

ezManager Library

# 사용자 설명서

Revision 1.0

Sollae Systems Co., Ltd.

<http://www.sollae.co.kr>

## 목차

<b>1</b>	<b>소개.....</b>	<b>- 0 -</b>
1.1	소개.....	- 0 -
1.2	표기법 .....	- 0 -
1.3	주의사항 .....	- 0 -
<b>2</b>	<b>라이브러리 시작과 종료.....</b>	<b>- 0 -</b>
2.1	Start_Library .....	- 0 -
2.2	End_Library.....	- 0 -
<b>3</b>	<b>제품(EZTCP) 검색.....</b>	<b>- 0 -</b>
3.1	EzTCP_Search.....	- 0 -
3.2	EzTCP_Read .....	- 1 -
3.3	get_eztcp_count .....	- 4 -
3.4	get_eztcp_info_by_index.....	- 5 -
<b>4</b>	<b>설정 값 읽기.....</b>	<b>- 0 -</b>
4.1	네트워크 .....	- 0 -
4.1.1	get_arp.....	- 0 -
4.1.2	get_dhcp.....	- 0 -
4.1.3	get_dhcp_dns.....	- 1 -
4.1.4	get_pppoe.....	- 2 -
4.1.5	get_ip4_local_address.....	- 2 -
4.1.6	get_ip4_subnet_mask.....	- 3 -
4.1.7	get_ip4_gateway_address.....	- 4 -
4.1.8	get_ip4_dns_address.....	- 5 -
4.1.9	get_pppoe_id.....	- 6 -
4.1.10	get_pppoe_pwd.....	- 7 -
4.1.11	get_ip6.....	- 8 -
4.1.12	get_ip6_gua_type.....	- 8 -
4.1.13	get_ip6_eui_type.....	- 10 -
4.1.14	get_ip6_local_address.....	- 11 -
4.1.15	get_ip6_subnet_prefix_length.....	- 12 -
4.1.16	get_ip6_gateway_address.....	- 13 -
4.1.17	get_ip6_dns_address.....	- 14 -
4.2	시리얼 포트 .....	- 15 -
4.2.1	get_uart_count.....	- 15 -
4.2.2	get_uart_min_baudrate .....	- 16 -
4.2.3	get_uart_max_baudrate.....	- 16 -
4.2.4	get_uart_serial_type .....	- 17 -
4.2.5	get_uart_ttl .....	- 18 -
4.2.6	get_uart_baudrate .....	- 18 -
4.2.7	get_uart_parity .....	- 19 -
4.2.8	get_uart_databit .....	- 20 -
4.2.9	get_uart_stopbit.....	- 21 -
4.2.10	get_uart_flow_control.....	- 22 -
4.2.11	get_uart_dtrdsr .....	- 23 -
4.2.12	get_uart_tx_delay .....	- 23 -
4.3	TCP/IP .....	- 24 -
4.3.1	get_uart_communication_mode.....	- 24 -
4.3.2	get_uart_peer_address .....	- 25 -

4.3.3	get_uart_peer_port.....	- 26 -
4.3.4	get_uart_local_port.....	- 27 -
4.3.5	get_uart_cod_tcp_server.....	- 28 -
4.3.6	get_uart_watermark.....	- 29 -
4.3.7	get_uart_timeout.....	- 31 -
4.3.8	get_uart_data_frame_interval.....	- 32 -
4.3.9	get_uart_separator_length.....	- 33 -
4.3.10	get_uart_separator_type.....	- 34 -
4.3.11	get_uart_separator.....	- 35 -
4.3.12	get_uart_tcp_nodelay.....	- 36 -
4.3.13	get_uart_rfc2217.....	- 37 -
4.3.14	get_uart_protocol.....	- 38 -
4.4	CSC-HR2.....	- 39 -
4.4.1	get_csc_hr2_communication_mode.....	- 39 -
4.4.2	get_csc_hr2_id.....	- 39 -
4.4.3	get_csc_hr2_network_timeout.....	- 40 -
4.4.4	get_csc_hr2_network_threshold.....	- 41 -
4.4.5	get_csc_hr2_server_timeout.....	- 42 -
4.4.6	get_csc_hr2_server_threshold.....	- 43 -
4.4.7	get_csc_hr2_check_port.....	- 44 -
4.4.8	get_csc_hr2_first_server_address.....	- 45 -
4.4.9	get_csc_hr2_first_server_port.....	- 47 -
4.4.10	get_csc_hr2_second_server_address.....	- 48 -
4.4.11	get_csc_hr2_second_server_port.....	- 49 -
4.5	I/O 제품.....	- 50 -
4.5.1	get_io_http.....	- 50 -
4.5.2	get_io_http_port.....	- 50 -
4.5.3	get_io_html_size.....	- 51 -
4.5.4	get_io_modbus.....	- 53 -
4.5.5	get_io_input_notification.....	- 53 -
4.5.6	get_io_output_automatic_initialize.....	- 54 -
4.5.7	get_io_modbus_type.....	- 55 -
4.5.8	get_io_unit_id.....	- 56 -
4.5.9	get_io_input_address.....	- 57 -
4.5.10	get_io_output_address.....	- 58 -
4.5.11	get_io_poll_interval.....	- 59 -
4.5.12	get_io_slave_control_type.....	- 60 -
4.5.13	get_io_master_control_type.....	- 61 -
4.5.14	get_io_tcp_type.....	- 62 -
4.5.15	get_io_multi_connection_number.....	- 64 -
4.5.16	get_io_modbus_peer_address.....	- 64 -
4.5.17	get_io_modbus_peer_port.....	- 65 -
4.5.18	get_io_modbus_local_port.....	- 66 -
4.5.19	get_io_output_number.....	- 68 -
4.5.20	get_io_input_number.....	- 68 -
4.5.21	get_io_event_notification.....	- 69 -
4.5.22	get_event_notification_email.....	- 69 -
4.5.23	get_io_event_notification_port.....	- 71 -
4.5.24	get_io_input_valid_time.....	- 72 -
4.5.25	get_io_macro_type.....	- 74 -
4.5.26	get_io_macro.....	- 74 -
4.5.27	get_io_port_macro.....	- 75 -
4.5.28	get_io_macro_text.....	- 77 -
4.5.29	get_io_output_delay.....	- 78 -
4.5.30	get_io_output_initial_state.....	- 79 -
4.5.31	get_io_comment.....	- 80 -
4.6	무선랜.....	- 82 -

4.6.1	get_wlan_type	- 82 -
4.6.2	get_wlan_channel	- 83 -
4.6.3	get_wlan_ssid	- 84 -
4.6.4	get_wlan_antenna	- 85 -
4.6.5	get_wlan_phy_mode	- 87 -
4.6.6	get_wlan_short_preamble	- 88 -
4.6.7	get_wlan_short_slot	- 89 -
4.6.8	get_wlan_cts_protection	- 90 -
4.6.9	get_wlan_background_scan	- 92 -
4.6.10	get_wlan_encryption_type	- 93 -
4.6.11	get_wlan_authentication_type	- 94 -
4.6.12	get_wlan_wep_key_length	- 96 -
4.6.13	get_wlan_wep_key_index	- 98 -
4.6.14	get_wlan_wep_key_data_type	- 99 -
4.6.15	get_wlan_wep_key	- 99 -
4.6.16	get_wlan_max_wpa_passphrase_length	- 101 -
4.6.17	get_wlan_wpa_passphrase	- 102 -
4.6.18	get_wlan_shared_key	- 103 -
4.6.19	get_wlan_wpa_enterprise	- 104 -
4.6.20	get_wlan_wpa_enterprise_id	- 106 -
4.6.21	get_wlan_wpa_enterprise_pwd	- 107 -
4.7	부가기능	- 109 -
4.7.1	get_telnet	- 109 -
4.7.2	get_send_mac_address	- 109 -
4.7.3	get_ssl	- 110 -
4.7.4	get_ssh	- 110 -
4.7.5	get_ip4_address_search	- 111 -
4.7.6	get_remote_debug	- 112 -
4.7.7	get_tcp_multi_connection	- 112 -
4.7.8	get_power_management	- 113 -
4.7.9	get_comment	- 114 -
4.7.10	get_allowed_ezmanager	- 115 -
4.7.11	get_allowed_mac_address	- 116 -
4.7.12	get_allowed_ip4_address	- 117 -
4.7.13	get_allowed_ip4_network_mask_type	- 118 -
4.7.14	get_allowed_ip6_address	- 120 -
4.7.15	get_allowed_ip6_subnet_prefix_length	- 121 -
4.7.16	get_ip4_change_notification_type	- 123 -
4.7.17	get_ip4_change_notification_data_type	- 124 -
4.7.18	get_ip4_change_notification_interval	- 125 -
4.7.19	get_ip4_change_notification_peer_port	- 127 -
4.7.20	get_ip4_change_notification_peer_address	- 128 -
4.7.21	get_ip4_change_notification_ddns_id	- 129 -
4.7.22	get_ip4_change_notification_ddns_pwd	- 130 -
5	설정 값 변경	- 0 -
5.1	네트워크	- 0 -
5.1.1	set_arp	- 0 -
5.1.2	set_dhcp	- 0 -
5.1.3	set_dhcp_dns	- 1 -
5.1.4	set_pppoe	- 2 -
5.1.5	set_ip4_local_address	- 3 -
5.1.6	set_ip4_subnet_mask	- 4 -
5.1.7	set_ip4_gateway_address	- 5 -
5.1.8	set_ip4_dns_address	- 6 -
5.1.9	set_pppoe_id	- 7 -
5.1.10	set_pppoe_pwd	- 8 -

5.1.11	set_ip6.....	- 8 -
5.1.12	set_ip6_gua_type .....	- 9 -
5.1.13	set_ip6_eui_type .....	- 10 -
5.1.14	set_ip6_local_address.....	- 11 -
5.1.15	set_ip6_subnet_prefix_length .....	- 12 -
5.1.16	set_ip6_gateway_address .....	- 13 -
5.1.17	set_ip6_dns_address .....	- 14 -
5.2	시리얼 포트 .....	- 15 -
5.2.1	set_uart_serial_type.....	- 15 -
5.2.2	set_uart_ttl.....	- 16 -
5.2.3	set_uart_baudrate.....	- 17 -
5.2.4	set_uart_parity.....	- 18 -
5.2.5	set_uart_databit.....	- 19 -
5.2.6	set_uart_stopbit .....	- 20 -
5.2.7	set_uart_flow_control.....	- 21 -
5.2.8	set_uart_dtrdsr.....	- 23 -
5.2.9	set_uart_tx_delay.....	- 24 -
5.3	TCP/IP .....	- 25 -
5.3.1	set_uart_communication_mode .....	- 25 -
5.3.2	set_uart_peer_address.....	- 26 -
5.3.3	set_uart_peer_port .....	- 28 -
5.3.4	set_uart_local_port.....	- 28 -
5.3.5	set_uart_cod_tcp_server.....	- 30 -
5.3.6	set_uart_watermark .....	- 31 -
5.3.7	set_uart_timeout .....	- 32 -
5.3.8	set_uart_data_frame_interval.....	- 33 -
5.3.9	set_uart_separator_length.....	- 34 -
5.3.10	set_uart_separator_type .....	- 35 -
5.3.11	set_uart_separator.....	- 36 -
5.3.12	set_uart_tcp_nodelay.....	- 37 -
5.3.13	set_uart_rfc2217.....	- 38 -
5.3.14	set_uart_protocol.....	- 40 -
5.4	CSC-HR2.....	- 41 -
5.4.1	set_csc_hr2_communication_mode .....	- 41 -
5.4.2	set_csc_hr2_id.....	- 42 -
5.4.3	set_csc_hr2_network_timeout.....	- 43 -
5.4.4	set_csc_hr2_network_threshold.....	- 43 -
5.4.5	set_csc_hr2_server_timeout .....	- 44 -
5.4.6	set_csc_hr2_server_threshold .....	- 45 -
5.4.7	set_csc_hr2_checkport.....	- 46 -
5.4.8	set_csc_hr2_first_server_address.....	- 47 -
5.4.9	set_csc_hr2_first_server_port .....	- 48 -
5.4.10	set_csc_hr2_second_server_address .....	- 49 -
5.4.11	set_csc_hr2_second_server_port.....	- 49 -
5.5	I/O 제품.....	- 50 -
5.5.1	set_io_http.....	- 50 -
5.5.2	set_io_http_port.....	- 51 -
5.5.3	set_io_html_size.....	- 52 -
5.5.4	set_io_modbus .....	- 53 -
5.5.5	set_io_input_notification .....	- 54 -
5.5.6	set_io_output_automatic_initialize.....	- 55 -
5.5.7	set_io_modbus_type.....	- 56 -
5.5.8	set_io_unit_id .....	- 57 -
5.5.9	set_io_input_address .....	- 58 -
5.5.10	set_io_output_address .....	- 59 -
5.5.11	set_io_poll_interval .....	- 60 -
5.5.12	set_io_slave_control_type.....	- 61 -

5.5.13	set_io_master_control_type	- 62 -
5.5.14	set_io_tcp_type	- 63 -
5.5.15	set_io_multi_connection_number	- 64 -
5.5.16	set_io_modbus_peer_address	- 65 -
5.5.17	set_io_modbus_peer_port	- 66 -
5.5.18	set_io_modbus_local_port	- 67 -
5.5.19	set_io_event_notification	- 68 -
5.5.20	set_event_notification_email	- 69 -
5.5.21	set_io_event_notification_port	- 70 -
5.5.22	set_io_input_valid_time	- 71 -
5.5.23	set_io_macro	- 72 -
5.5.24	set_io_port_macro	- 73 -
5.5.25	set_io_macro_text	- 74 -
5.5.26	set_io_output_delay	- 75 -
5.5.27	set_io_output_initial_state	- 76 -
5.5.28	set_io_comment	- 77 -
5.6	무선랜	- 78 -
5.6.1	set_wlan_type	- 78 -
5.6.2	set_wlan_channel	- 80 -
5.6.3	set_wlan_ssid	- 80 -
5.6.4	set_wlan_antenna	- 81 -
5.6.5	set_wlan_phy_mode	- 82 -
5.6.6	set_wlan_short_preamble	- 84 -
5.6.7	set_wlan_short_slot	- 85 -
5.6.8	set_wlan_cts_protection	- 86 -
5.6.9	set_wlan_background_scan	- 87 -
5.6.10	set_wlan_encryption_type	- 88 -
5.6.11	set_wlan_authentication_type	- 89 -
5.6.12	set_wlan_wep_key_length	- 91 -
5.6.13	set_wlan_wep_key_index	- 92 -
5.6.14	set_wlan_wep_key_data_type	- 93 -
5.6.15	set_wlan_wep_key	- 94 -
5.6.16	set_wlan_wpa_passphrase	- 95 -
5.6.17	set_wlan_shared_key	- 96 -
5.6.18	set_wlan_wpa_enterprise	- 97 -
5.6.19	set_wlan_wpa_enterprise_id	- 99 -
5.6.20	set_wlan_wpa_enterprise_pwd	- 100 -
5.7	부가기능	- 101 -
5.7.1	set_telnet	- 101 -
5.7.2	set_send_mac_address	- 101 -
5.7.3	set_ssl	- 102 -
5.7.4	set_ssh	- 104 -
5.7.5	set_ip4_address_search	- 105 -
5.7.6	set_remote_debug	- 106 -
5.7.7	set_tcp_multi_connection	- 107 -
5.7.8	set_power_management	- 108 -
5.7.9	set_comment	- 109 -
5.7.10	set_allowed_mac_address	- 109 -
5.7.11	set_allowed_ip4_address	- 110 -
5.7.12	set_allowed_ip4_network_mask_type	- 111 -
5.7.13	set_allowed_ezmanager	- 112 -
5.7.14	set_allowed_ip6_address	- 113 -
5.7.15	set_allowed_ip6_subnet_prefix_length	- 114 -
5.7.16	set_ip4_change_notification_type	- 115 -
5.7.17	set_ip4_change_notification_data_type	- 116 -
5.7.18	set_ip4_change_notification_interval	- 117 -
5.7.19	set_ip4_change_notification_peer_port	- 118 -

