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

System Configuration

PX5_IP_MAX_REASSEMBLY_TIMEOUT
PX5_NET_MAX_INTERFACES
PX5_NET_THREAD_STACK_SIZE
PX5_NET_THREAD_PRIORITY
PX5_NET_TIMER_FREQ
PX5_NET_GLOBALS_EXTERNAL
PX5_RAND

ARP Configuration

PX5_ARP_REQUEST_RETRY_MAX
PX5_ARP_CACHE_ENTRY_TIMEOUT_MAX
PX5_ARP_REQUEST_TIMEOUT_VALUE
PX5_ARP_TRANSMIT_QUEUE_MAX_DEFAULT

IPv4 Configuration

PX5_NET_IP_FRAGMENT_ENABLE
PX5_NET_STATIC_ROUTING_ENABLE
PX5_NET_MAX_STATIC_ROUTING_ENTRIES
PX5_IPV4_MULTICAST_ENABLE
PX5_IPV4_MAX_MULTICAST_GROUPS

IPv6 Configuration

PX5_IPV6_ENABLE
PX5_IPV6_MAX_ADDRESSES_PER_INTERFACE
PX5_IPV6_MAX_MULTICAST_ADDRESSES
PX5_IPV6_MAX_DEFAULT_ROUTERS
PX5_IPV6_PREFIX_MAX_SIZE
PX5_IPV6_DESTINATION_TABLE_SIZE
PX5_IPV6_DELAY_FIRST_PROBE_TIME
PX5_IPV6_DUPADDRDETECTTRANSMITS
PX5_IPV6_RETRANS_TIMER
PX5_IPV6_REACHABLE_TIME
PX5_IPV6_RTR_SOLICITATION_INTERVAL
PX5_IPV6_MAX_RTR_SOLICITATIONS
PX5_IPV6_MAX_MULTICAST_SOLICIT
PX5_IPV6_MAX_UNICAST_SOLICIT
PX5_IPV6_MAX_NEIGHBOR_ADVERTISEMENT
PX5_IPV6_RANDOM_DELAY_IN_MS
PX5_IPV6_MINIMUM_MTU
PX5_ND_CACHE_MAX_SIZE

Socket Configuration

PX5_NET_DYNAMIC_PORT_START
PX5_NET_MAX_SOCKETS
PX5_NET_MAX_TCP_SOCKETS
PX5_NET_SOCKET_EVENT_CALLBACK_ENABLE
PX5_NET_RAW_SOCKET_ENABLE

TCP Configuration

PX5_TCP_ACK_DELAY_TIMER_MS
PX5_TCP_INITIAL_RECEIVE_WINDOW
PX5_TCP_ACK_TIMEOUT
PX5_TCP_RETRY_LIMITS
PX5_NET_TCP_DEFAULT_IDLE_TIMEOUT

DHCP Configuration

PX5_DHCP_DISABLE
PX5_MAX_DHCP_INTERFACES

DNS Configuration

PX5_DNS_DISABLE
PX5_DNS_MAX_HOST_NAME
PX5_DNS_MAX_IP_ADDRESSES
PX5_DNS_MAX_IPV6_ADDRESSES
PX5_DNS_CACHE_SIZE
PX5_DNS_MAX_SERVERS
PX5_DNS_QUERY_TIMEOUT
PX5_DNS_MAX_RETRIES

SNTP Configuration

PX5_SNTP_ENABLE
PX5_SNTP_PORT
PX5_SNTP_SERVER_TIMEOUT

PX5 NET Error Codes

PX5_NET_SUCCESS	PX5_SUCCESS
PX5_NET_FAILURE	(5001U)
PX5_NET_ENTRY_NOT_FOUND	(5002U)
PX5_NET_CONTINUE	(5003U)
PX5_NET_UNKNOWN_OP	(5004U)
PX5_NET_INVALID_INTERFACE	(5005U)
PX5_NET_INTERFACE_NOT_READY	(5006U)
PX5_NET_INVALID_PARAM	(5007U)
PX5_NET_NO_SPACE	(5008U)
PX5_NET_NOT_STARTED	(5009U)
PX5_NET_NOT_CONFIGURED	(5010U)
PX5_NET_INVALID_SOCKET	(5011U)
PX5_NET_DUPLICATED_ENTRY	(5012U)
PX5_NET_NO_PACKETS	(5013U)
PX5_NET_NOT_ONLINK	(5014U)
PX5_NET_SOCKET_FAILED	(5015U)
PX5_NET_DRIVER_ERROR	(5016U)
PX5_NET_FEATURE_UNSUPPORTED	(5017U)

