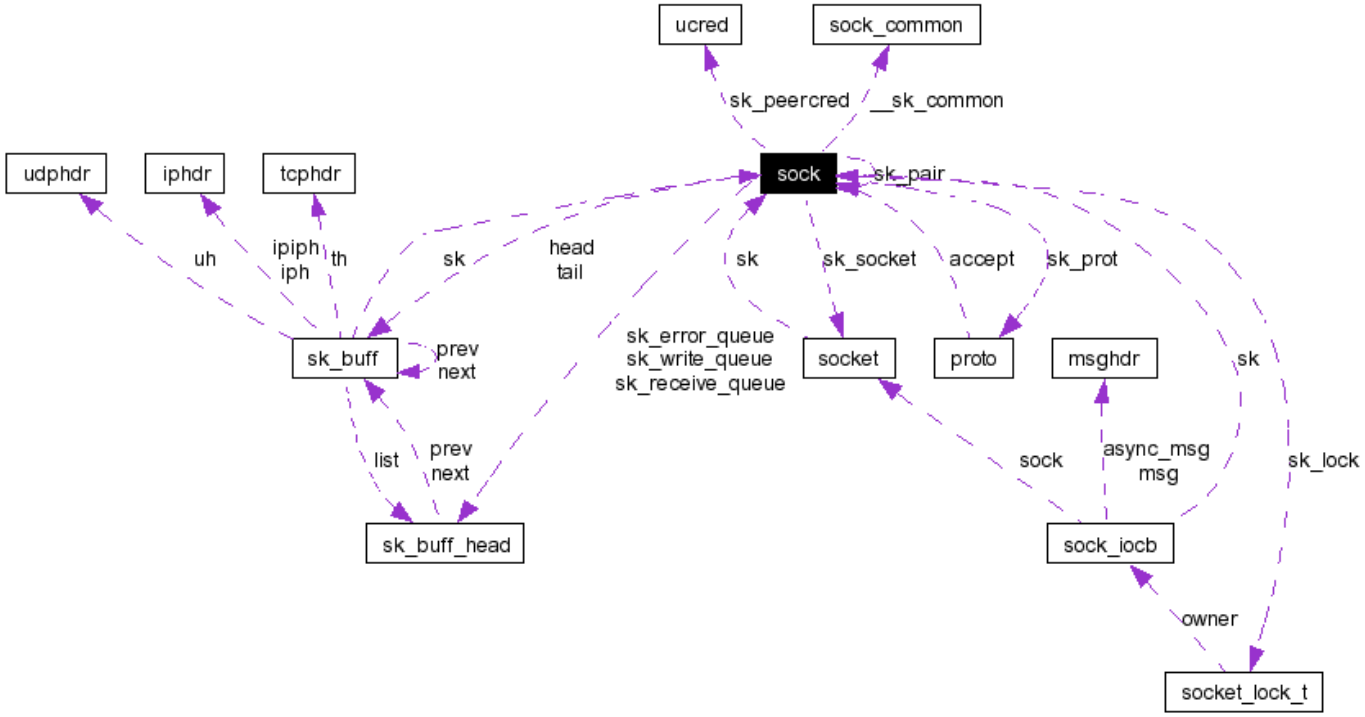
# sock Struct Reference

```
#include <sock.h>
```

Collaboration diagram for sock:




[legend]

## Data Fields

struct **sock_common** **__sk_common**
volatile unsigned char **sk_zapped**
unsigned char **sk_shutdown**
unsigned char **sk_use_write_queue**
unsigned char **sk_userlocks**
**socket_lock_t** **sk_lock**
int **sk_rcvbuf**
wait_queue_head_t* **sk_sleep**
struct dst_entry* **sk_dst_cache**
rwlock_t **sk_dst_lock**
struct xfrm_policy* **sk_policy** [2]
atomic_t **sk_rmem_alloc**
struct **sk_buff_head** **sk_receive_queue**
atomic_t **sk_wmem_alloc**
struct **sk_buff_head** **sk_write_queue**
atomic_t **sk_omem_alloc**
int **sk_wmem_queued**
int **sk_forward_alloc**
unsigned int **sk_allocation**
int **sk_sndbuf**
unsigned long **sk_flags**
char **sk_no_check**
unsigned char **sk_debug**
unsigned char **sk_rcvtstamp**
unsigned char **sk_no_largesend**
int **sk_route_caps**
unsigned long **sk_lingertime**
int **sk_hashent**
struct sock* **sk_pair**

```
struct {
  struct sk_buff*   head
  struct sk_buff*   tail
}                          sk_backlog
                 rwlock_t sk_callback_lock
    struct sk_buff_head sk_error_queue
            struct proto* sk_prot
                      int sk_err
                      int sk_err_soft
          unsigned short sk_ack_backlog
          unsigned short sk_max_ack_backlog
                    __u32 sk_priority
          unsigned short sk_type
           unsigned char sk_localroute
           unsigned char sk_protocol
             struct ucred sk_peercred
                      int sk_rcvlowat
                     long sk_rcvtimeo
                     long sk_sndtimeo
        struct sk_filter* sk_filter
                    void* sk_protinfo
       kmem_cache_t* sk_slab
        struct timer_list sk_timer
          struct timeval sk_stamp
           struct socket* sk_socket
                    void* sk_user_data
         struct module* sk_owner
                    void* sk_security
                void (* sk_state_change )(struct sock *sk)
                void (* sk_data_ready )(struct sock *sk, int bytes)
                void (* sk_write_space )(struct sock *sk)
                void (* sk_error_report )(struct sock *sk)
                  int (* sk_backlog_rcv )(struct sock *sk, struct sk_buff *skb)
                void (* sk_create_child )(struct sock *sk, struct sock *newsk)
                void (* sk_destruct )(struct sock *sk)
```

