PX5 NET Symbols

Version Symbols

PX5_NET_MAJOR_VERSION
PX5_NET_MINOR_VERSION
PX5_NET_UPDATE_VERSION
PX5_NET_PATCH_VERSION

Error Checking Symbols

PX5_NET_FUNCTION_POINTER_VERIFY_ENABLE
PX5_NET_PARAMETER_CHECKING_DISABLE
PX5_PACKET_STRUCT_VERIFY_ENABLE

PX5 NET Configuration Symbols

PX5_DHCP_DISABLE
PX5_DHCP_REQUESTING_TIMEOUT
PX5_DHCP_SELECTING_TIMEOUT
PX5_DNS_DISABLE
PX5_DNS_MAX_HOST_NAME
PX5_DNS_MAX_IP_ADDRESSES
PX5_DNS_CACHE_SIZE
PX5_DNS_MAX_SERVERS
PX5_DNS_QUERY_TIMEOUT
PX5_DNS_MAX_RETRIES
PX5_IP_MAX_REASSEMBLY_TIMEOUT
PX5_NET_MAX_INTERFACES
PX5_NET_MAX_SOCKETS
PX5_NET_MAX_TCP_SOCKETS
PX5_NET_THREAD_STACK_SIZE
PX5_NET_TIMER_MS
PX5_NET_THREAD_PRIORITY
PX5_TCP_ACK_DELAY_TIMER_MS
PX5_TCP_INITIAL_RECEIVE_WINDOW
PX5_TCP_INITIAL_TIMEOUT
PX5_TCP_RETRY_LIMITS
PX5_TCP_TIMEOUT_LIMITS_MS

PX5 NET Error Code

PX5_NET_SUCCESS	0
PX5_NET_FAILURE	1
PX5_NET_ENTRY_NOT_FOUND	2
PX5_NET_CONTINUE	3
PX5_NET_UNKNOWN_OP	4
PX5_NET_INVALID_INTERFACE	5
PX5_NET_INTERFACE_NOT_READY	6
PX5_NET_INVALID_PARAM	7
PX5_NET_NO_MORE_SPACE	8
PX5_NET_NOT_STARTED	9
PX5_NET_NOT_CONFIGURED	10
PX5_NET_INVALID_SOCKET	11
PX5_NET_DUPLICATED_ENTRY	12
PX5_NET_NO_PACKETS	13
PX5_NET_NOT_ONLINK	14
PX5_NET_SOCKET_FAILED	15
PX5_NET_DRIVER_ERROR	16

PX5 NET Programmer's Reference Card

11440 West Bernardo Court, Suite 300, San Diego, CA 92127 | (858) 753-1715 | info@px5rtos.com

PX5 NET APIs

```
int     px5_net_create(u_int priority, u_char *stack_base, u_int stack_size);
void    px5_net_delete(void);
int     px5_net_default_gateway_set(u_long default_gateway_address);
int     px5_net_driver_buffer_get(u_char **memory_ptr, u_int *memory_size, u_int *handle);
void    px5_net_driver_buffer_put(u_char *buffer, u_int packet_handle);
void    px5_net_driver_data_received(u_int interface_id, u_char *buffer, u_int size,
                                     u_int packet_handle);
void    px5_net_driver_event_set(u_int interface_id, u_int events);
int     px5_net_driver_mac_address_get(u_int interface_id, u_char *mac_buffer,
                                       u_int *mac_length);
int     px5_net_driver_mac_address_set(u_int interface_id, u_char *mac_buffer,
                                       u_int mac_length);
int     px5_net_interface_change(u_int interface_id, u_int cause, u_long new_ipaddress);
void    px5_net_interface_change_callback_set(int (*cb)(u_int, u_int, u_int));
int     px5_net_interface_ip_address_set(u_int interface_id, u_long ip_addr,
                                         u_long subnet_mask);
int     px5_net_interface_physical_address_set(u_int interface_id,
                                               u_char *physical_addr, u_int physical_addr_length);
int     px5_net_interface_feature_set(int interface_id, u_int feature);
int     px5_net_interface_start(u_int interface_id, u_long ip_addr,
                                u_long subnet_mask);
int     px5_net_interface_stop(u_int interface_id);
```