5.7.20	<i>set_ip4_change_notification_peer_address</i> .....	- 118 -
5.7.21	<i>set_ip4_change_notification_ddns_id</i> .....	- 119 -
5.7.22	<i>set_ip4_change_notification_ddns_pwd</i> .....	- 120 -
<b>6</b>	<b>설정 값 저장</b> .....	<b>- 0 -</b>
6.1	EzTCP_Write.....	- 0 -
6.2	EzTCP_Initialize.....	- 3 -
<b>7</b>	<b>비밀번호 관리</b> .....	<b>- 0 -</b>
7.1	EzTCP_ChangePassword .....	- 0 -
<b>8</b>	<b>현재상태 보기</b> .....	<b>- 0 -</b>
8.1	EzTCP_Status .....	- 0 -
8.2	get_status_string_length .....	- 1 -
8.3	get_status_string .....	- 1 -
8.4	get_tcpip4_session_count .....	- 3 -
8.5	get_tcpip6_session_count .....	- 5 -
8.6	get_tcpip4_session_info.....	- 6 -
8.7	get_tcpip6_session_info.....	- 8 -
8.8	EzTCP_CloseTcpIp4 .....	- 10 -
8.9	EzTCP_CloseTcpIp6 .....	- 12 -
<b>9</b>	<b>기타</b> .....	<b>- 0 -</b>
9.1	get_version.....	- 0 -
9.2	is_ip6 .....	- 1 -
9.3	is_uart_rs232 .....	- 1 -
9.4	is_uart_rs485 .....	- 2 -
9.5	is_uart_rs422 .....	- 3 -
9.6	is_uart_ttl .....	- 3 -
9.7	is_uart_8_databit_only.....	- 4 -
9.8	is_uart_7_and_8_databit_only .....	- 4 -
9.9	is_uart_one5_stopbit .....	- 5 -
9.10	is_uart_dtrdsr .....	- 6 -
9.11	is_uart_xonxoff.....	- 6 -
9.12	is_uart_tx_delay.....	- 7 -
9.13	is_uart_data_frame_interval .....	- 7 -
9.14	is_uart_separator.....	- 8 -
9.15	is_uart_tcp_nodelay.....	- 9 -
9.16	is_uart_rfc2217.....	- 9 -
9.17	is_uart_protocol .....	- 10 -
9.18	is_uart_tcp_server.....	- 11 -
9.19	is_uart_at_command.....	- 11 -
9.20	is_uart_tcp_client.....	- 12 -
9.21	is_uart_udp .....	- 13 -
9.22	is_uart_serial_modbus .....	- 13 -
9.23	is_csc_hr2 .....	- 14 -
9.24	is_io .....	- 15 -
9.25	is_io_output_automatic_initialize.....	- 15 -
9.26	is_event_notification.....	- 16 -
9.27	is_wlan.....	- 16 -
9.28	is_wlan_soft_ap .....	- 17 -
9.29	is_wlan_antenna .....	- 18 -
9.30	is_wlan_phy_mode .....	- 18 -
9.31	is_wlan_background_scan .....	- 19 -
9.32	is_wlan_rsn .....	- 19 -

9.33 is_wlan_authentication_type .....	- 20 -
9.34 is_wlan_wpa_enterprise.....	- 21 -
9.35 is_send_mac_address.....	- 21 -
9.36 is_ssl .....	- 22 -
9.37 is_ssh .....	- 23 -
9.38 is_remote_debug.....	- 23 -
9.39 is_tcp_multi_connection.....	- 24 -
9.40 is_power_management .....	- 24 -
9.41 is_password .....	- 25 -
<b>10    면책 고지 사항.....</b>	<b>- 0 -</b>
10.1 면책 고지 사항 .....	- 0 -
<b>11    문서 변경 이력.....</b>	<b>- 0 -</b>



# 1 소개

## 1.1 소개

- ezManager 프로그램에서 제공하는 주요 기능을 라이브러리 형태로 제공하므로 사용자가 직접 ezManager와 같은 프로그램을 작성할 수 있습니다.
- ezManager Library는 정적 라이브러리(static library)와 동적 라이브러리(dynamic-link library) 형태로 제공되고 있습니다.

## 1.2 표기법

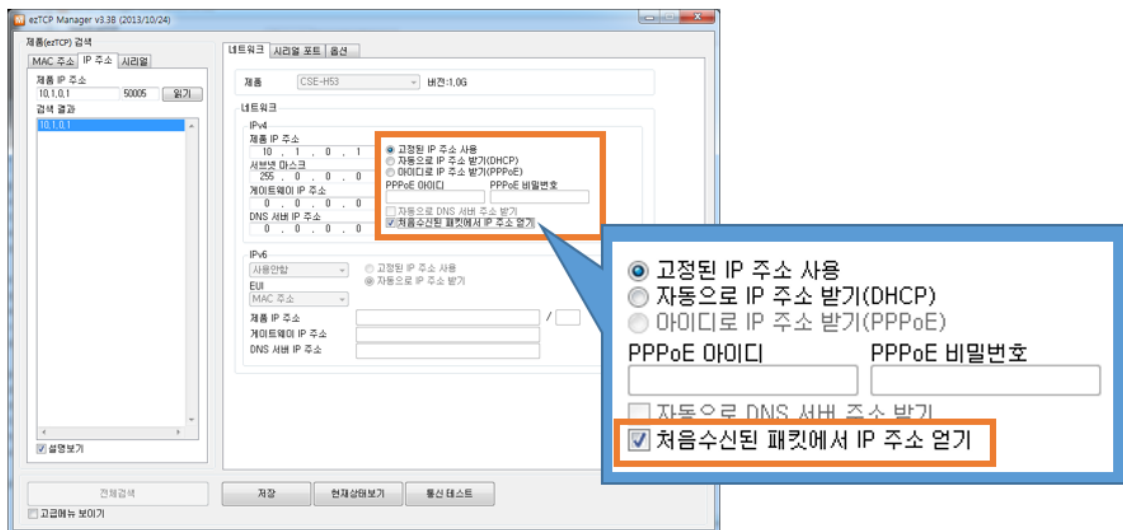
- 본 문서 전체에서 사용한 '['과 ']'으로 감싸진 굵은 글씨를 사용한 문구는 ezManager 프로그램에서 사용되는 용어들입니다.

### 4.1.1 get\_arp

```
int get_arp(unsigned char *mac_address);
```

#### Description

get\_arp는 **[처음 수신된 패킷에서 IP 주소 얻기]** 기능의 선택 여부를 반환합니다.



## 1.3 주의사항

- 비정상적인 라이브러리 사용으로 인한 피해는 당사에서 책임지지 않습니다.

## 2 라이브러리 시작과 종료

### 2.1 Start\_Library

```
void Start_Library(void);
```

#### Description

ezManager 라이브러리 사용을 위해 필요한 메모리를 동적 할당하고 라이브러리 함수들이 사용하는 변수들을 초기화 합니다.

#### Parameters

#### Return values

#### Examples

#### See also

End\_Library

#### Remarks

ezManager 라이브러리 함수들을 오류 없이 사용하기 위해서 응용프로그램이 시작될 때 *Start\_Library* 함수를 호출 하십시오.

### 2.2 End\_Library

```
void End_Library(void);
```

#### Description

*Start\_Library* 함수 호출을 통해서 동적으로 할당한 메모리의 사용을 해제합니다.

#### Parameters

#### Return values

#### Examples

#### See also

Start\_Library

**Remarks**

ezManager 라이브러리 사용을 종료 하는 경우 *End\_Library* 함수를 호출하여 라이브러리 내부에서 동적으로 할당하여 사용한 메모리를 해제 하십시오. 그렇지 않은 경우 메모리 누수가 생길 수 있습니다.

## 3 제품(ezTCP) 검색

### 3.1 EzTCP\_Search

```
int EzTCP_Search(int udp_port_number, int *error);
```

#### Description

로컬 네트워크에서 제품(ezTCP)을 검색합니다. UDP 50005번, 50007번 포트를 사용하여 암호화된 패킷을 브로드캐스트 한 후 2초간 제품(ezTCP)의 응답을 기다립니다.

#### Parameters

*int udp\_port\_number*

제품(ezTCP)을 검색할 때 사용하는 포트 번호를 지정할 수 있습니다. 0을 입력하면 기본 포트번호인 50005번, 50007번을 사용 합니다.

*in \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

#### Return values

*EzTCP\_Search*는 함수 호출이 성공하면 *EZTCP\_SUCCESS*를 반환하고, 그렇지 않으면 *EZTCP\_ERR* 을 반환하고 *error*에 시스템 오류 번호를 저장합니다.

#### Examples

```
int error = 0;
int res = 0;

start_library();

res = EzTCP_Search(0, &error);

if ( res == EZTCP_ERR )
{
    #if defined(__LINUX)
        printf("Error[%d] %s\r\n", error, strerror(error));
    #elif defined(__WINDOWS)
        LPVOID lpMsgBuf;

        FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS, NULL, error, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPSTR)&lpMsgBuf, 0, NULL);

        printf("Error[%d] %s\r\n", error, (char*)lpMsgBuf);
    #endif
}
```

```

else
{
    int i = 0;
    int eztcp_count = get_eztcp_count();

    for(i = 0; i < eztcp_count; i++)
    {
        struct eztcp_info out_eztcp_info;
        int ret = get_eztcp_info_by_index(i, &out_eztcp_info);

        if(ret == EZTCP_SUCCESS)
        {
            printf("MAC Addr. [%02X: %02X: %02X: %02X: %02X: %02X]\r\n",
                out_eztcp_info.mac_address[0], out_eztcp_info.mac_address[1],
                out_eztcp_info.mac_address[2], out_eztcp_info.mac_address[3],
                out_eztcp_info.mac_address[4], out_eztcp_info.mac_address[5]);
        }
    }
}

End_Library();

```

**See also**

EzTCP\_Read, get\_eztcp\_count, get\_eztcp\_info\_by\_index

**Remarks**

*EzTCP\_Search*는 제품(ezTCP)을 로컬 네트워크에서 검색하기 위해서 UDP 브로드캐스트 패킷을 사용하고 있습니다. 이것은 사용자 컴퓨터 시스템에서 실행 중인 방화벽 등 보안 프로그램들 때문에 동작이 차단될 수 있습니다.

## 3.2 EzTCP\_Read

```

int EzTCP_Read(int search_method, unsigned char *mac_address, char *ip_addr,
               int udp_port_number, int *error);

```

**Description**

MAC주소 또는 IP주소를 사용하여 네트워크에서 제품(ezTCP)을 검색합니다.

**Parameters**

*int search\_method*

제품(ezTCP)을 검색할 때 MAC주소를 사용하는 경우 0을, IP주소를 사용하는 경우 1을 입력합니다

*unsigned char \*mac\_address*

제품의 MAC주소로 6바이트입니다.

*char \*ip\_addr*

제품의 IP주소 입니다.

*int udp\_port\_number*

제품(ezTCP)을 검색할 때 사용하는 포트 번호를 지정할 수 있습니다. 0을 입력하면 기본 포트번호인 50005번, 50007번을 사용 합니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

## Return values

*EzTCP\_Read*는 함수 호출이 성공하면 *EZTCP\_SUCCESS*를 반환하고, 그렇지 않으면 *EZTCP\_ERR* 을 반환하고 error에 시스템 오류 번호를 저장합니다.

## Examples

```
int res = 0;
int error = 0;
unsigned char mac_address[6];
char *ip_addr = "10.1.0.1";

mac_address[0] = 0x00;
mac_address[1] = 0x30;
mac_address[2] = 0xf9;
mac_address[3] = 0x00;
mac_address[4] = 0x00;
mac_address[5] = 0x01;

start_library();

// MAC address
res = EzTCP_Read(0, mac_address, NULL, 0, &error);
if ( res == EZTCP_SUCCESS )
{
    int i = 0;
    int eztcp_count = get_eztcp_count();

    for(i = 0; i < eztcp_count; i++)
    {
```

```

        struct eztcp_info out_eztcp_info;
        int ret = get_eztcp_info_by_index(i, &out_eztcp_info);

        if(ret == EZTCP_SUCCESS)
        {
            printf("MAC Addr. [%02X: %02X: %02X: %02X: %02X: %02X]\r\n",
                out_eztcp_info.mac_address[0], out_eztcp_info.mac_address[1],
                out_eztcp_info.mac_address[2], out_eztcp_info.mac_address[3],
                out_eztcp_info.mac_address[4], out_eztcp_info.mac_address[5]);
        }
    }
}

// IP address
res = EzTCP_Read(1, NULL, ip_addr, 0, &error);
if ( res == EZTCP_SUCCESS )
{
    int i = 0;
    int eztcp_count = get_eztcp_count();

    for(i = 0; i < eztcp_count; i++)
    {
        struct eztcp_info out_eztcp_info;
        int ret = get_eztcp_info_by_index(i, &out_eztcp_info);

        if(ret == EZTCP_SUCCESS)
        {
            printf("IP Addr. [%s]\r\n", out_eztcp_info.hostname);
        }
    }
}

End_Library();

```

### See also

EzTCP\_Search, get\_eztcp\_count, get\_eztcp\_info\_by\_index

### Remarks

*EzTCP\_Read*는 *search\_method*에 입력한 값에 따라서 제품(ezTCP)을 네트워크에서 검색하기 위해 UDP 브로드캐스트 패킷을 사용할 수 있습니다. 이것은 사용자 시스템에서 실행 중인 방화벽 등 보안 프로그램들 때문에 동작이 차단될 수 있습니다.

