```
SCSI_ALLOC_SDEV                                   COMPLETE REQUESST
 ↳ → QUEUE = SCSI_OLD_ALLOC_QUEUE()
          ↳ BLK_QUEUE_SOFT_IRQ_DONE → SCSI_SOFTIRQ_DONE()
     → RQ → SOFTIRQ_DONE_FN = SCSI_SOFTIRQ_DONE() ⊛
```

STATUS LOWER LEVEL ↓

```
SCSI_DEVICE_FROM_QUEUE                   RUN (QUEUE → SCSI_REQUEST_FN()
 ↳ REQ QUEUE → REQUEST_FN = SCSI_REQUEST_FN()        ↳ SCSI_DISPATCH_CMD()
            ↳ SCSI CMD → DONE = SCSI_DONE()             ↳ ISCSI_QUEUECOMMAND()
                                                       AFTER QUEUING, IF ERROR
            ↳ SCSI_DISPATCH_CMD()        ↳ BLK_COMPLETE_REQUEST()
                 ↳ ...                     ↳ BLOCKSOFTIRQ ⊛
                                         ISCSI_COMPLETE_TASK()
```

```
BLK_INIT_ALLOCATED_QUEUE()
 ↳ QUEUE → TIMEOUT_WORK = BLK_TIMEOUT_WORK()
      ↳ PER CMD TIMEOUT
                    ↳ BLK_RQ_CHECK_EXPIRED()
                       ↳ BLK_RQ_TIMED_OUT()
                          ↳ REQ QUEUE → RQ_TIMED_OUT_FN()                ⊛
                             ↳ COMPLETE_REQUESST → BLOCKSOFTIRQ
                             ↳ CLEAR_RQ_COMPLETE
```

SOFTIRQ

BLOCK_SOFTIRQ → BLK_DONE_SOFTIRQ()

↳ STRUCT REQUEST RQ = LOCAL_LIST IPI LIST

RQ → Q → SOFTIRQ_DONE_FN() = SCSI_SOFTIRQ_DONE() ⊛

REQUEST

REQUEST QUEUE

SCSI_OLD_ALLOC_QUEUE()

↳ CALLS → END I/O

WORKED?

FAIL

⊛ SCSI_SOFTIRQ_DONE()

CMD RQ→TIMEOUT

RQ

- SCSI_DECIDE_DISPOSITION() → STATUS CODE FROM LOWER LEVEL

→ SENSE

→ ACTION

→ SENSE_KEY

→ FAIL → SCSI_END_REQUEST

→ REPREP → SCSI_REQUEUE_COMMAND

→ RETRY

→ DELAYED RETRY → SCSI_QUEUE_INSERT

RETRY SAME

BASED ON SCSI CMD SENSE

- SUCCESS

SCSI_FINISH_COMMAND() → SCSI_IO_COMPLETION()

- NEEDS RETRY →

SCSI_QUEUE_INSERT() → SCSI_MLQUEUE_EH_RETRY

- ADD_TO_MLQUEUE

SCSI_QUEUE_INSERT() → SCSI_MLQUEUE_DEVICE_BUSY

HOST IS BUSY

DEVICE QUEUE FULL

↳ ZERO CMD RESULT

BLOCK_REQUEUE_REQUEST

↳ DELETE TIMER

↳ CLEAR COMPLETE

↳ ELV_REQUEUE_REQUEST

- DEFAULT

SCSI_EH_SCMD_ADD

SCSI_DONE FROM SCSI_CMD

#1

↳ ON ERRORS, USUALLY: ISCSI_COMPLETE_TASK() → ISCSI_PUT_TASK() → ISCSI_FREE_TASK() → SCSI_DONE()

- ↳ BLK_COMPLETE_REQUEST()
  - ↳ BLOCKSOFTIRQ *

FOR SPECIFIC COMMANDS

#2 NETWORK

ISCSI_TCP_HDR_DISSECT
- ↳ ISCSI_COMPLETE_PDU
  - ↳ ISCSI_COMPLETE_TASK
    - ↳ ISCSI_PUT_TASK
      - ↳ ISCSI_FREE_TASK
        - ↳ SCSI_DONE
          - ↳ BLK_COMPLETE_REQ
            - ↳ BLOCKSOFTIRQ