BSD Sockets APIs

```
int     accept(int sock_fd, struct sockaddr *peer_addr, u_int *address_length);
int     bind(int sock_fd, struct sockaddr *local_address, int address_length);
int     close(int sock_fd);
int     connect(int sock_fd, struct sockaddr *peer_address, int address_length);
int     fcntl(int sock_fd, int command, ...);
void    freeaddrinfo(struct addrinfo *ai);
int     getaddrinfo(const char *node, const char *service,
                    const struct addrinfo *hints, struct addrinfo **res);
int     getpeername(int sock_fd, struct sockaddr *peer_address,
                    int *address_length);
int     getsockopt(int sock_fd, int level, int optname, void *optval, int *optlen);
int     listen(int sock_fd, int backlog);
int     read(int sock_fd, void *buffer, u_int bytes_to_received);
int     recv(int sock_fd, void *buffer, u_int bytes_read, int flags);
int     recv_zerocopy(int sock_fd, px5_packet_t *packet_ptr, int flags);
int     recvfrom(int sock_fd, void *bfufer, u_int bytes_received, int flags,
                 struct sockaddr *from_address, u_int *addrLength);
int     recvfrom_zerocopy(int sock_fd, px5_packet_t **return_packet_ptr, int flags,
                          struct sockaddr *from_addr, u_int *addrLength);
int     select(int maxFDs, fd_set *readFDs, fd_set *writeFDs, fd_set *exceptFDs,
               struct timeval *timeOut);
int     send(int sock_fd, void *buffer, u_int bytes_sent, int flags);
int     send_zerocopy(int sock_fd, px5_packet_t **return_packet_ptr, int flags);
int     sendto(int sock_fd, void *buffer, u_int bytes_sent, int flags,
               struct sockaddr *to_address, u_int addrLength);
int     sendto_zerocopy(int sock_fd, px5_packet_t *packet_ptr, int flags,
                        struct sockaddr *to_addr, u_int addrLength);
int     setsockopt(int sock_fd, int level, int optname, const void *optval, int optlen);
int     shutdown(int sock_fD, int how);
int     socket(int family, int type, int protocol);
int     write(int sock_fd, void *buffer, u_int bytes_to_write);
void    FD_CLR(int fd, px5_fd_set *fdset);
int     FD_ISSET(int fd, px5_fd_set *fdset);
void    FD_SET(int fd, px5_fd_set *fdset);
void    FD_ZERO(px5_fd_set *fdset);
```

Packet APIs

```
int     px5_packetpool_create(u_char *memory_ptr, u_int memory_size,
            u_int num_main_packets, u_int main_packet_size, u_int num_aux_packets,
                u_int aux_packet_size, u_int alignment, u_int *bytes_allocated);
void    px5_packet_put(px5_packet_t *packet_ptr);
        px5_packet_t *px5_packet_get(u_int pool_type);
int     px5_packet_payload_length_set(px5_packet_t* packet_ptr, u_int length);
int     px5_packet_buffer_and_length_get(px5_packet_t* packet_ptr,
                                         u_char **buffer_ptr, u_int *length);
int     px5_packet_payload_memory_get(px5_packet_t* packet_ptr, int sock_fd,
            u_long dest_ip, u_char **payload_memory, u_int *memory_size);
```

DHCP APIs

```
int     px5_dhcp_release(u_int interface_id);
int     px5_dhcp_parameter_requests_set(u_char *parameters, int count);
int     px5_dhcp_parameter_parse_callback_set(int (*cb)(u_int, u_char, int, char*));
```

DNS APIs

```
int     px5_dns_start(u_long dns_server);
int     px5_dns_server_add(u_long dns_server);
int     px5_dns_server_remove(u_long dns_server);
int     px5_dns_stop(void);
int     px5_dns_address_count_get(u_char *hostname, int hostname_length, u_int
                                  *num_ip_addresses);
int     px5_dns_entry_info_get(u_char *hostname, u_int hostname_length, u_int
                               index, u_long *ip_address, u_int *timeout);
int     px5_dns_lookup(u_char *hostname, u_int hostname_length,
                       u_long *ip_address);
```

Ethernet APIs

```
int     px5_ethernet_initialize(u_int interface_id, px5_ethernet_t *eth_mac,
                    void *arp_memory, u_int arp_memory_size,
                u_char *mac_address, u_int mac_addr_len,
            int (*driver)(u_int interface_id, struct px5_net_driver_info_s *driver_info));
```