### 3.3 get\_eztcp\_count

```
int get_eztcp_count(void);
```

#### Description

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 개수를 확인할 수 있습니다.

#### Parameters

#### Return values

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 총 개수를 반환합니다.

#### Examples

```
int error = 0;
int res = 0;

Start_Library();

res = EzTCP_Search(0, &error);

if ( res == EZTCP_ERR )
{
#ifdef __LINUX
    printf("Error[%d] %s\r\n", error, strerror(error));
#elif defined(__WINDOWS)
    LPVOID lpMsgBuf;

    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS, NULL, error, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPSTR)&lpMsgBuf, 0, NULL);

    printf("Error[%d] %s\r\n", error, (char*)lpMsgBuf);
#endif
}
else
{
    int i = 0;
    int eztcp_count = get_eztcp_count();

    for(i = 0; i < eztcp_count; i++)
    {
        struct eztcp_info out_eztcp_info;
```



```

        int ret = get_eztcp_info_by_index(i, &out_eztcp_info);

        if(ret == EZTCP_SUCCESS)
        {
            printf("MAC Addr.[%02X: %02X: %02X: %02X: %02X: %02X]\r\n",
                out_eztcp_info.mac_address[0],out_eztcp_info.mac_address[1],
                out_eztcp_info.mac_address[2],out_eztcp_info.mac_address[3],
                out_eztcp_info.mac_address[4],out_eztcp_info.mac_address[5]);
        }
    }
}

End_Library();

```

**See also**

EzTCP\_Search, EzTCP\_Read, get\_eztcp\_info\_by\_index

**Remarks**

### 3.4 get\_eztcp\_info\_by\_index

```
int get_eztcp_info_by_index(int index, struct eztcp_info *out_eztcp_info);
```

**Description**

검색된 제품(ezTCP)의 정보를 조회 합니다.

**Parameters**

*int index*

검색된 제품(ezTCP) 중 정보를 조회할 제품의 순번으로 0부터 시작 합니다.

*struct eztcp\_info \*out\_eztcp\_info*

제품(ezTCP) 정보를 저장할 eztcp\_info 구조체의 포인터입니다.

**Return values**

*index* 값이 유효한 경우 *EZTCP\_SUCCESS*을 반환하고 제품(ezTCP) 정보를 *out\_eztcp\_info*에 저장합니다. 그렇지 않은 경우 *EZTCP\_ERR*을 반환합니다.

**Examples**

```
int error = 0;
```



```

int res = 0;

Start_Library();

res = EzTCP_Search(0, &error);

if ( res == EZTCP_ERR )
{
#ifdef __LINUX
    printf("Error[%d] %s\r\n", error, strerror(error));
#elif defined(__WINDOWS)
    LPVOID lpMsgBuf;
    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS, NULL, error, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPSTR)&lpMsgBuf, 0, NULL);
    printf("Error[%d] %s\r\n", error, (char*)lpMsgBuf);
#endif
}
else
{
    int i = 0;
    int eztcp_count = get_eztcp_count();

    for(i = 0; i < eztcp_count; i++)
    {
        struct eztcp_info out_eztcp_info;
        int ret = get_eztcp_info_by_index(i, &out_eztcp_info);

        if(ret == EZTCP_SUCCESS)
        {
            printf("MAC Addr. [%02X: %02X: %02X: %02X: %02X: %02X]\r\n",
                out_eztcp_info.mac_address[0], out_eztcp_info.mac_address[1],
                out_eztcp_info.mac_address[2], out_eztcp_info.mac_address[3],
                out_eztcp_info.mac_address[4], out_eztcp_info.mac_address[5]);
        }
    }
}

End_Library();

```

**See also**

EzTCP\_Search, EzTCP\_Read, get\_eztcp\_count



## Remarks

eztcp\_info 구조체는 다음과 같은 구조를 가지고 있습니다.

```
struct eztcp_info
{
    unsigned char    mac_address[6];
    char            hostname[NI_MAXHOST];
    unsigned char    comment[65];
    int             env_status;
    char            product_name[32];
    int             search_method;
    int             search_port;
}
```

### *mac\_address*

제품(ezTCP) MAC주소 입니다.

### *hostname*

제품(ezTCP) IP주소 입니다.

### *comment*

제품(ezTCP) 설명 입니다. *NI\_MAXHOST*는 *ezManagerLibrary.h*에 1025로 정의되어 있습니다.

### *env\_status*

제품(ezTCP) 설정 값의 상태를 나타냅니다. 설정 값이 정상인 경우 1, CRC 검사 에러등의 문제가 있는 경우 0 입니다.

### *product\_name*

제품(ezTCP) 이름 입니다.

### *search\_method*

제품(ezTCP)을 검색한 방법입니다. *search\_method*가 0인 경우 MAC 주소 검색, 1인 경우 IP 주소 검색입니다.

### *search\_port*

제품(ezTCP)을 검색한 UDP 포트번호 입니다.

## 4 설정 값 읽기

### 4.1 네트워크

#### 4.1.1 get\_arp

```
int get_arp(unsigned char *mac_address);
```

##### Description

*get\_arp*는 [처음 수신된 패킷에서 IP 주소 얻기] 기능의 선택 여부를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_arp*는 [처음 수신된 패킷에서 IP 주소 얻기] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

##### Remarks

#### 4.1.2 get\_dhcp

```
int get_dhcp(unsigned char *mac_address);
```

##### Description

*get\_dhcp*는 [자동으로 IP 주소 받기(DHCP)] 기능의 선택 여부를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_dhcp*는 [자동으로 IP 주소 받기(DHCP)] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

*get\_dhcp\_dns*

### Remarks

#### 4.1.3 get\_dhcp\_dns

```
int get_dhcp_dns(unsigned char *mac_address);
```

### Description

*get\_dhcp\_dns*는 [자동으로 DNS 서버 주소 받기] 기능의 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_dhcp\_dns*는 [자동으로 DNS 서버 주소 받기] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

*get\_dhcp*

### Remarks

[자동으로 DNS 서버 주소 받기]가 설정된 경우 제품(ezTCP)은 DHCP 서버로부터 할당 받은 DNS 서버 IP 주소를 사용합니다. 그렇지 않은 경우 사용자가 입력한 DNS 서버 IP 주소를 사용합니다.

#### 4.1.4 get\_pppoe

```
int get_pppoe(unsigned char *mac_address);
```

##### Description

*get\_pppoe*는 [아이디로 IP 주소 받기(PPPoE)] 기능의 선택 여부를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_pppoe*는 [아이디로 IP 주소 받기(PPPoE)] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

*get\_pppoe\_id*, *get\_pppoe\_pwd*

##### Remarks

#### 4.1.5 get\_ip4\_local\_address

```
int get_ip4_local_address(unsigned char *mac_address, char *out_address);
```

##### Description

*get\_ip4\_local\_address*는 제품(ezTCP)의 IPv4 [제품 IP 주소]를 *out\_addr*에 저장합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[제품 IP 주소]가 저장될 포인터입니다.

##### Return values

*get\_ip4\_local\_address*는 *out\_addr*에 [제품 IP 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    memset(buf, 0x00, 128);
    get_ip4_local_address(out_eztcp_info.mac_address, buf);
    printf("Local IPv4 : %s\r\n", buf);
}
```

## See also

`get_ip4_local_address`, `get_ip4_subnet_mask`, `get_ip4_gateway_address`, `get_ip4_dns_address`

## Remarks

*get\_ip4\_local\_address*는 IPv4의 점으로 구분된 십진수 표기법으로 [제품 IP 주소]를 *out\_address*에 저장합니다.

### 4.1.6 get\_ip4\_subnet\_mask

```
int get_ip4_subnet_mask(unsigned char *mac_address, char *out_subnet_mask);
```

## Description

*get\_ip4\_subnet\_mask*는 제품(ezTCP)의 [서브넷 마스크]를 *out\_addr*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소입니다.

*char \*out\_subnet\_mask*

[서브넷 마스크]가 저장될 포인터입니다.

## Return values

*get\_ip4\_subnet\_mask*는 *out\_subnet\_mask*에 [서브넷 마스크]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    memset(buf, 0x00, 128);
    get_ip4_subnet_mask(out_eztcp_info.mac_address, buf);
    printf("Subnet mask : %s\r\n", buf);
}
```

## See also

get\_ip4\_local\_address, get\_ip4\_subnet\_mask, get\_ip4\_gateway\_address, get\_ip4\_dns\_address

## Remarks

*get\_ip4\_subnet\_mask*는 IPv4의 점으로 구분된 십진수 표기법으로 [서브넷 마스크]를 *out\_addr*에 저장합니다.

### 4.1.7 get\_ip4\_gateway\_address

```
int get_ip4_gateway_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_ip4\_gateway\_address*는 제품(ezTCP)의 IPv4 [게이트웨이 IP 주소]를 *out\_addr*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[게이트웨이 IP 주소]가 저장될 포인터입니다.

## Return values

*get\_ip4\_gateway\_address*는 *out\_address*에 [게이트웨이 IP 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples





```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    memset(buf, 0x00, 128);
    get_ip4_gateway_address(out_eztcp_info.mac_address, buf);
    printf("Gateway : %s\r\n", buf);
}

```

**See also**

get\_ip4\_local\_address, get\_ip4\_subnet\_mask, get\_ip4\_gateway\_address, get\_ip4\_dns\_address

**Remarks**

*get\_ip4\_gateway\_address*는 IPv4의 점으로 구분 된 십진수 표기법으로 [게이트웨이 IP 주소]를 *out\_address*에 저장합니다.

**4.1.8 get\_ip4\_dns\_address**

```
int get_ip4_dns_address(unsigned char *mac_address, char *out_address);
```

**Description**

*get\_ip4\_dns\_address*는 제품(ezTCP)의 IPv4 [DNS 서버 IP 주소]를 *out\_addr*에 저장합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[DNS 서버 IP 주소]가 저장될 포인터입니다.

**Return values**

*get\_ip4\_dns\_address*는 *out\_address*에 [DNS 서버 IP 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples**

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

```

```

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    memset(buf, 0x00, 128);
    get_ip4_dns_address(out_eztcp_info.mac_address, buf);
    printf("DNS : %s\r\n", buf);
}

```

### See also

get\_ip4\_local\_address, get\_ip4\_subnet\_mask, get\_ip4\_gateway\_address, get\_ip4\_dns\_address.

### Remarks

*get\_ip4\_dns\_address*는 IPv4의 점으로 구분 된 십진수 표기법으로 [DNS 서버 IP 주소]를 *out\_address*에 저장합니다.

#### 4.1.9 get\_pppoe\_id

```
int get_pppoe_id(unsigned char *mac_address, char *out_pppoe_id);
```

### Description

*get\_pppoe\_id*는 제품(ezTCP)의 [PPPoE 아이디]를 *pppoe\_id*에 저장합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_pppoe\_id*

[PPPoE 아이디]가 저장될 포인터입니다.

### Return values

*get\_pppoe\_id*는 *out\_pppoe\_id*에 [PPPoE 아이디]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{

```

```

char buf[128];
memset(buf, 0x00, 128);

get_pppoe_id(out_eztcp_info.mac_address, buf);

printf("PPPoE ID : %s\r\n", buf);
}

```

## See also

get\_pppoe, get\_pppoe\_pwd

## Remarks

[PPPoE 아이디]는 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_pppoe\_id*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.1.10 get\_pppoe\_pwd

```
int get_pppoe_pwd(unsigned char *mac_address, char *out_pppoe_pwd);
```

## Description

*get\_pppoe\_pwd*는 제품(ezTCP)의 [PPPoE 비밀번호]를 *out\_pppoe\_pwd*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_pppoe\_pwd*

[PPPoE 비밀번호]가 저장될 포인터입니다.

## Return values

*get\_pppoe\_pwd*는 *out\_pppoe\_pwd*에 [PPPoE 비밀번호]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    memset(buf, 0x00, 128);

```

```

    get_ppoe_pwd(out_eztcp_info.mac_address, buf);
    printf("PPPoE password : %s\r\n", buf);
}

```

**See also**

get\_pppoe, get\_pppoe\_id

**Remarks**

[PPPoE 비밀번호]는 NULL을 제외하고 최대 16바이트입니다. 따라서 *out\_pppoe\_pwd*는 최소 17바이트의 메모리 공간을 가지고 있어야 합니다.

**4.1.11 get\_ip6**

```
int get_ip6(unsigned char *mac_address);
```

**Description**

*get\_ip6*는 **IPv6** 기능의 선택 여부를 반환합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*get\_ip6*는 **IPv6** 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다. *mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

is\_ip6, get\_ip6\_gua\_type, get\_ip6\_eui\_type, get\_ip6\_local\_address, get\_ip6\_subnet\_prefix\_length, get\_ip6\_gateway\_address, get\_ip6\_dns\_address.

**Remarks**

**IPv6**에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

**4.1.12 get\_ip6\_gua\_type**

```
int get_ip6_gua_type(unsigned char *mac_address);
```

## Description

*get\_ip6\_gua\_type*은 IPv6 [제품 IP 주소] 설정 방식을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_ip6\_gua\_type*은 IPv6 [제품 IP 주소] 설정 방식을 반환하며 그 값은 아래와 같습니다.

0

[자동으로 IP 주소 받기]

1

[고정된 IP 주소 사용]

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_ip6(out_eztcp_info.mac_address) == 1)
    {
        int gua_type = get_ip6_gua_type(out_eztcp_info.mac_address);
        if(gua_type == 0)
            printf("[IPv6] Network Type : Obtain an IP automatically.\r\n");
        else if(gua_type == 1)
            printf("[IPv6] Network Type : Use static IP address.\r\n");
    }
}
```

## See also

*is\_ip6*, *get\_ip6*, *get\_ip6\_eui\_type*, *get\_ip6\_local\_address*, *get\_ip6\_subnet\_prefix\_length*,  
*get\_ip6\_gateway\_address*, *get\_ip6\_dns\_address*

## Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

### 4.1.13 get\_ip6\_eui\_type

```
int get_ip6_eui_type(unsigned char *mac_address);
```

#### Description

*get\_ip6\_eui\_type*은 IPv6 주소의 Interface ID 생성 방식을 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_ip6\_eui\_type*은 Interface ID 생성 방식을 반환하며 그 값은 아래와 같습니다.

0

[MAC 주소]

1

[Random]

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_ip6(out_eztcp_info.mac_address) == 1)
    {
        int eui_type = get_ip6_eui_type(out_eztcp.mac_address);
        if(gua_type == 0)
            printf("[IPv6] EUI : MAC address.\r\n");
        else if(gua_type == 1)
            printf("[IPv6] EUI : Random.\r\n");
    }
}
```

#### See also

is\_ip6, get\_ip6, get\_ip6\_gua\_type, get\_ip6\_local\_address, get\_ip6\_subnet\_prefix\_length,  
get\_ip6\_gateway\_address, get\_ip6\_dns\_address

#### Remarks



IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

#### 4.1.14 get\_ip6\_local\_address

```
int get_ip6_local_address(unsigned char *mac_address, char *out_address);
```

#### Description

*get\_ip6\_local\_address*는 제품(ezTCP)의 IPv6 [제품 IP 주소]를 *out\_addr*에 저장합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

IPv6 [제품 IP 주소]가 저장될 포인터입니다.