#3 BLOCK DEV TIMEOUT

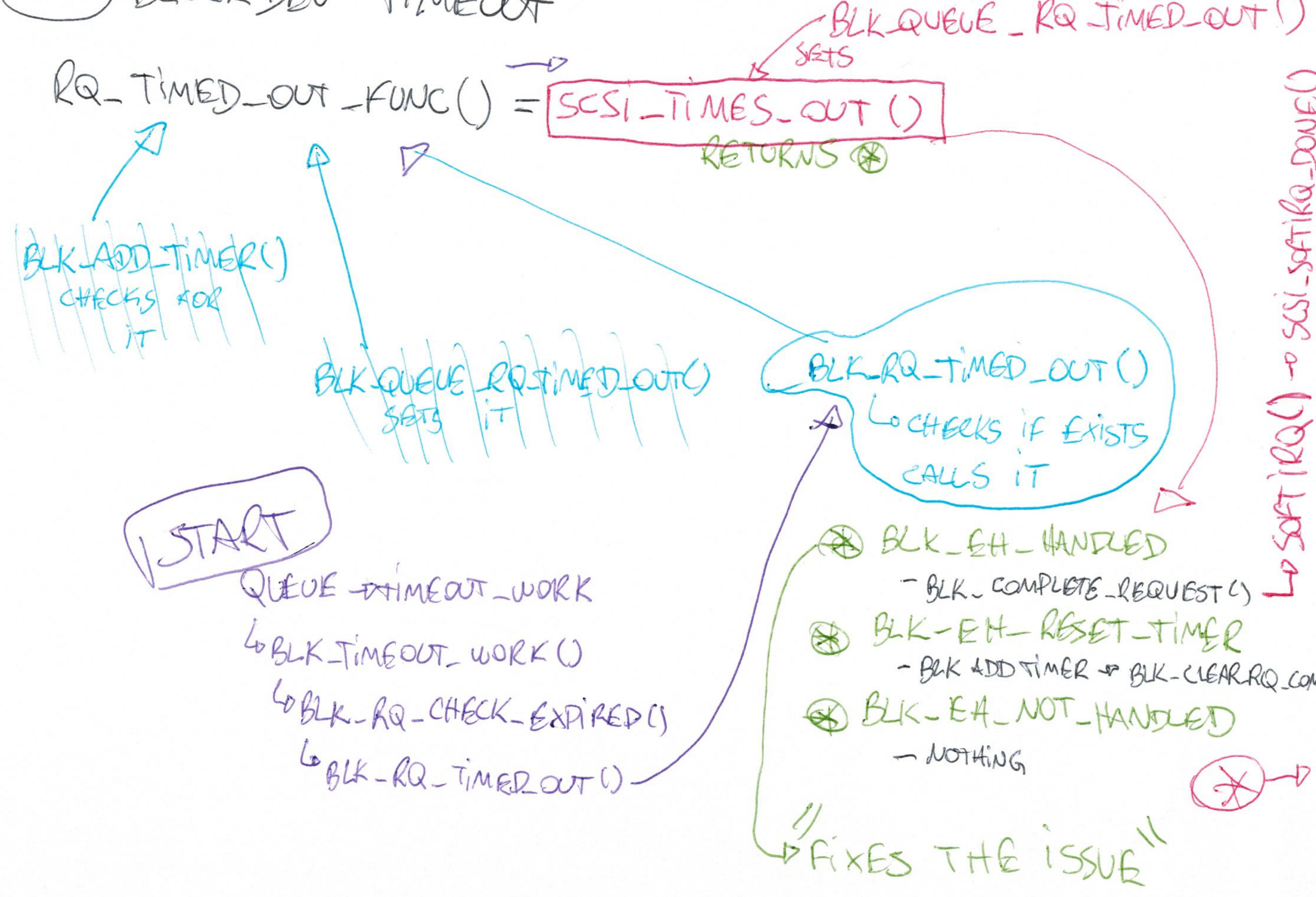
QUEUE → TIMEOUT_WORK
- ↳ BLK_TIMEOUT_WORK()
  - ↳ BLK_RQ_CHECK_EXPIRED()
    - ↳ BLK_RQ_TIMED_OUT()
      - ↳ Q → RQ_TIMED_OUT_FUNC()
        - ↳ COMPLETE_RQ → BLOCKSOFTIRQ
        - ↳ CLEAR_RQ_COMPLETE
          - ↳ RESETS TIMER ONLY

