
drivers\rpmsg\qcom_glink_native.c
qcom_glink_native_probe
申请中断
INIT_WORK(&glink->rx_work, qcom_glink_work);
devm_request_irq(.,qcom_glink_native_intr,.);
qcom_glink_work
qcom_glink_native_intr
case RPM_CMD_OPEN:
qcom_glink_rx_open();
case RPM_CMD_OPEN:
ret = qcom_glink_rx_defer();
qcom_glink_rx_open
qcom_glink_rx_defer
rpmsg_register_device(rpdev);
//rpdev->ops = &glink_device_ops;
schedule_work(&glink->rx_work);
设置
回调
qcom_glink_create_ept
case RPM_CMD_TX_DATA_CONT:
ret = com_glink_rx_data();
qcom_glink_announce_create
qcom_glink_rx_data
ept->ops = &glink_endpoint_ops;
qcom_glink_rx_peak
glink->rx_pipe->peak();
static const struct rpmsg_endpoint_ops glink_endpoint_ops = {
.destroy_ept = qcom_glink_destroy_ept,
.send = qcom_glink_send,
.trysend = qcom_glink_trysend,
.get_sigs = qcom_glink_get_sigs,
.set_sigs = qcom_glink_set_sigs,
};
channel->ept.cb();
qcom_glink_trysend
__qcom_glink_send
qcom_glink_tx
qcom_glink_tx_write
mbox_send_message(glink->mbox_chan, NULL);
mbox_client_txdone(glink->mbox_chan, 0);
drivers\soc\qcom\glink_probe.c
glink_probe_init
glink_probe
probe_subsystem(&pdev->dev, cn);
glink_probe_smem_reg
glink = qcom_glink_native_probe();
memcpy_fromio
glink_smem_rx_peak
glink_smem_tx_write
glink_smem_tx_write_one
memcpy
qcom_glink_smem_register(dev, einfo->node);
drivers\rpmsg\qcom_glink_smem.c
chan->mbox->ops->send_data(chan, data);
msg_submit
drivers\mailbox\mailbox.c
chan_tcs_write
drivers\mailbox\qcom-rpmh-mailbox.c
初始化流程
写操作流程
读操作流程