#### Return values

*get\_ip6\_local\_address*는 *out\_address*에 IPv6 [제품 IP 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp.mac_address;
    if(is_ip6(mac_address) == 1)
    {
        char buf[128];
        memset(buf, 0x00, 128);
        get_ip6_local_address(mac_address, buf);
        printf("Local IPv6 : %s\r\n", buf);
    }
}
```

#### See also

is\_ip6, get\_ip6, get\_ip6\_gua\_type, get\_ip6\_eui\_type, get\_ip6\_local\_address\_prefix,  
get\_ip6\_gateway\_address, get\_ip6\_dns\_address



**Remarks**

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

**4.1.15 get\_ip6\_subnet\_prefix\_length**

```
int get_ip6_subnet_prefix_length(unsigned char *mac_address,
                                char *out_subnet_prefix_length);
```

**Description**

*get\_ip6\_subnet\_prefix\_length*는 제품(ezTCP)의 IPv6 서브넷 접두사 길이를 *out\_subnet\_prefix\_length*에 저장합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_subnet\_prefix\_length*

IPv6 서브넷 접두사 길이가 저장될 포인터입니다.

**Return values**

*get\_ip6\_subnet\_prefix\_length*는 *out\_subnet\_prefix\_length*에 IPv6 서브넷 접두사 길이를 저장하는 데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples**

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp.mac_address;
    if(is_ip6(mac_address) == 1)
    {
        char buf[128];
        memset(buf, 0x00, 128);
        get_ip6_subnet_prefix_length(mac_address, buf);
        printf("Subnet prefix length : %s\r\n", buf);
    }
}
```



## See also

is\_ip6, get\_ip6, get\_ip6\_gua\_type, get\_ip6\_eui\_type, get\_ip6\_local\_address, get\_ip6\_gateway\_address, get\_ip6\_dns\_address

## Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

### 4.1.16 get\_ip6\_gateway\_address

```
int get_ip6_gateway_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_ip6\_gateway\_address*는 제품(ezTCP)의 IPv6 [게이트웨이 IP 주소]를 *out\_addr*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

IPv6 [게이트웨이 IP 주소]가 저장될 포인터입니다.

## Return values

*get\_ip6\_gateway\_address*는 *out\_address*에 IPv6[제품 IP 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp.mac_address;
    if(is_ip6(mac_address) == 1)
    {
        char buf[128];
        memset(buf, 0x00, 128);
        get_ip6_gateway_address(mac_address, buf);
        printf("Gateway IPv6 : %s\r\n", buf);
    }
}
```

```
}
}
```

## See also

is\_ip6, get\_ip6, get\_ip6\_gua\_type, get\_ip6\_eui\_type, get\_ip6\_local\_address, get\_ip6\_subnet\_prefix\_length, get\_ip6\_dns\_address

## Remarks

**IPv6**에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

### 4.1.17 get\_ip6\_dns\_address

```
int get_ip6_dns_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_ip6\_dns\_address*는 제품(ezTCP)의 IPv6 [DNS 서버 IP 주소]를 *out\_addr*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

IPv6 [DNS 서버 IP 주소]가 저장될 포인터입니다.

## Return values

*get\_ip6\_dns\_address*는 *out\_address*에 IPv6 [DNS 서버 IP 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_ip6(out_eztcp_info.mac_address) == 1)
    {
        char buf[128];
        memset(buf, 0x00, 128);
        get_ip6_dns_address(out_eztcp_info.mac_address, buf);
```

```

        printf("DNS IPV6 : %s\r\n", buf);
    }
}

```

### See also

is\_ip6, get\_ip6, get\_ip6\_gua\_type, get\_ip6\_eui\_type, get\_ip6\_local\_address, get\_ip6\_subnet\_prefix\_length, get\_ip6\_gateway\_address.

### Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 IPv6 가이드 문서를 참조하시기 바랍니다.

## 4.2 시리얼 포트

### 4.2.1 get\_uart\_count

```
int get_uart_count(unsigned char *mac_address);
```

### Description

*get\_uart\_count*는 제품(ezTCP)이 지원하는 시리얼 포트 개수를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_uart\_count*는 제품(ezTCP)이 지원하는 시리얼 포트 개수를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

get\_uart\_min\_baudrate, get\_uart\_max\_baudrate, get\_uart\_baudrate, get\_uart\_ttl, get\_uart\_telnet\_type, get\_uart\_tx\_delay, get\_uart\_parity, get\_uart\_databit, get\_uart\_stopbit, get\_uart\_flow\_control, get\_uart\_dtrdsr.

### Remarks



### 4.2.2 get\_uart\_min\_baudrate

```
int get_uart_min_baudrate(unsigned char *mac_address);
```

#### Description

*get\_uart\_min\_baudrate*는 제품(ezTCP)이 지원하는 최저 시리얼 통신속도를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_uart\_min\_baudrate*는 제품(ezTCP)이 지원하는 최저 시리얼 통신속도를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

*get\_uart\_count*, *get\_uart\_max\_baudrate*, *get\_uart\_baudrate*, *get\_uart\_serial\_type*, *get\_uart\_ttl*,  
*get\_uart\_tx\_delay*, *get\_uart\_parity*, *get\_uart\_databit*, *get\_uart\_stopbit*, *get\_uart\_flow\_control*,  
*get\_uart\_dtrdsr*.

#### Remarks

### 4.2.3 get\_uart\_max\_baudrate

```
int get_uart_max_baudrate(unsigned char *mac_address, int uart_index);
```

#### Description

*get\_uart\_max\_baudrate*는 제품(ezTCP)이 지원하는 최대 시리얼 통신속도를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

#### Return values

*get\_uart\_max\_baudrate*는 제품(ezTCP)이 지원하는 최고 시리얼 통신속도를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

`get_uart_count`, `get_uart_min_baudrate`, `get_uart_baudrate`, `get_uart_serial_type`, `get_uart_ttl`,  
`get_uart_tx_delay`, `get_uart_parity`, `get_uart_databit`, `get_uart_stopbit`, `get_uart_flow_control`,  
`get_uart_dtrdsr`.

## Remarks

제품(ezTCP)이 지원하는 최대 시리얼 통신속도는 다른 설정 값에 따라 달라질 수 있습니다.

### 4.2.4 get\_uart\_serial\_type

```
int get_uart_serial_type(unsigned char *mac_address, int uart_index);
```

## Description

*get\_uart\_serial\_type*은 *uart\_index*로 지정된 시리얼 포트의 [시리얼 종류]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

## Return values

*get\_uart\_serial\_type*은 *uart\_index*로 지정된 시리얼 포트의 [시리얼 종류]를 반환하며 그 값은 아래와 같습니다.

0

RS-232

1

RS-485

2

RS-422

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also



is\_uart\_rs232, is\_uart\_rs485, is\_uart\_rs422, get\_uart\_count, get\_uart\_min\_baudrate,  
get\_uart\_max\_baudrate, get\_uart\_baudrate, get\_uart\_ttl, get\_uart\_tx\_delay, get\_uart\_parity,  
get\_uart\_databit, get\_uart\_stopbit, get\_uart\_flow\_control, get\_uart\_dtrdsr.

## Remarks

### 4.2.5 get\_uart\_ttl

```
int get_uart_ttl(unsigned char *mac_address, int uart_index);
```

## Description

*get\_uart\_ttl*은 *uart\_index*로 지정된 시리얼 포트의 [TTL] 출력 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

## Return values

*get\_uart\_ttl*은 *uart\_index*로 지정된 시리얼 포트가 [TTL] 출력 기능을 사용 중 이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

get\_uart\_count, get\_uart\_min\_baudrate, get\_uart\_max\_baudrate, get\_uart\_baudrate, get\_uart\_serial\_type,  
is\_uart\_ttl, get\_uart\_tx\_delay, get\_uart\_parity, get\_uart\_databit, get\_uart\_stopbit, get\_uart\_flow\_control,  
get\_uart\_dtrdsr

## Remarks

### 4.2.6 get\_uart\_baudrate

```
int get_uart_baudrate(unsigned char *mac_address, int uart_index);
```

## Description



*get\_uart\_baudrate*는 *uart\_index*로 지정된 시리얼 포트의 [시리얼 통신속도]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_baudrate*는 *uart\_index*로 지정된 시리얼 포트의 [시리얼 통신속도]를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

*get\_uart\_count*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *get\_uart\_serial\_type*, *get\_uart\_ttl*,  
*get\_uart\_tx\_delay*, *get\_uart\_parity*, *get\_uart\_databit*, *get\_uart\_stopbit*, *get\_uart\_flow\_control*,  
*get\_uart\_dtrdsr*,

### Remarks

*uart\_index*로 지정된 시리얼 포트의 [시리얼 통신속도]는 다른 설정 값에 따라 자동으로 변경될 수 있습니다.

#### 4.2.7 get\_uart\_parity

```
int get_uart_parity(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_parity*는 *uart\_index*로 지정된 시리얼 포트의 [패리티]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_parity*은 *uart\_index*로 지정된 시리얼 포트의 [패리티]를 반환하며 그 값은 아래와 같습니다.

0	NONE
1	EVEN
2	ODD
3	MARK
4	SPACE

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

`get_uart_count`, `get_uart_min_baudrate`, `get_uart_max_baudrate`, `get_uart_baudrate`, `get_uart_ttl`,  
`get_uart_tx_delay`, `get_uart_databit`, `get_uart_stopbit`, `get_uart_flow_control`, `get_uart_dtrdsr`,

### Remarks

#### 4.2.8 `get_uart_databit`

```
int get_uart_databit(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_databit*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 비트]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_databit*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 비트]를 반환하며 그 값은 아래와 같습니다.

0
5비트



1

6비트

2

7비트

3

8비트

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

`get_uart_count`, `get_uart_min_baudrate`, `get_uart_max_baudrate`, `get_uart_baudrate`, `is_uart_ttl`, `get_uart_tx_delay`, `get_uart_parity`, `is_uart_8_databit_only`, `is_uart_7_and_8_databit_only`, `get_uart_stopbit`, `get_uart_flow_control`, `get_uart_dtrdsr`

## Remarks

*is\_uart\_8\_databit\_only*이 1을 반환하면 8비트 데이터 비트만 지원합니다.

*is\_uart\_7\_and\_8\_databit\_only*이 1을 반환하면 7, 8비트 데이터 비트만 지원합니다.

### 4.2.9 get\_uart\_stopbit

```
int get_uart_stopbit(unsigned char *mac_address, int uart_index);
```

## Description

*get\_uart\_stopbit*는 *uart\_index*로 지정된 시리얼 포트의 [정지 비트]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

## Return values

*get\_uart\_stopbit*은 *uart\_index*로 지정된 시리얼 포트의 [정지 비트]를 반환하며 그 값은 아래와 같습니다.

0

1비트

1

1.5비트

2

2비트

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

`get_uart_count`, `get_uart_min_baudrate`, `get_uart_max_baudrate`, `get_uart_baudrate`, `get_uart_ttl`,  
`get_uart_tx_delay`, `get_uart_parity`, `get_uart_databit`, `is_uart_one5_stopbit`, `get_uart_flow_control`,  
`get_uart_dtrdsr`,

### Remarks

*is\_uart\_one5\_stopbit*가 1을 반환하는 경우에만 1.5비트 [정지 비트]를 사용할 수 있습니다.

#### 4.2.10 get\_uart\_flow\_control

```
int get_uart_flow_control(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_flow\_control*는 *uart\_index*로 지정된 시리얼 포트의 [흐름 제어]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_flow\_control*은 *uart\_index*로 지정된 시리얼 포트의 [흐름 제어]를 반환하며 그 값은 아래와 같습니다.

0

NONE

1

RTS/CTS

2

XON/XOFF

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples



**See also**

get\_uart\_count, get\_uart\_min\_baudrate, get\_uart\_max\_baudrate, get\_uart\_baudrate, get\_uart\_ttl,  
get\_uart\_tx\_delay, get\_uart\_parity, get\_uart\_databit, get\_uart\_stopbit, get\_uart\_dtrdsr.

**Remarks****4.2.11 get\_uart\_dtrdsr**

```
int get_uart_dtrdsr(unsigned char *mac_address, int uart_index);
```

**Description**

*get\_uart\_dtrdsr*은 *uart\_index*로 지정된 시리얼 포트의 [DTR/DSR] 기능의 선택 여부를 반환합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

**Return values**

*get\_uart\_dtrdsr*은 *uart\_index*로 지정된 시리얼 포트가 [DTR/DSR] 기능을 사용 중 이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

get\_uart\_count, get\_uart\_min\_baudrate, get\_uart\_max\_baudrate, get\_uart\_baudrate, get\_uart\_ttl,  
get\_uart\_tx\_delay, get\_uart\_parity, get\_uart\_databit, get\_uart\_stopbit, get\_uart\_flow\_control, is\_uart\_dtrdsr

**Remarks****4.2.12 get\_uart\_tx\_delay**

```
int get_uart_tx_delay(unsigned char *mac_address, int uart_index);
```

**Description**

*get\_uart\_tx\_delay*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 전송 간격]을 반환합니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

**Return values***get\_uart\_tx\_delay*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 전송 간격]을 반환합니다.*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.**Examples****See also***get\_uart\_count*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *get\_uart\_baudrate*, *get\_uart\_ttl*, *is\_uart\_tx\_delay*, *get\_uart\_parity*, *get\_uart\_databit*, *get\_uart\_stopbit*, *get\_uart\_flow\_control*, *get\_uart\_dtrdsr***Remarks**

## 4.3 TCP/IP

### 4.3.1 get\_uart\_communication\_mode

```
int get_uart_communication_mode(unsigned char *mac_address, int uart_index);
```

**Description***get\_uart\_communication\_mode*는 *uart\_index*로 지정된 시리얼 포트의 [통신모드]를 반환합니다.**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

**Return values***get\_uart\_communication\_mode*는 *uart\_index*로 지정된 시리얼 포트의 [통신모드]를 반환하며 그 값은 아래와 같습니다.

0

- T2S – TCP 서버
- 1
- ATC – AT 명령
- 2
- COD – TCP 클라이언트
- 3
- U2S – UDP
- 4
- 시리얼 Modbus/TCP

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*,  
*get\_uart\_peer\_address*, *get\_uart\_peer\_port*, *get\_uart\_local\_port*, *get\_uart\_cod\_tcp\_server*,  
*get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*,  
*get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *get\_uart\_protocol*

### Remarks

#### 4.3.2 *get\_uart\_peer\_address*

```
int get_uart_peer_address(unsigned char *mac_address, int uart_index,
                          char *out_address);
```

### Description

*get\_uart\_peer\_address*는 *uart\_index*로 지정된 시리얼 포트의 [통신할 주소]를 *out\_addr*에 저장합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_address*

[통신할 주소]가 저장될 포인터입니다.

### Return values



*get\_uart\_peer\_address*는 *out\_address*에 [통신할 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    for(i=0;i<uart_count;i++)
    {
        memset(buf, 0x00, 65);
        get_uart_peer_address(out_eztcp_info.mac_address, i, buf);
        printf("UART[%d] Peer address : %s\r\n", i, buf);
    }
}
```

## See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_port*, *get\_uart\_local\_port*, *get\_uart\_cod\_tcp\_server*, *get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*, *get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *get\_uart\_protocol*

## Remarks

[통신할 주소]는 NULL을 제외하고 최대 64바이트입니다. 따라서 *out\_address*은 최소 65바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.3.3 get\_uart\_peer\_port

```
int get_uart_peer_port(unsigned char *mac_address, int uart_index, char *out_port);
```

## Description

*get\_uart\_peer\_port*는 *uart\_index*로 지정된 시리얼 포트의 [통신할 포트]를 *out\_port*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_port*

[통신할 포트]가 저장될 포인터입니다.