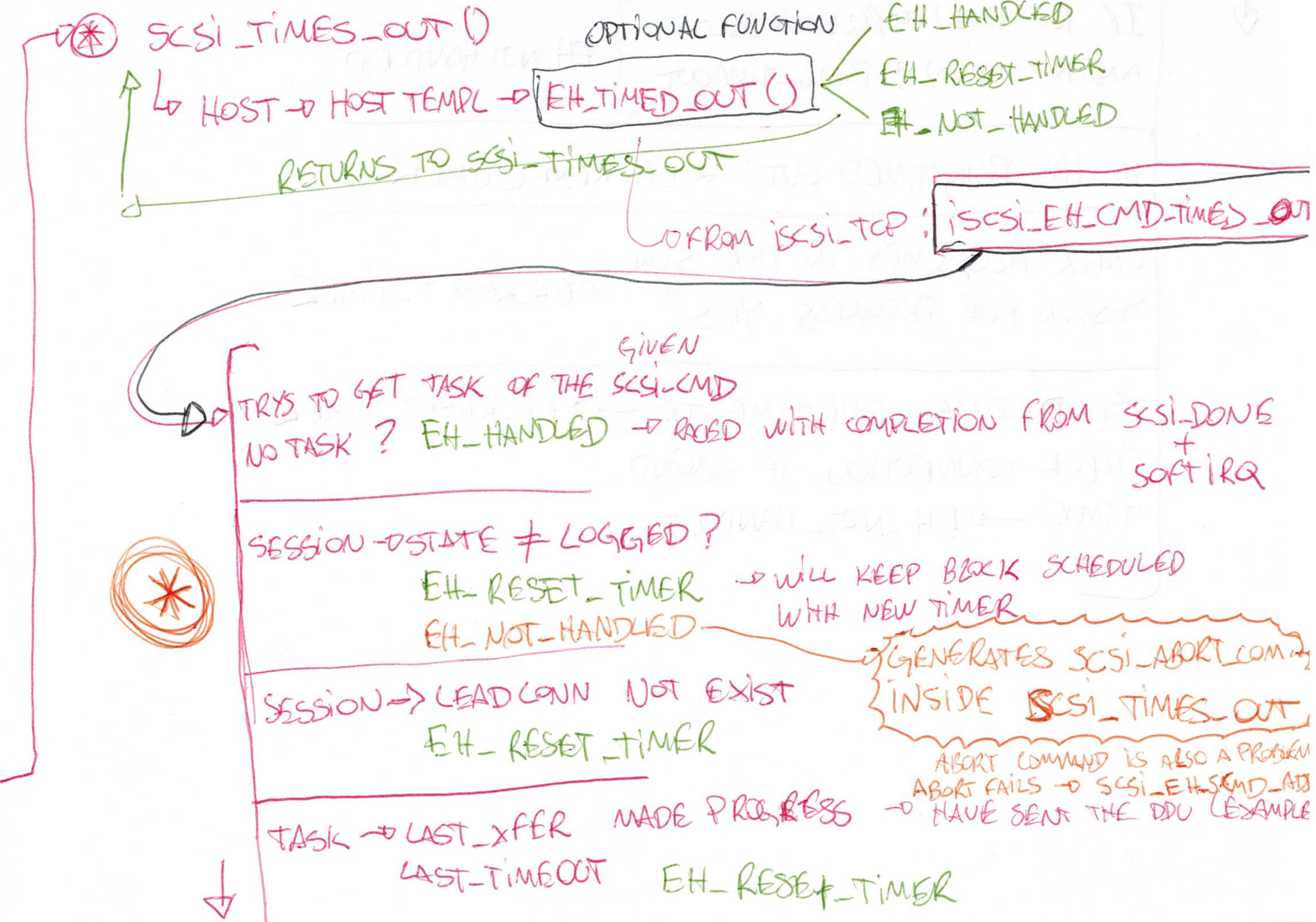
REQ QUEUE ⇒ SCSI_TIMES_OUT()

TIMEOUT FOR NORMAL SCSI_CMDS

BLOCKSOFTIRQ → COMPLETES I/O OR RE-SCHEDULES

#3 BLOCK DEV TIMEOUT




SCSI_TIMES_OUT()
HOST → HOST TEMPL → EH_TIMED_OUT()
OPTIONAL FUNCTION
EH_HANDLED
EH_RESET_TIMER
EH_NOT_HANDLED
RETURNS TO SCSI_TIMES_OUT
FROM ISCSI_TCP : ISCSI_EH_CMD_TIMED_OUT
GIVEN
TRYS TO GET TASK OF THE SCSI_CMD
NO TASK ? EH_HANDLED → RACED WITH COMPLETION FROM SCSI_DONE + SOFTIRQ
SESSION → STATE ≠ LOGGED ?
EH_RESET_TIMER → WILL KEEP BLOCK SCHEDULED WITH NEW TIMER
EH_NOT_HANDLED
GENERATES SCSI_ABORT_COM
INSIDE SCSI_TIMES_OUT
ABORT COMMAND IS ALSO A PROBLEM
ABORT FAILS → SCSI_EH_SCMD_ADD
SESSION → LEAD CONN NOT EXIST
EH_RESET_TIMER
TASK → LAST_XFER MADE PROGRESS → HAVE SENT THE PDU (EXAMPLE
LAST_TIMEOUT EH_RESET_TIMER

⇩

IF NO CONN → RECV_TIMEOUT
AND NO CONN → PING_TIMEOUT } EH_NOT HANDLED

IF HAS PING TIMED OUT → EH_RESET_TIMER

CHECK ALL CMDS IN THE SAME SESSION FOR PROGRESS. YES? → EH_RESET_TIMER

SENDS PING ONE TIME TO CHECK CONNECTION. → EH_RESET_TIMER
IF SECOND TIME → EH_NOT_HANDLED