---

## Detailed Description

struct sock - network layer representation of sockets @__sk_common - shared layout with **tcp_tw_bucket** @sk_zapped - ax25 & ipx means !linked @sk_shutdown - mask of SEND_SHUTDOWN and/or RCV_SHUTDOWN @sk_use_write_queue - wheter to call sk->sk_write_space in sock_wfree @sk_userlocks - SO_SNDBUF and SO_RCVBUF settings @sk_lock - synchronizer @sk_rcvbuf - size of receive buffer in bytes @sk_sleep - sock wait queue @sk_dst_cache - destination cache @sk_dst_lock - destination cache lock @sk_policy - flow policy @sk_rmem_alloc - receive queue bytes committed @sk_receive_queue - incoming packets @sk_wmem_alloc - transmit queue bytes committed @sk_write_queue - Packet sending queue @sk_omem_alloc - "o" is "option" or "other" @sk_wmem_queued - persistent queue size @sk_forward_alloc - space allocated forward @sk_allocation - allocation mode @sk_sndbuf - size of send buffer in bytes @sk_flags - SO_LINGER (l_onoff), SO_BROADCAST, SO_KEEPALIVE, SO_OOBINLINE settings @sk_no_check - SO_NO_CHECK setting, wether or not checkup packets @sk_debug - SO_DEBUG setting @sk_rcvtstamp - SO_TIMESTAMP setting @sk_no_largesend - whether to sent large segments or not @sk_route_caps - route capabilities (e.g. NETIF_F_TSO) @sk_lingertime - SO_LINGER l_linger setting @sk_hashent - hash entry in several tables (e.g. tcp_ehash) @sk_pair - **socket** pair (e.g. AF_UNIX/unix_peer) @sk_backlog - always used with the per-**socket** spinlock held @sk_callback_lock - used with the callbacks in the end of this struct @sk_error_queue - rarely used @sk_prot - protocol handlers inside a network family @sk_err - last error @sk_err_soft - errors that don't cause failure but are the cause of a persistent failure not just 'timed out' @sk_ack_backlog - current listen backlog @sk_max_ack_backlog - listen backlog set in listen() @sk_priority - SO_PRIORITY setting @sk_type - **socket** type (SOCK_STREAM, etc) @sk_localroute - route locally only, SO_DONTROUTE setting @sk_protocol - which protocol this **socket** belongs in this network family @sk_peercred - SO_PEERCRED setting @sk_rcvlowat - SO_RCVLOWAT setting @sk_rcvtimeo - SO_RCVTIMEO setting @sk_sndtimeo - SO_SNDTIMEO setting @sk_filter - **socket** filtering instructions @sk_protinfo - private area, net family specific, when not using slab @sk_slab - the slabcache this instance was allocated from @sk_timer - sock cleanup timer @sk_stamp - time stamp of last packet received @sk_socket - Identd and reporting IO signals @sk_user_data - RPC and Tux layer private data @sk_owner - module that owns this **socket** @sk_state_change - callback to indicate change in the state of the sock @sk_data_ready - callback to indicate there is data to be processed @sk_write_space - callback to indicate there is bf sending space available @sk_error_report - callback to indicate errors (e.g. MSG_ERRQUEUE) @sk_create_child - callback to get new **socket** events @sk_backlog_rcv - callback to process the backlog @sk_destruct - called at sock freeing time, i.e. when all refcnt == 0

Definition at line **177** of file **sock.h**.

---

## Field Documentation

**struct sock_common sock::__sk_common**

Definition at line **182** of file **sock.h**.

**struct sk_buff* sock::head**

Definition at line **224** of file **sock.h**.

**unsigned short sock::sk_ack_backlog**

Definition at line **232** of file **sock.h**.

**unsigned int sock::sk_allocation**

Definition at line **207** of file **sock.h**.

**struct { ... } sock::sk_backlog**

**int(* sock::sk_backlog_rcv)(struct sock *sk, struct sk_buff *skb)**

**rwlock_t sock::sk_callback_lock**

Definition at line **227** of file **sock.h**.