## Return values

*get\_uart\_peer\_port*는 *out\_port*에 [통신할 포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    for(i=0; i<uart_count; i++)
    {
        memset(buf, 0x00, 65);
        get_uart_peer_port(out_eztcp_info.mac_address, i, buf);
        printf("UART[%d] Peer port : %s\r\n", i, buf);
    }
}
```

## See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_address*, *get\_uart\_local\_port*, *get\_uart\_cod\_tcp\_server*,  
*get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*,  
*get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *get\_uart\_protocol*

## Remarks

### 4.3.4 get\_uart\_local\_port

```
int get_uart_local_port(unsigned char *mac_address, int uart_index, char *out_port);
```

## Description

*get\_uart\_local\_port*는 *uart\_index*로 지정된 시리얼 포트의 [제품 로컬포트]를 *out\_port*에 저장합니다.



## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_port*

[제품 로컬포트]가 저장될 포인터입니다.

## Return values

*get\_uart\_local\_port*는 *out\_port*에 [제품 로컬포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    for(i=0; i<uart_count; i++)
    {
        memset(buf, 0x00, 65);
        get_uart_local_port(out_eztcp_info.mac_address, i, buf);
        printf("UART[%d] Local port : %s\r\n", i, buf);
    }
}
```

## See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_address*, *get\_uart\_peer\_port*, *get\_uart\_cod\_tcp\_server*,  
*get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*,  
*get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *get\_uart\_protocol*

## Remarks

### 4.3.5 get\_uart\_cod\_tcp\_server

```
int get_uart_cod_tcp_server(unsigned char *mac_address, int uart_index);
```



## Description

*get\_uart\_cod\_tcp\_server*은 *uart\_index*로 지정된 시리얼 포트의 [TCP 서버] 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

## Return values

*get\_uart\_cod\_tcp\_server*은 *uart\_index*로 지정된 시리얼 포트가 [TCP 서버] 기능을 사용 중 이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_address*, *get\_uart\_peer\_port*, *get\_uart\_local\_port*, *get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*, *get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *get\_uart\_protocol*

## Remarks

### 4.3.6 get\_uart\_watermark

```
int get_uart_watermark(unsigned char *mac_address, int uart_index, char *out_watermark);
```

## Description

*get\_uart\_watermark*는 *uart\_index*로 지정된 시리얼 포트의 [접속전 데이터 크기] 또는 [패킷 블록 설정(바이트)]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_watermark*

[접속전 데이터 크기] 또는 [패킷 블록 설정(바이트)]가 저장될 포인터입니다.

## Return values

*get\_uart\_watermark*는 *out\_watermark*에 [접속전 데이터 크기] 또는 [패킷 블록 설정(바이트)]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    for(i=0; i<uart_count; i++)
    {
        memset(buf, 0x00, 65);
        int comm_mode = get_uart_communication_mode(out_eztcp_info.mac_address, i);
        if(comm_mode==MUX_TYPE_T2S || comm_mode==MUX_TYPE_ATC || comm_mode==MUX_TYPE_COD)
        {
            get_uart_watermark(out_eztcp_info.mac_address, i, buf);
            printf("UART[%d] Event byte : %s\r\n", i, buf);
        }
        else if(comm_mode==MUX_TYPE_U2S)
        {
            get_uart_watermark(out_eztcp_info.mac_address, i, buf);
            printf("UART[%d] Block size (byte) : %s\r\n", i, buf);
        }
        else
        {
            printf("Not available.\r\n");
        }
    }
}
```

## See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_address*, *get\_uart\_peer\_port*, *get\_uart\_local\_port*,  
*get\_uart\_cod\_tcp\_server*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *get\_uart\_separator\_length*,

get\_uart\_separator\_type, get\_uart\_separator, get\_uart\_tcp\_nodelay, get\_uart\_rfc2217, get\_uart\_protocol

## Remarks

[통신모드]가 0 (T2S - TCP 서버), 1 (ATC - AT 명령), 2 (COD - TCP 클라이언트)인 경우 [접속전 데이터 크기]로 사용됩니다.

[통신모드]가 3 (U2S - UDP)인 경우 [패킷 블록 설정 (바이트)]로 사용됩니다.

### 4.3.7 get\_uart\_timeout

```
int get_uart_timeout(unsigned char *mac_address, int uart_index, char *out_timeout);
```

## Description

*get\_uart\_timeout*은 *uart\_index*로 지정된 시리얼 포트의 [접속종료 대기시간(초)]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_timeout*

[접속종료 대기시간(초)]이 저장될 포인터입니다.

## Return values

*get\_uart\_timeout*은 *out\_timeout*에 [접속종료 대기시간(초)]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    for(i=0; i<uart_count; i++)
    {
        memset(buf, 0x00, 65);
```

```

        get_uart_timeout(out_eztcp_info.mac_address,i,buf);
        printf("UART[%d] Timeout : %s\r\n", i, buf);
    }
}

```

## See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port, get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_data\_frame\_interval, get\_uart\_separator\_length, get\_uart\_separator\_type, get\_uart\_separator, get\_uart\_tcp\_nodelay, get\_uart\_rfc2217, get\_uart\_protocol

## Remarks

### 4.3.8 get\_uart\_data\_frame\_interval

```

int get_uart_data_frame_interval(unsigned char *mac_address, int uart_index,
                                char *out_data_frame_interval);

```

## Description

*get\_uart\_data\_frame\_interval*은 *uart\_index*로 지정된 시리얼 포트의 [데이터 프레임 간격(10ms)]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*out\_data\_frame\_interval*

[데이터 프레임 간격(10ms)]이 저장될 포인터입니다.

## Return values

*get\_uart\_data\_frame\_interval*은 *out\_data\_frame\_interval*에 [데이터 프레임 간격(10ms)]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)

```

```

{
    int i = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    unsigned char *mac_address = out_eztcp_info.mac_address;
    for(i=0;i<uart_count;i++)
    {
        memset(buf, 0x00, 65);

        get_uart_data__frame_interval(mac_address, i, buf);
        printf("UART[%d] Data frame interval : %s\r\n", i, buf);
    }
}

```

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port,  
get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, is\_uart\_data\_frame\_interval,  
get\_uart\_separator\_length, get\_uart\_separator\_type, get\_uart\_separator, get\_uart\_tcp\_nodelay,  
get\_uart\_rfc2217, get\_uart\_protocol

### Remarks

#### 4.3.9 get\_uart\_separator\_length

```
int get_uart_separator_length(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_separator\_length*는 *uart\_index*로 지정된 시리얼 포트의 [구분자 길이]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_separator\_length*는 [구분자 길이]를 반환하며 그 값은 0 ~ 4입니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port, get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, get\_uart\_data\_frame\_interval, is\_uart\_separator, get\_uart\_separator\_type, get\_uart\_separator, get\_uart\_tcp\_nodelay, get\_uart\_rfc2217, get\_uart\_protocol

### Remarks

#### 4.3.10 get\_uart\_separator\_type

```
int get_uart_separator_type(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_separator\_type*은 *uart\_index*로 지정된 시리얼 포트의 [구분자 동작방식]을 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_separator\_type*은 [구분자 동작방식]을 반환하며 그 값은 아래와 같습니다.

0

구분자까지 전송

1

구분자+1바이트 전송

2

구분자+2바이트 전송

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port, get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, get\_uart\_data\_frame\_interval, is\_uart\_separator, get\_uart\_separator\_length, get\_uart\_separator, get\_uart\_tcp\_nodelay, get\_uart\_rfc2217,

get\_uart\_protocol

## Remarks

### 4.3.11 get\_uart\_separator

```
int get_uart_separator(unsigned char *mac_address, int uart_index, int separator_index,
                      char *out_separator);
```

## Description

*get\_uart\_separator*는 *uart\_index*로 지정된 시리얼 포트의 [구분자(16진수)]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int separator\_index*

구분자의 순번으로 0부터 시작합니다.

*char \*out\_separator*

*separator\_index*로 지정된 구분자가 16진수 문자열로 저장될 포인터입니다.

## Return values

*get\_uart\_separator*는 *out\_separator*에 *separator\_index*로 지정된 구분자를 16진수 문자열로 저장하는데 성공하면 1을 반환합니다.

*separator\_index*가 지정된 범위를 벗어나는 경우에는 -2를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int i = 0;
    int j = 0;
    char buf[65];
    int uart_count = get_uart_count(out_eztcp_info.mac_address);
    unsigned char *mac_address = out_eztcp_info.mac_address;
```

```

for(i=0;i<uart_count;i++)
{
    memset(buf, 0x00, 65);

    int sep_count = get_uart_separator_length(mac_address, i);
    for(j=0;j<sep_count;j++)
    {
        get_uart_separator(mac_address, i, j, buf);
        printf("UART[%d] Separator[%d] - %s\r\n", i, j, buf);
    }
}
}

```

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port,  
 get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, get\_uart\_data\_frame\_interval,  
 is\_uart\_separator, get\_uart\_separator\_length, get\_uart\_separator\_type, get\_uart\_tcp\_nodelay,  
 get\_uart\_rfc2217, get\_uart\_protocol

### Remarks

#### 4.3.12 get\_uart\_tcp\_nodelay

```
int get_uart_tcp_nodelay(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_tcp\_nodelay*는 *uart\_index*로 지정된 시리얼 포트의 [전송지연 기능 사용안함] 기능의 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_tcp\_nodelay*는 *uart\_index*로 지정된 시리얼 포트가 [전송지연 기능 사용안함] 기능을 사용 중 이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.



## Examples

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port, get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, get\_uart\_data\_frame\_interval, is\_uart\_separator, get\_uart\_separator\_type, get\_uart\_separator, is\_uart\_tcp\_nodelay, get\_uart\_rfc2217, get\_uart\_protocol

### Remarks

#### 4.3.13 get\_uart\_rfc2217

```
int get_uart_rfc2217(unsigned char *mac_address, int uart_index);
```

### Description

*get\_uart\_rfc2217*은 *uart\_index*로 지정된 시리얼 포트의 [시리얼 포트 설정/상태 전송 (RFC2217)] 기능의 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*get\_uart\_rfc2217*은 *uart\_index*로 지정된 시리얼 포트가 [시리얼 포트 설정/상태 전송 (RFC2217)] 기능을 사용 중 이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

get\_uart\_communication\_mode, get\_uart\_peer\_address, get\_uart\_peer\_port, get\_uart\_local\_port, get\_uart\_cod\_tcp\_server, get\_uart\_watermark, get\_uart\_timeout, get\_uart\_data\_frame\_interval, is\_uart\_separator, get\_uart\_separator\_type, get\_uart\_separator, get\_uart\_tcp\_nodelay, is\_uart\_rfc2217, get\_uart\_protocol

### Remarks

### 4.3.14 get\_uart\_protocol

```
int get_uart_protocol(unsigned char *mac_address, int uart_index);
```

#### Description

*get\_uart\_protocol*은 *uart\_index*로 지정된 시리얼 포트가 사용 중인 [프로토콜]을 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

#### Return values

*get\_uart\_protocol*은 *uart\_index*로 지정된 시리얼 포트가 사용 중인 [프로토콜]을 반환하며 그 값은 아래와 같습니다.

0

TCP

1

TCP + TELNET

2

TCP + SSL

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

*get\_uart\_communication\_mode*, *get\_uart\_peer\_address*, *get\_uart\_peer\_port*, *get\_uart\_local\_port*, *get\_uart\_cod\_tcp\_server*, *get\_uart\_watermark*, *get\_uart\_timeout*, *get\_uart\_data\_frame\_interval*, *is\_uart\_separator*, *get\_uart\_separator\_type*, *get\_uart\_separator*, *get\_uart\_tcp\_nodelay*, *get\_uart\_rfc2217*, *is\_uart\_protocol*.

#### Remarks

현재 CSE-T16, CSE-T32, CSE-T48만 사용 가능하며 추후 지원가능 모델은 추가 될 수 있습니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 4.4 CSC-HR2

### 4.4.1 get\_csc\_hr2\_communication\_mode

```
int get_csc_hr2_communication_mode (unsigned char *mac_address);
```

#### Description

*get\_csc\_hr2\_communication\_mode*는 CSC-HR2의 [통신모드]를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_csc\_hr2\_communication\_mode*는 CSC-HR2의 [통신모드]를 반환하며 그 값은 아래와 같습니다.

0

자동

1

EZU-100 펌웨어 변경

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

is\_csc\_hr2, get\_csc\_hr2\_id, get\_csc\_hr2\_network\_timeout, get\_csc\_hr2\_network\_threshold,  
get\_csc\_hr2\_server\_timeout, get\_csc\_hr2\_server\_threshold, get\_csc\_hr2\_check\_port,  
get\_csc\_hr2\_first\_server\_address, get\_csc\_hr2\_first\_server\_port, get\_csc\_hr2\_second\_server\_address,  
get\_csc\_hr2\_second\_server\_port.

#### Remarks

### 4.4.2 get\_csc\_hr2\_id