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

```
void    px5_net_delete(void);
int     px5_net_default_gateway_set(u_int4 default_gateway_address);
int     px5_net_create(u_int priority, u_char *stack_base, size_t stack_size);
int     px5_net_interface_feature_set(u_int interface_id, u_int feature);
int     px5_net_interface_ip_address_get(u_int interface_id, u_int4 *ip_address,
                                         u_int4 *subnet_mask);
int     px5_net_interface_ip_address_set(u_int interface_id, u_int4 ip_addr, u_int4 subnet_mask);
int     px5_net_interface_mtu_set(u_int interface_id, u_int ip_mtu);
int     px5_net_interface_physical_address_set(u_int interface_id, u_char *physical_addr,
                                               u_int physical_addr_length);
int     px5_net_interface_start(u_int interface_id, u_int4 ip_addr, u_int4 subnet_mask);
int     px5_net_interface_stop(u_int interface_id);
int     px5_net_ping4(u_int dest_ip, u_char *ping_data, u_int data_size, u_int num_pings,
                      u_int wait_time, u_int *pings_sent, u_int *pings_received);
int     px5_net_ipv4_static_route_clear(u_int4 dest_network);
int     px5_net_ipv4_static_route_set(u_int4 dest_network, u_int4 dest_netmask, u_int4 next_hop);
```

PX5 NET Driver APIs

```
int     px5_net_driver_buffer_get(u_char **buffer_ptr, u_int *buffer_size, void **handle);
void    px5_net_driver_buffer_put(u_char *buffer, void *handle);
int     px5_net_driver_data_received(u_int interface_id, u_char *buffer, u_int size, void *handle);
int     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_driver_mac_address_size_set(u_int interface_id, u_int mac_address_size);
```

ARP APIs

```
void    px5_net_arp_transmit_queue_max_set(int interface_id, u_short max_queue_length);
```

IPv6 APIs

```
int     px5_ipv6_interface_address_set(u_int interface_id, u_char *ipv6_addr, int prefix_length,
                                       int state);
```

BSD Sockets APIs

```
int     accept(int sock_fd, struct sockaddr *peer_addr, size_t *address_length);
int     bind(int sock_fd, struct sockaddr *local_address, size_t address_length);
int     close(int sock_fd);
int     connect(int sock_fd, struct sockaddr *peer_address, size_t 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, size_t *address_length);
int     getsockname(int sock_fd, struct sockaddr *localAddr, size_t *addrLength);
int     getsockopt(int sock_fd, int level, int optname, void *optval, size_t *optlen);
int     inet_pton(int af, const char *src, void *sock_addr);
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, size_t bytes_read, int flags);
int     recv_zerocopy(int sock_fd, px5_packet_t **packet_ptr, int flags);
int     recvfrom(int sock_fd, void *buffer, size_t bytes_received, int flags, struct sockaddr *from_address,
                 size_t *addrLength);
int     recvfrom_zerocopy(int sock_fd, px5_packet_t **return_packet_ptr, int flags,
                          struct sockaddr *from_addr, size_t *addrLength);
int     select(int maxFDs, fd_set *readFDs, fd_set *writeFDs, fd_set *exceptFDs, struct timeval *timeOut);
int     send(int sock_fd, void *buffer, size_t bytes_sent, int flags);
int     send_zerocopy(int sock_fd, px5_packet_t *packet_ptr, int flags);
int     sendto(int sock_fd, void *buffer, size_t bytes_sent, int flags, struct sockaddr *to_address,
               size_t addrLength);
int     sendto_zerocopy(int sock_fd, px5_packet_t *packet_ptr, int flags, struct sockaddr *to_addr,
                        size_t addrLength);
int     setsockopt(int sock_fd, int level, int optname, const void *optval, size_t 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);
```

BSD Sockets APIs (continued)

```
void    FD_ZERO(px5_fd_set *fdset);
int     px5_socket_recv_data_size_get(int sock_fd, u_int *bytes);
int     px5_socket_event_callback_set(int sock_fd, void (*cb)(int sock_fd, int reason), int mask);
```

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 *buf_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_payload_area_get(px5_packet_t* packet_ptr, u_char **buffer_ptr, u_int *length);
int     px5_packet_payload_area_config(px5_packet_t *packet_ptr, int sock_fd,
                                       u_char **payload_memory, u_int *memory_size);
int     px5_packet_payload_retrieve(px5_packet_t *packet_ptr, u_char *buffer_ptr,
                                    u_int *payload_size);
int     px5_packet_payload_set(px5_packet_t *packet_ptr, u_char *payload_ptr, u_int payload_size);
```

DHCP APIs

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

DNS APIs

```
int     px5_dns_start(u_int4 dns_server);
int     px5_dns_server_add(u_int4 dns_server);
int     px5_dns_server_remove(u_int4 dns_server);
int     px5_dns_stop(void);
int     px5_dns_address_count_get(u_char *hostname, u_int hostname_length, u_int address_type,
                                  u_int *num_ip_addresses);
int     px5_dns_entry_info_get(u_char *hostname, u_int hostname_length, u_int index,
                               u_int address_type, u_int4 *ip_address, u_int address_size, u_int *timeout);
void    px5_dns_entry_remove(u_char *hostname, u_int hostname_length);
int     px5_dns_lookup(u_char *hostname, u_int hostname_length, u_int address_type,
                       u_int4 *ip_address, u_int address_size);
```

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));
```

SNTP APIs

```
int     px5_sntp_client_start(u_int4 ip_address, u_int time_sync_interval,
                              void (*time_update_callback)(struct timespec local_time),
                              void (*unsuitable_server_callback)(int reason, int info));
int     px5_sntp_client_server_change(u_int4 ip_address, u_int time_sync_interval);
int     px5_sntp_client_local_time_get(struct timespec *time_ptr);
```