**void(* sock::sk_create_child)(struct sock *sk, struct sock *newsk)**

**void(* sock::sk_data_ready)(struct sock *sk, int bytes)**

**unsigned char sock::sk_debug**

Definition at line **211** of file **sock.h**.

**void(* sock::sk_destruct)(struct sock *sk)**

**struct dst_entry * sock::sk_dst_cache**

Definition at line **197** of file **sock.h**.

**rwlock_t sock::sk_dst_lock**

Definition at line **198** of file **sock.h**.

**int sock::sk_err**

Definition at line **230** of file **sock.h**.

**int sock::sk_err_soft**

Definition at line **230** of file **sock.h**.

**struct sk_buff_head sock::sk_error_queue**

Definition at line **228** of file **sock.h**.

**void(* sock::sk_error_report)(struct sock *sk)**

**struct sk_filter * sock::sk_filter**

Definition at line **242** of file **sock.h**.

**unsigned long sock::sk_flags**

Definition at line **209** of file **sock.h**.

**int sock::sk_forward_alloc**

Definition at line **206** of file **sock.h**.

**int sock::sk_hashent**

Definition at line **216** of file **sock.h**.

**unsigned long sock::sk_lingertime**

Definition at line **215** of file **sock.h**.

**unsigned char sock::sk_localroute**

Definition at line **236** of file **sock.h**.

**socket_lock_t sock::sk_lock**

Definition at line **194** of file **sock.h**.

**unsigned short sock::sk_max_ack_backlog**

Definition at line **233** of file **sock.h**.

**char sock::sk_no_check**

Definition at line **210** of file **sock.h**.

**unsigned char sock::sk_no_largesend**

Definition at line **213** of file **sock.h**.

**atomic_t sock::sk_omem_alloc**

Definition at line **204** of file **sock.h**.

**struct module * sock::sk_owner**

Definition at line **249** of file **sock.h**.

**struct sock * sock::sk_pair**

Definition at line **217** of file **sock.h**.

**struct ucred sock::sk_peercred**

Definition at line **238** of file **sock.h**.

**struct xfrm_policy * sock::sk_policy**

Definition at line **199** of file **sock.h**.

**__u32 sock::sk_priority**

Definition at line **234** of file **sock.h**.

**struct proto * sock::sk_prot**

Definition at line **229** of file **sock.h**.

**void * sock::sk_protinfo**

Definition at line **243** of file **sock.h**.

**unsigned char sock::sk_protocol**

Definition at line **237** of file **sock.h**.

**int sock::sk_rcvbuf**

Definition at line **195** of file **sock.h**.

**int sock::sk_rcvlowat**

Definition at line **239** of file **sock.h**.

**long sock::sk_rcvtimeo**

Definition at line **240** of file **sock.h**.

**unsigned char sock::sk_rcvtstamp**

Definition at line **212** of file **sock.h**.

**struct sk_buff_head sock::sk_receive_queue**

Definition at line **201** of file **sock.h**.

**atomic_t sock::sk_rmem_alloc**

Definition at line **200** of file **sock.h**.

**int sock::sk_route_caps**

Definition at line **214** of file **sock.h**.

**void * sock::sk_security**

Definition at line **250** of file **sock.h**.

**unsigned char sock::sk_shutdown**

Definition at line **191** of file **sock.h**.

**kmem_cache_t * sock::sk_slab**

Definition at line **244** of file **sock.h**.

**wait_queue_head_t * sock::sk_sleep**

Definition at line **196** of file **sock.h**.

**int sock::sk_sndbuf**

Definition at line **208** of file **sock.h**.

**long sock::sk_sndtimeo**

Definition at line **241** of file **sock.h**.

**struct socket * sock::sk_socket**

Definition at line **247** of file **sock.h**.

**struct timeval sock::sk_stamp**

Definition at line **246** of file **sock.h**.

**void(* sock::sk_state_change)(struct sock *sk)**

**struct timer_list sock::sk_timer**

Definition at line **245** of file **sock.h**.

**unsigned short sock::sk_type**

Definition at line **235** of file **sock.h**.

**unsigned char sock::sk_use_write_queue**

Definition at line **192** of file **sock.h**.

**void * sock::sk_user_data**

Definition at line **248** of file **sock.h**.

**unsigned char sock::sk_userlocks**

Definition at line **193** of file **sock.h**.

**atomic_t sock::sk_wmem_alloc**

Definition at line **202** of file **sock.h**.

**int sock::sk_wmem_queued**

Definition at line **205** of file **sock.h**.

**struct sk_buff_head sock::sk_write_queue**

Definition at line **203** of file **sock.h**.

**void(* sock::sk_write_space)(struct sock *sk)**

**volatile unsigned char sock::sk_zapped**

Definition at line **190** of file **sock.h**.

**struct sk_buff* sock::tail**

Definition at line **225** of file **sock.h**.

---

The documentation for this struct was generated from the following file:

- **sock.h**

---