```
int get_csc_hr2_id(unsigned char *mac_address, char *out_id);
```

#### Description

*get\_csc\_hr2\_id*는 CSC-HR2의 [CSC-HR2 아이디]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_addr*

[CSC-HR2 아이디]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_id*는 *out\_id*에 [CSC-HR2 아이디]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char csc_hr2_id[17];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(csc_hr2_id, 0x00, 17);
        get_csc_hr2_id(out_eztcp_info.mac_address, csc_hr2_id);
        printf("ID of CSC-HR2 : %s\r\n", csc_hr2_id);
    }
}
```

## See also

is_csc_hr2,	get_csc_hr2_communication_mode,	get_csc_hr2_network_timeout,
get_csc_hr2_network_threshold,	get_csc_hr2_server_timeout,	get_csc_hr2_server_threshold,
get_csc_hr2_check_port,	get_csc_hr2_first_server_address,	get_csc_hr2_first_server_port,
get_csc_hr2_second_server_address,	get_csc_hr2_second_server_port,	

## Remarks

[CSC-HR2 아이디]는 NULL을 제외하고 최대 16바이트입니다. 따라서 *out\_id*은 최소 17바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.4.3 get\_csc\_hr2\_network\_timeout

```
int get_csc_hr2_network_timeout(unsigned char *mac_address, char *out_timeout);
```

## Description



*get\_csc\_hr2\_network\_timeout*은 CSC-HR2의 [절체 타임아웃]을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_timeout*

[절체 타임아웃]이 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_network\_timeout*은 *out\_timeout*에 [절체 타임아웃]을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_csc_hr2_network_timeout(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Network Timeout : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_threshold*,  
*get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*, *get\_csc\_hr2\_check\_port*,  
*get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

### 4.4.4 get\_csc\_hr2\_network\_threshold

```
int get_csc_hr2_network_threshold(unsigned char *mac_address, char *out_threshold);
```

## Description

*get\_csc\_hr2\_network\_threshold*는 CSC-HR2의 [절체 바이트 수]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_threshold*

[절체 바이트 수]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_network\_threshold*는 *out\_threshold*에 [절체 바이트 수]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_csc_hr2_network_threshold(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Network Threshold : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*, *get\_csc\_hr2\_check\_port*,  
*get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

### 4.4.5 get\_csc\_hr2\_server\_timeout

```
int get_csc_hr2_server_timeout(unsigned char *mac_address, char *out_timeout);
```

## Description

*get\_csc\_hr2\_server\_timeout*은 CSC-HR2의 [서버변경 타임아웃]을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_timeout*

[서버변경 타임아웃]이 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_server\_timeout*은 *out\_timeout*에 [서버변경 타임아웃]을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_csc_hr2_server_timeout(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Server Timeout : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_threshold*, *get\_csc\_hr2\_check\_port*,  
*get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

### 4.4.6 get\_csc\_hr2\_server\_threshold

```
int get_csc_hr2_server_threshold(unsigned char *mac_address, char *out_threshold);
```

## Description

*get\_csc\_hr2\_server\_threshold*는 CSC-HR2의 [서버변경 바이트 수]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_threshold*

[서버변경 바이트 수]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_server\_threshold*는 *out\_threshold*에 [서버변경 바이트 수]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_csc_hr2_server_threshold(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Server Threshold : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_check\_port*,  
*get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

### 4.4.7 get\_csc\_hr2\_check\_port

```
int get_csc_hr2_check_port(unsigned char *mac_address, char *out_check_port);
```



## Description

*get\_csc\_hr2\_check\_port*는 CSC-HR2의 [통신품질 점검포트]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_threshold*

[통신품질 점검포트]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_check\_port*는 *out\_check\_port*에 [통신품질 점검포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_csc_hr2_check_port(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Check Port : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*,  
*get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

### 4.4.8 get\_csc\_hr2\_first\_server\_address

```
int get_csc_hr2_first_server_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_csc\_hr2\_first\_server\_address*는 CSC-HR2의 [첫 번째 서버 주소]를 *out\_addr*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[첫 번째 서버 주소]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_first\_server\_address*는 *out\_address*에 [첫 번째 서버 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_csc_hr2_first_server_address(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] First Server Address : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*,  
*get\_csc\_hr2\_check\_port*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

## Remarks

[첫 번째 서버 주소]는 NULL을 제외하고 최대 64바이트입니다. 따라서 *out\_address*은 최소 65바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.4.9 get\_csc\_hr2\_first\_server\_port

```
int get_csc_hr2_first_server_port(unsigned char *mac_address, char *out_port);
```

##### Description

*get\_csc\_hr2\_first\_server\_port*는 CSC-HR2가 통신할 첫 번째 서버의 [포트번호]를 *out\_port*에 저장합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_port*

[포트번호]가 저장될 포인터입니다.

##### Return values

*get\_csc\_hr2\_first\_server\_port*는 *out\_port*에 통신할 첫 번째 서버의 [포트번호]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_csc_hr2_first_server_port(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] First Server Port Number : %s\r\n", buf);
    }
}
```

##### See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*,  
*get\_csc\_hr2\_check\_port*, *get\_csc\_hr2\_first\_server\_address*, *get\_csc\_hr2\_second\_server\_address*,  
*get\_csc\_hr2\_second\_server\_port*.

##### Remarks



#### 4.4.10 get\_csc\_hr2\_second\_server\_address

```
int get_csc_hr2_second_server_address(unsigned char *mac_address, char *out_address);
```

##### Description

*get\_csc\_hr2\_second\_server\_address*는 CSC-HR2의 [두 번째 서버 주소]를 *out\_address*에 저장합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[두 번째 서버 주소]가 저장될 포인터입니다.

##### Return values

*get\_csc\_hr2\_second\_server\_address*는 *out\_address*에 [두 번째 서버 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_csc_hr2_second_server_address(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Second Server Address : %s\r\n", buf);
    }
}
```

##### See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,  
*get\_csc\_hr2\_network\_threshold*, *get\_csc\_hr2\_server\_timeout*, *get\_csc\_hr2\_server\_threshold*,  
*get\_csc\_hr2\_check\_port*, *get\_csc\_hr2\_first\_server\_port*, *get\_csc\_hr2\_second\_server\_port*.

## Remarks

[두 번째 서버 주소]는 NULL을 제외하고 최대 64바이트입니다. 따라서 *out\_address*은 최소 65바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.4.11 get\_csc\_hr2\_second\_server\_port

```
int get_csc_hr2_second_server_port(unsigned char *mac_address, char *out_port);
```

## Description

*get\_csc\_hr2\_second\_server\_port*는 CSC-HR2가 통신할 두 번째 서버의 [포트번호]를 *out\_port*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_port*

[포트번호]가 저장될 포인터입니다.

## Return values

*get\_csc\_hr2\_second\_server\_port*는 *out\_port*에 통신할 두 번째 서버의 [포트번호]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    if(is_csc_hr2(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_csc_hr2_second_server_port(out_eztcp_info.mac_address, buf);
        printf("[CSC-HR2] Second Server Port Number : %s\r\n", buf);
    }
}
```

## See also

*is\_csc\_hr2*, *get\_csc\_hr2\_communication\_mode*, *get\_csc\_hr2\_id*, *get\_csc\_hr2\_network\_timeout*,



get\_csc\_hr2\_network\_threshold, get\_csc\_hr2\_server\_timeout, get\_csc\_hr2\_server\_threshold,  
get\_csc\_hr2\_check\_port, get\_csc\_hr2\_first\_server\_address, get\_csc\_hr2\_second\_server\_address.

## Remarks

## 4.5 I/O 제품

### 4.5.1 get\_io\_http

```
int get_io_http(unsigned char *mac_address);
```

## Description

*get\_io\_http*는 I/O 제품 [웹(HTTP)] 제어방식 기능 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_http*는 [웹(HTTP)] 제어방식 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

is\_io, get\_io\_http, get\_io\_html\_size

## Remarks

### 4.5.2 get\_io\_http\_port

```
int get_io_http_port(unsigned char *mac_address, char *out_http_port);
```

## Description

*get\_io\_http\_port*는 I/O 제품 [웹(HTTP) 포트] 번호를 반환합니다.

## Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_http\_port*

[웹(HTTP) 포트] 번호가 저장될 포인터입니다.

## Return values

*get\_io\_http\_port*는 *out\_http\_port*에 [웹(HTTP) 포트] 번호를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 8);
        get_io_http_port(out_eztcp_info.mac_address, buf);
        printf("[I/O] Web(HTTP) Port : %s\r\n", buf);
    }
}
```

## See also

*is\_io*, *get\_io\_http*, *get\_io\_html\_size*

## Remarks

### 4.5.3 get\_io\_html\_size

```
int get_io_html_size(unsigned char *mac_address);
```

## Description

*get\_io\_html\_size*는 I/O 제품 [웹(HTTP)페이지 크기]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_io\_html\_size*는 [웹(HTTP)페이지 크기]를 반환 하며 그 값은 아래와 같습니다.

0

80KB

1

96KB

2

112KB

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        int html_size = get_io_html_size(out_eztcp_info.mac_address);
        printf("[I/O] Size of web(HTTP) : ");
        switch(html_size)
        {
            case 0:
                printf("80KB\r\n");
                break;
            case 1:
                printf("96KB\r\n");
                break;
            case 2:
                printf("112KB\r\n");
                break;
        }
    }
}
```

### See also

is\_io, get\_io\_http, get\_io\_http\_port.



## Remarks

### 4.5.4 get\_io\_modbus

```
int get_io_modbus(unsigned char *mac_address);
```

## Description

*get\_io\_modbus*는 I/O 제품 [Modbus/TCP] 제어방식 기능 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_modbus*는 [Modbus/TCP] 제어방식 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

## Remarks

### 4.5.5 get\_io\_input\_notification

```
int get_io_input_notification(unsigned char *mac_address);
```

## Description

*get\_io\_input\_notification*은 I/O 제품의 [입력포트 변경 알림] 기능 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_input\_notification*은 [입력포트 변경 알림] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

## Remarks

### 4.5.6 *get\_io\_output\_automatic\_initialize*

```
int get_io_output_automatic_initialize(unsigned char *mac_address);
```

## Description

*get\_io\_output\_automatic\_initialize*는 I/O 제품의 [출력포트 상태 초기화(Modbus/TCP 접속 종료 시 출력포트[초기상태] 값으로 상태 변경) 기능 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_output\_automatic\_initialize*은 [출력포트 상태 초기화(Modbus/TCP 접속 종료 시 출력포트[초기상태] 값으로 상태 변경) 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *is\_io\_output\_automatic\_initialize*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

**Remarks**

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**4.5.7 get\_io\_modbus\_type**

```
int get_io_modbus_type(unsigned char *mac_address);
```

**Description**

*get\_io\_modbus\_type*은 I/O 제품의 [마스터/슬레이브]를 반환합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*get\_io\_modbus\_type*은 [마스터/슬레이브]를 반환 하며 그 값은 아래와 같습니다.

0

슬레이브

1

마스터

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples**

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        int Modbus_type = get_io_modbus_type(out_eztcp_info.mac_address);
        switch(Modbus_type)
        {
            case 0:
                printf("[I/O] This is Modbus slave.\r\n");
                break;
            case 1:
                printf("[I/O] This is Modbus master.\r\n");
                break;
```

```

    }
}
}

```

## See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

## Remarks

### 4.5.8 get\_io\_unit\_id

```
int get_io_unit_id(unsigned char *mac_address, char *out_unit_id);
```

## Description

*get\_io\_unit\_id*는 I/O 제품의 [유니트 아이디]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_unit\_id*

[유니트 아이디]가 저장될 포인터입니다.

## Return values

*get\_io\_unit\_id*는 *out\_unit\_id*에 [유니트 아이디]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 8);
        get_io_unit_id(out_eztcp_info.mac_address, buf);
    }
}

```

```

        printf("[I/O] Unit ID : %s\r\n", buf);
    }
}

```

## See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

## Remarks

### 4.5.9 get\_io\_input\_address

```
int get_io_input_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_io\_input\_address*는 I/O 제품의 [입력포트 주소]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[입력포트 주소]가 저장될 포인터입니다.

## Return values

*get\_io\_input\_address*는 *out\_address*에 [입력포트 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        memset(buf, 0x00, 8);
    }
}

```

```

        get_io_input_address(out_eztcp_info.mac_address, buf);
        printf("[I/O] Input Port Base Address : %s\r\n", buf);
    }
}

```

## See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

## Remarks

### 4.5.10 get\_io\_output\_address

```
int get_io_output_address(unsigned char *mac_address, char *out_address);
```

## Description

*get\_io\_output\_address*는 I/O 제품의 [출력포트 주소]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[출력포트 주소]가 저장될 포인터입니다.

## Return values

*get\_io\_output\_address*는 *out\_address*에 [출력포트 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    if(is_io(out_eztcp_info.mac_address) == 1)
    {

```

```

        memset(buf, 0x00, 8);
        get_io_output_address(out_eztcp_info.mac_address, buf);
        printf("[I/O] Output Port Base Address : %s\r\n", buf);
    }
}

```

### See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

### Remarks

#### 4.5.11 get\_io\_poll\_interval

```
int get_io_poll_interval(unsigned char *mac_address, char *out_poll_interval);
```

### Description

*get\_io\_poll\_interval*은 I/O 제품의 마스터 [통신 주기]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_poll\_interval*

[통신 주기]가 저장될 포인터입니다.

### Return values

*get\_io\_poll\_interval*은 *out\_poll\_interval*에 마스터 [통신 주기]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    if(is_io(out_eztcp_info.mac_address) == 1)

```

```

{
    memset(buf, 0x00, 8);
    get_io_poll_interval(out_eztcp_info.mac_address, buf);
    printf("[I/O] Poll Interval : %s\r\n", buf);
}
}

```

### See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

### Remarks

#### 4.5.12 get\_io\_slave\_control\_type

```
int get_io_slave_control_type(unsigned char *mac_address);
```

### Description

*get\_io\_slave\_control\_type*은 I/O 제품의 [슬레이브 출력포트 제어방식]을 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_io\_slave\_control\_type*은 I/O 제품의 [슬레이브 출력포트 제어방식]을 반환 하며 그 값은 아래와 같습니다.

0

FC 16 (동시제어)

1

FC 05 (개별제어)

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)

```



```

{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        int ctrl_type = get_io_slave_control_type(out_eztcp_info.mac_address);
        switch(ctrl_type)
        {
            case 0:
                printf("[I/O] FC 16(Multiple).\r\n");
                break;
            case 1:
                printf("[I/O] FC 05(Single).\r\n");
                break;
        }
    }
}

```

### See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

### Remarks

보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

## 4.5.13 get\_io\_master\_control\_type

```
int get_io_master_control_type(unsigned char *mac_address);
```

### Description

*get\_io\_master\_control\_type*은 I/O 제품의 [마스터 출력포트 제어방식]을 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_io\_master\_control\_type*은 I/O 제품의 [마스터 출력포트 제어방식]을 반환 하며 그 값은 아래와 같습니다.

0

AND  
I  
OR  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        int ctrl_type = get_io_master_control_type(out_eztcp_info.mac_address);
        switch(ctrl_type)
        {
            case 0:
                printf("[I/O] AND.\r\n");
                break;
            case 1:
                printf("[I/O] OR.\r\n");
                break;
        }
    }
}
```

## See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

## Remarks

보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

### 4.5.14 get\_io\_tcp\_type

```
int get_io_tcp_type(unsigned char *mac_address);
```

## Description

*get\_io\_tcp\_type*은 I/O 제품의 Modbus/TCP 접속 방법을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_tcp\_type*은 I/O 제품의 Modbus/TCP 접속 방법을 반환 하며 그 값은 아래와 같습니다.

0

수동접속

1

능동접속

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        int ctrl_type = get_io_tcp_type(out_eztcp_info.mac_address);
        switch(ctrl_type)
        {
            case 0:
                printf("[I/O] Passive Connection.\r\n");
                break;
            case 1:
                printf("[I/O] Active Connection.\r\n");
                break;
        }
    }
}
```

## See also

is\_io, get\_io\_modbus, get\_io\_input\_notification, get\_io\_output\_automatic\_initialize, get\_io\_modbus\_type, get\_io\_unit\_id, get\_io\_input\_address, get\_io\_output\_address, get\_io\_poll\_interval, get\_io\_slave\_control\_type, get\_io\_master\_control\_type, get\_io\_tcp\_type, get\_io\_multi\_connection\_number, get\_io\_modbus\_peer\_address, get\_io\_modbus\_peer\_port, get\_io\_modbus\_local\_port.

## Remarks



보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

#### 4.5.15 get\_io\_multi\_connection\_number

```
int get_io_multi_connection_number(unsigned char *mac_address);
```

##### Description

*get\_io\_multi\_connection\_number*은 I/O 제품이 Modbus/TCP 수동접속인 경우 동시 접속 가능한 Modbus/TCP 클라이언트의 개수를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_io\_multi\_connection\_number*은 I/O 제품이 Modbus/TCP 수동접속인 경우 동시 접속 가능한 Modbus/TCP 클라이언트의 개수를 반환하며 그 값은 1 ~ 8입니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

##### Remarks

#### 4.5.16 get\_io\_modbus\_peer\_address

```
int get_io_modbus_peer_address(unsigned char *mac_address, char *out_address);
```

##### Description

*get\_io\_modbus\_peer\_address*는 I/O 제품이 Modbus/TCP 능동접속인 경우 [통신할 주소]를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
`char *out_address`

[통신할 주소]가 저장될 포인터입니다.

### Return values

*get\_io\_modbus\_peer\_address*는 *out\_address*에 [통신할 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && get_io_tcp_type(mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_io_modbus_peer_address(out_eztcp_info.mac_address, buf);
        printf("[I/O] Peer Address : %s\r\n", buf);
    }
}
```

### See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*,  
*get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*,  
*get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*,  
*get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

### Remarks

[통신할 주소]는 NULL을 제외하고 최대 64바이트입니다. 따라서 *out\_address*은 최소 65바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.5.17 *get\_io\_modbus\_peer\_port*

```
int get_io_modbus_peer_port(unsigned char *mac_address, char *out_port);
```

### Description

*get\_io\_modbus\_peer\_port*는 I/O 제품이 Modbus/TCP 능동접속인 경우 [통신할 포트]를 *out\_port*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_port*

[통신할 포트]가 저장될 포인터입니다.

## Return values

*get\_io\_modbus\_peer\_port*는 *out\_port*에 [통신할 포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && get_io_tcp_type(mac_address) == 1)
    {
        memset(buf, 0x00, 65);
        get_io_modbus_peer_port(out_eztcp_info.mac_address, buf);
        printf("[I/O] Peer Port : %s\r\n", buf);
    }
}
```

## See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

## Remarks

### 4.5.18 get\_io\_modbus\_local\_port

```
int get_io_modbus_local_port(unsigned char *mac_address, char *out_port);
```

## Description



*get\_io\_modbus\_local\_port*는 I/O 제품이 Modbus/TCP 수동접속인 경우 Modbus/TCP [제품 로컬포트]를 *out\_port*에 저장합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_port*

Modbus/TCP [제품 로컬포트]가 저장될 포인터입니다.

## Return values

*get\_io\_modbus\_local\_port*는 *out\_port*에 Modbus/TCP [제품 로컬포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && get_io_tcp_type(mac_address) == 0)
    {
        memset(buf, 0x00, 65);
        get_io_modbus_local_port(out_eztcp_info.mac_address, buf);
        printf("[I/O] Local Port : %s\r\n", buf);
    }
}
```

## See also

*is\_io*, *get\_io\_modbus*, *get\_io\_input\_notification*, *get\_io\_output\_automatic\_initialize*, *get\_io\_modbus\_type*, *get\_io\_unit\_id*, *get\_io\_input\_address*, *get\_io\_output\_address*, *get\_io\_poll\_interval*, *get\_io\_slave\_control\_type*, *get\_io\_master\_control\_type*, *get\_io\_tcp\_type*, *get\_io\_multi\_connection\_number*, *get\_io\_modbus\_peer\_address*, *get\_io\_modbus\_peer\_port*, *get\_io\_modbus\_local\_port*.

## Remarks

#### 4.5.19 get\_io\_output\_number

```
int get_io_output_number(unsigned char *mac_address);
```

##### Description

*get\_io\_output\_port\_number*는 I/O 제품에 설치되어 있는 출력포트 개수를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_io\_output\_port\_number*는 I/O 제품에 설치되어 있는 출력포트 개수를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

##### Remarks

#### 4.5.20 get\_io\_input\_number

```
int get_io_input_number(unsigned char *mac_address);
```

##### Description

*get\_io\_input\_port\_number*는 I/O 제품에 설치되어 있는 입력포트 개수를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_io\_input\_port\_number*는 I/O 제품에 설치되어 있는 입력포트 개수를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.



## Examples

### See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

### Remarks

#### 4.5.21 get\_io\_event\_notification

```
int get_io_event_notification(unsigned char *mac_address);
```

### Description

*get\_io\_event\_notification*은 I/O 제품의 [입력 또는 출력포트 변경 알림(전자메일)] 기능 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_io\_event\_notification*는 [입력 또는 출력포트 변경 알림(전자메일)] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

### Remarks

#### 4.5.22 get\_event\_notification\_email

```
int get_event_notification_email(unsigned char *mac_address, char *out_email);
```

## Description

*get\_event\_notification\_email*은 I/O 제품의 입력 또는 출력포트에 변경이 발생한 경우 알림을 발송할 [전자메일 주소]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_email*

[전자메일 주소]가 저장될 포인터입니다.

## Return values

*get\_event\_notification\_email*은 *out\_email*에 [전자메일 주소]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && is_event_notification(mac_address) == 1)
    {
        if(get_io_event_notification(mac_address) == 1)
        {
            memset(buf, 0x00, 65);
            get_event_notification_email(out_eztcp_info.mac_address, buf);
            printf("[I/O] Email Address : %s\r\n", buf);
        }
    }
}
```

## See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

## Remarks



[전자메일 주소]는 NULL을 포함하여 최대 64바이트입니다. 따라서 *out\_email*은 최소 64바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.5.23 get\_io\_event\_notification\_port

```
int get_io_event_notification_port(unsigned char *mac_address, int port_flag,
                                   int port_index);
```

#### Description

*get\_io\_event\_notification\_port*는 I/O 제품에 설치된 입력 또는 출력포트의 [입력 또는 출력포트 변경 알림(전자메일)] 기능 선택 여부를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소입니다.

*int port\_flag*

*port\_flag*가 0이면 입력포트를 의미하고 1이면 출력포트를 의미합니다.

*int port\_index*

입력 또는 출력포트의 순번으로 0부터 시작합니다.

#### Return values

*get\_io\_event\_notification\_port*는 *port\_flag*와 *port\_index*가 가리키는 입력 또는 출력포트에 [입력 또는 출력포트 변경 알림(전자메일)] 기능이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx = 0;
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && is_event_notification(mac_address) == 1)
    {
        if(get_io_event_notification(mac_address) == 1)
        {
            int input_number = get_io_input_number(mac_address);
```

```

        int output_number = get_io_output_number(mac_address);

        for(idx = 0; idx < input_number; idx++)
        {
            if(get_io_event_notification_port(mac_address, 0, idx) == 1)
            {
                printf("[I/O] Input port#%d is enabled\r\n", idx);
            }
            else
            {
                printf("[I/O] Input port#%d is disabled\r\n", idx);
            }
        }

        for(idx = 0; idx < output_number; idx++)
        {
            if(get_io_event_notification_port(mac_address, 1, idx) == 1)
            {
                printf("[I/O] Output port#%d is enabled\r\n", idx);
            }
            else
            {
                printf("[I/O] Output port#%d is disabled\r\n", idx);
            }
        }
    }
}

```

### See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

### Remarks

#### 4.5.24 get\_io\_input\_valid\_time

```

int get_io_input_valid_time(unsigned char *mac_address, int port_index,
                           char *out_input_valid_time);

```

### Description



*get\_io\_input\_valid\_time*은 *port\_index*로 지정된 입력포트의 [신호유지 시간(밀리초)]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

입력포트의 순번으로 0부터 시작합니다.

*char \*out\_input\_valid\_time*

*port\_index*로 지정된 입력포트의 [신호유지 시간(밀리초)]가 저장될 포인터입니다.

## Return values

*get\_io\_input\_valid\_time*은 *out\_input\_valid\_time*에 [신호유지 시간(밀리초)]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    int idx = 0;
    int input_number = get_io_input_number(out_eztcp_info.mac_address);
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        for(idx = 0; idx < input_number; idx++)
        {
            memset(buf, 0x00, 8);
            get_io_input_valid_time(out_eztcp_info.mac_address, idx, buf);
            printf("[I/O] Input Port#%d Valid Time(ms) : %s\r\n", idx, buf);
        }
    }
}
```

## See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

## Remarks

### 4.5.25 get\_io\_macro\_type

```
int get_io_macro_type(unsigned char *mac_address);
```

## Description

*get\_io\_macro\_type*은 I/O 제품이 지원하는 매크로 기능의 종류를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_macro\_type*은 I/O 제품에 설치된 출력포트에 매크로 기능을 각각 선택할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*get\_io\_macro\_type*이 0을 반환하는 I/O 제품은 매크로 기능을 선택하면 모든 출력포트가 매크로 기능을 사용하게 됩니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

## Remarks

### 4.5.26 get\_io\_macro

```
int get_io_macro(unsigned char *mac_address);
```

## Description

*get\_io\_macro*는 I/O 제품의 [매크로 기능] 선택 여부를 반환합니다. *get\_io\_macro\_type*이 0을 반환하는 경우에만 사용할 수 있습니다.

## Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_io\_macro*는 I/O 제품이 [매크로 기능]이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        if(get_io_macro_type(out_eztcp_info.mac_address) == 0)
        {
            if(get_io_macro(out_eztcp_info.mac_address) == 1)
                printf("[I/O] Macro is enabled.\r\n");
            else
                printf("[I/O] Macro is disabled.\r\n");
        }
    }
}
```

## See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

## Remarks

### 4.5.27 get\_io\_port\_macro

```
int get_io_port_macro(unsigned char *mac_address, int port_index);
```

## Description

*get\_io\_port\_macro*는 *port\_index*로 지정된 출력포트의 [매크로 기능] 선택 여부를 반환합니다. *get\_io\_macro\_type*이 1을 반환하는 경우에만 사용할 수 있습니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

## Return values

*get\_io\_port\_macro*는 *port\_index*로 지정된 출력포트에 [매크로 기능]이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx = 0;
    int output_number = 0;
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && get_io_macro_type(mac_address) == 1)
    {
        output_number = get_io_output_number(mac_address);
        for(idx = 0; idx < output_number; idx++)
        {
            if(get_io_port_macro(mac_address, idx) == 1)
                printf("[I/O] Output Port#%d Macro is enabled.\r\n", idx);
            else
                printf("[I/O] Output Port#%d Macro is disabled.\r\n", idx);
        }
    }
}
```

## See also

*is\_io*, *get\_io\_output\_number*, *get\_io\_input\_number*, *is\_event\_notification*, *get\_io\_event\_notification*, *get\_event\_notification\_email*, *get\_io\_event\_notification\_port*, *get\_io\_input\_valid\_time*, *get\_io\_macro\_type*, *get\_io\_macro*, *get\_io\_port\_macro*, *get\_io\_macro\_text*, *get\_io\_output\_delay*, *get\_io\_output\_initial\_state*, *get\_io\_comment*.

## Remarks





### 4.5.28 get\_io\_macro\_text

```
int get_io_macro_text(unsigned char *mac_address, int port_index, char *out_macro_text);
```

#### Description

*get\_io\_macro\_text*는 *port\_index*로 지정된 출력포트의 매크로 문자열을 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*char \*out\_macro\_text*

*port\_index*로 지정된 출력포트의 매크로 문자열이 저장될 포인터입니다.

#### Return values

*get\_io\_macro\_text*는 *out\_macro\_text*에 매크로 문자열을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx = 0;
    int output_number = 0;
    char buf[IO_SCRIPT_LEN + 1];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1)
    {
        output_number = get_io_output_number(mac_address);
        for(idx = 0; idx < output_number; idx++)
        {
            memset(buf, 0x00, IO_SCRIPT_LEN + 1);
            get_io_macro_text(mac_address, idx, buf);
            printf("[I/O] Output port#%d Macro : %s\r\n", idx, buf);
        }
    }
}
```

}

**See also**

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

**Remarks**

매크로 문자열은 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_macro\_text*는 최소 33 바이트의 메모리 공간을 가지고 있어야 합니다.

매크로 문자열 길이는 **IO\_SCRIPT\_LEN**에 정의 되어 있습니다.

**4.5.29 get\_io\_output\_delay**

```
int get_io_output_delay(unsigned char *mac_address, int port_index,
                        char *out_output_delay);
```

**Description**

*get\_io\_output\_delay*는 *port\_index*로 지정된 출력포트의 [출력지연(밀리초)]을 반환합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*char \*out\_output\_delay*

*port\_index*로 지정된 출력포트의 [출력지연(밀리초)]가 저장될 포인터입니다.

**Return values**

*get\_io\_output\_delay*는 *out\_output\_delay*에 [출력지연(밀리초)]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples**

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
```

```

char buf[8];
int idx = 0;
int output_number = get_io_output_number(out_eztcp_info.mac_address);
if(is_io(out_eztcp_info.mac_address) == 1)
{
    for(idx = 0; idx < output_number; idx++)
    {
        memset(buf, 0x00, 8);
        get_io_output_delay(out_eztcp_info.mac_address, idx, buf);
        printf("[I/O] Output Port#%d Delay(ms) : %s\r\n", idx, buf);
    }
}
}

```

### See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

### Remarks

#### 4.5.30 get\_io\_output\_initial\_state

```
int get_io_output_initial_state(unsigned char *mac_address, int port_index);
```

### Description

*get\_io\_output\_initial\_state*는 *port\_index*로 지정된 출력포트의 [초기상태]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

### Return values

*get\_io\_output\_initial\_state*는 출력포트의 [초기상태]를 반환하며 그 값은 아래와 같습니다.

0

OFF

1

ON

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx = 0;
    unsigned char *mac_address = out_eztcp_info.mac_address;
    int output_number = get_io_output_number(mac_address);
    if(is_io(out_eztcp_info.mac_address) == 1)
    {
        for(idx = 0; idx < output_number; idx++)
        {
            if(get_io_output_initial_state(mac_address, idx) == 1)
                printf("[I/O] Output Port#%d initial state is ON\r\n");
            else
                printf("[I/O] Output Port#%d initial state is OFF\r\n");
        }
    }
}
```

## See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

## Remarks

### 4.5.31 get\_io\_comment

```
int get_io_comment(unsigned char *mac_address, int port_flag, int port_index,
                  char *out_comment);
```

## Description

*get\_io\_comment*는 I/O 제품에 설치된 입력 또는 출력포트의 **[I/O 포트 설명]**을 반환합니다.

## Parameters

*unsigned char \*mac\_address*



*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
*int port\_flag*

*port\_flag*가 0이면 입력포트를 의미하고 1이면 출력포트를 의미합니다.

*int port\_index*

입력 또는 출력포트의 순번으로 0부터 시작합니다.

*char \*out\_comment*

*port\_index*로 지정된 입력 또는 출력포트의 **[I/O 포트 설명]**이 저장될 포인터입니다.

## Return values

*get\_io\_comment*는 *out\_comment*에 *port\_flag*와 *port\_index*가 가리키는 입력 또는 출력포트의 **[I/O 포트 설명]**을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx = 0;
    char buf[IO_COMMENT_LEN + 1];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_io(mac_address) == 1 && is_event_notification(mac_address) == 1)
    {
        if(get_io_event_notification(mac_address) == 1)
        {
            int input_number = get_io_input_number(mac_address);
            int output_number = get_io_output_number(mac_address);

            for(idx = 0; idx < input_number; idx++)
            {
                memset(buf, 0x00, IO_COMMENT_LEN + 1);
                get_io_comment(mac_address, 0, idx, buf);
                printf("[I/O] Input port#%d Comment : %s\r\n", idx, buf);
            }

            for(idx = 0; idx < output_number; idx++)
            {
                memset(buf, 0x00, IO_COMMENT_LEN + 1);
                get_io_comment(mac_address, 1, idx, buf);
                printf("[I/O] Output port#%d Comment : %s\r\n", idx, buf);
            }
        }
    }
}
```

```

    }
}
}
}

```

## See also

is\_io, get\_io\_output\_number, get\_io\_input\_number, is\_event\_notification, get\_io\_event\_notification, get\_event\_notification\_email, get\_io\_event\_notification\_port, get\_io\_input\_valid\_time, get\_io\_macro\_type, get\_io\_macro, get\_io\_port\_macro, get\_io\_macro\_text, get\_io\_output\_delay, get\_io\_output\_initial\_state, get\_io\_comment.

## Remarks

[I/O 포트 설명]은 NULL을 제외하고 최대 16바이트입니다. 따라서 *out\_comment*는 최소 17바이트의 메모리 공간을 가지고 있어야 합니다.

[I/O 포트 설명] 길이는 **IO\_COMMENT\_LEN**에 정의 되어 있습니다.

## 4.6 무선랜

### 4.6.1 get\_wlan\_type

```
int get_wlan_type(unsigned char *mac_address);
```

## Description

*get\_wlan\_type*은 무선랜 제품의 [무선랜 종류]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_type*은 무선랜 제품의 [무선랜 종류]를 반환 하며 그 값은 아래와 같습니다.

0

애드혹

1

인프라스트럭처

2

Soft AP

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        int wlan_type = get_wlan_type(mac_address);
        if(wlan_type == 0)
            printf("[WLAN] WLAN Topology : Ad-hoc\r\n");
        else if(wlan_type == 1)
            printf("[WLAN] WLAN Topology : Infrastructure\r\n");
        else if(is_wlan_soft_ap(mac_address) == 1 && wlan_type == 2)
            printf("[WLAN] WLAN Topology : Soft AP\r\n");
    }
}

```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, get\_wlan\_antenna, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

### 4.6.2 get\_wlan\_channel

```
int get_wlan_channel(unsigned char *mac_address);
```

## Description

*get\_wlan\_channel*은 무선랜 제품의 [채널]을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_channel*은 무선랜 제품의 [채널]을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        int wlan_type = get_wlan_type(mac_address);
        if(wlan_type == 0 || (is_wlan_soft_ap(mac_address) == 1 && wlan_type == 2))
            printf("[WLAN] Channel : %d\r\n", get_wlan_channel(mac_address));
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, get\_wlan\_antenna,  
 get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection,  
 get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type,  
 get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type,  
 get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase,  
 get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id,  
 get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

### 4.6.3 get\_wlan\_ssid

```
int get_wlan_ssid(unsigned char *mac_address, char *out_ssid);
```

## Description

*get\_wlan\_ssid*는 무선랜 제품의 [SSID]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.



*char \*out\_ssid*

무선랜 제품의 [SSID]가 저장될 포인터입니다.

### Return values

*get\_wlan\_ssid*는 *out\_ssid*에 무선랜 제품의 [SSID]를 저장하는데 성공하면 1을 반환합니다. *mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[33];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        memset(buf, 0x00, 33);
        get_wlan_ssid(mac_address, buf);
        printf("[WLAN] SSID : %s\r\n", buf);
    }
}
```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, get\_wlan\_antenna, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

### Remarks

[SSID]는 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_ssid*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.6.4 get\_wlan\_antenna

```
int get_wlan_antenna(unsigned char *mac_address);
```

### Description



*get\_wlan\_antenna*는 무선랜 제품의 [안테나]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_antenna*는 무선랜 제품의 [안테나]를 반환 하며 그 값은 아래와 같습니다.

0

내장안테나

1

외장안테나

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_antenna(mac_address) == 1)
    {
        if(get_wlan_antenna(mac_address) == 0)
            printf("[WLAN] Internal Antenna.\r\n");
        else
            printf("[WLAN] External Antenna.\r\n");
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna,  
 get\_wlan\_antenna, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog,  
 get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type,  
 get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index,  
 get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length,  
 get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise,  
 get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks



### 4.6.5 get\_wlan\_phy\_mode

```
int get_wlan_phy_mode(unsigned char *mac_address);
```

#### Description

*get\_wlan\_phy\_mode*는 무선랜 제품의 무선 고급설정들 중에서 **[Phy Mode]**를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_wlan\_phy\_mode*는 무선랜 제품의 무선 고급설정들 중에서 **[Phy Mode]**를 반환 하며 그 값은 아래와 같습니다.

1

802.11

2

802.11b

3

802.11b/g

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_phy_mode(mac_address) == 1)
    {
        int phy_mode = get_wlan_phy_mode(mac_address);

        if(phy_mode == 1 )
            printf("[WLAN] Phy mode : 802.11\r\n");
        else if(phy_mode == 2)
            printf("[WLAN] Phy mode : 802.11b\r\n");
        else if(phy_mode == 3)
            printf("[WLAN] Phy mode : 802.11b/g\r\n");
    }
}
```

```
}

```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd,

## Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.6.6 get\_wlan\_short\_preamble

```
int get_wlan_short_preamble(unsigned char *mac_address);
```

## Description

*get\_wlan\_short\_preamble*은 무선랜 제품의 무선 고급설정들 중에서 [Short Preamble] 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_short\_preamble*은 무선랜 제품의 무선 고급설정들 중에서 [Short Preamble]이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_phy_mode(mac_address) == 1)
    {
        int phy_mode = get_wlan_phy_mode(mac_address);
    }
}
```

```

        if(phy_mode == 2 || phy_mode == 3)
        {
            if(get_wlan_short_preamble(mac_address) == 1)
                printf("[WLAN] Short Preamble is enabled.\r\n");
            else
                printf("[WLAN] Short Preamble is disabled.\r\n");
        }
    }
}

```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key.

### Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.6.7 get\_wlan\_short\_slot

```
int get_wlan_short_slot(unsigned char *mac_address);
```

### Description

*get\_wlan\_short\_slot*은 무선랜 제품의 무선 고급설정들 중에서 **[Short Slot]** 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_wlan\_short\_slot*은 무선랜 제품의 무선 고급설정들 중에서 **[Short Slot]**이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples



```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_phy_mode(mac_address) == 1)
    {
        int phy_mode = get_wlan_phy_mode(mac_address);

        if(phy_mode == 3)
        {
            if(get_wlan_short_slot(mac_address) == 1)
                printf("[WLAN] Short Slot is enabled.\r\n");
            else
                printf("[WLAN] Short Slot is disabled.\r\n");
        }
    }
}

```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key.

### Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 4.6.8 get\_wlan\_cts\_protection

```
int get_wlan_cts_protection(unsigned char *mac_address);
```

### Description

*get\_wlan\_cts\_protection*은 무선랜 제품의 무선 고급설정들 중에서 [CTS Protection] 선택 여부를 반환합니다.

### Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_cts\_protection*은 무선랜 제품의 무선 고급설정들 중에서 [CTS Protection]이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_phy_mode(mac_address) == 1)
    {
        int phy_mode = get_wlan_phy_mode(mac_address);

        if(phy_mode == 3)
        {
            if(get_wlan_cts_protection(mac_address) == 1)
                printf("[WLAN] CTS Protection is enabled.\r\n");
            else
                printf("[WLAN] CTS Protection is disabled.\r\n");
        }
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key.

## Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 4.6.9 get\_wlan\_background\_scan

```
int get_wlan_background_scan(unsigned char *mac_address);
```

### Description

*get\_wlan\_background\_scan*은 무선랜 제품의 무선 고급설정들 중에서 **[Background Scan]** 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_wlan\_background\_scan*은 무선랜 제품의 무선 고급설정들 중에서 **[Background Scan]**이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_background_scan(mac_address) == 1)
    {
        if(get_wlan_background_scan(mac_address) == 1)
            printf("[WLAN] Background Scan is enabled.\r\n");
        else
            printf("[WLAN] Background Scan is disabled.\r\n");
    }
}
```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna,  
get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble,  
get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan,  
get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wpa\_enterprise,  
get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_wep\_key\_length,  
get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key,  
get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key.



## Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.6.10 get\_wlan\_encryption\_type

```
int get_wlan_encryption_type(unsigned char *mac_address);
```

## Description

*get\_wlan\_encryption\_type*은 무선랜 제품의 [암호화 방식]을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_encryption\_type*은 무선랜 제품의 [암호화 방식]을 반환 하며 그 값은 아래와 같습니다.

0

없음

1

WEP

2

WPA

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wlan\_rsn0*이 1을 반환하는 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 0)
    {
        int enc_type = get_wlan_encryption_type(mac_address);
        if(enc_type == 0)
            printf("[WLAN] Encryption : None.\r\n");
        else if(enc_type == 1)
```

```

        printf("[WLAN] Encyption : WEP.\r\n");
    else if(enc_type == 2)
        printf("[WLAN] Encyption : WPA.\r\n");
    }
}

```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, is\_wlan\_rsn, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key.

## Remarks

### 4.6.11 get\_wlan\_authentication\_type

```
int get_wlan_authentication_type(unsigned char *mac_address);
```

## Description

*get\_wlan\_authentication\_type*은 무선랜 제품의 [인증 방식] 또는 WPA [인증 방식/암호화 방법]을 반환합니다. *get\_wlan\_authentication\_type*이 반환 값은 *get\_wlan\_encryption\_type*의 반환 값에 따라서 의미가 달라집니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_encryption\_type*이 0(없음)을 반환하는 경우 *get\_wlan\_authentication\_type*은 항상 1을 반환합니다.

1

개방 모드

*get\_wlan\_encryption\_type*이 1(WEP)을 반환하는 경우 *get\_wlan\_authentication\_type*은 WEP [인증 방식]을 반환하며 그 값은 아래와 같습니다.

1

개방 모드

2

공유 모드

3

자동 모드

*get\_wlan\_encryption\_type*이 2(WPA)를 반환하는 경우 *get\_wlan\_authentication\_type*은 WPA [인증 방식/암호화 방법]을 반환하며 그 값은 아래와 같습니다.

0

WPA PSK – TKIP

1

WPA PSK – AES

2

WPA PSK – TKIP/AES

3

WPA2 PSK – TKIP

4

WPA2 PSK – AES

5

WPA2 PSK- TKIP/AES

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wlan\_rsn*이 1을 반환하는 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 0)
    {
        int enc_type = get_wlan_encryption_type(mac_address);
        int auth_type = get_wlan_authentication_type(mac_address);
        if(enc_type == 0)
        {
            printf("[WLAN] Encryption : None.\r\n");
            if(auth_type == 1)
                printf("[WLAN] Authentication : Open System.\r\n");
        }
        else if(enc_type == 1)
```

```

        {
            printf("[WLAN] Encyption : WEP.\r\n");
            if(auth_type == 1)
                printf("[WLAN] Authentication : Open System.\r\n");
            else if(auth_type == 2)
                printf("[WLAN] Authentication : Shared Key.\r\n");
            else if(auth_type == 3)
                printf("[WLAN] Authentication : Auth.\r\n");
        }
        else if(enc_type == 2)
        {
            printf("[WLAN] Encyption : WPA.\r\n");
            if(auth_type == 0)
                printf("[WLAN] Authentication : WPA PSK - TKIP.\r\n");
            else if(auth_type == 1)
                printf("[WLAN] Authentication : WPA PSK - AES.\r\n");
            else if(auth_type == 2)
                printf("[WLAN] Authentication : WPA PSK - TKIP/AES.\r\n");
            else if(auth_type == 3)
                printf("[WLAN] Authentication : WPA2 PSK - TKIP.\r\n");
            else if(auth_type == 4)
                printf("[WLAN] Authentication : WPA2 PSK - AES.\r\n");
            else if(auth_type == 5)
                printf("[WLAN] Authentication : WPA2 PSK - TKIP/AES.\r\n");
        }
    }
}

```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, is\_wlan\_rsn, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_shared\_key.

## Remarks

### 4.6.12 get\_wlan\_wep\_key\_length

```
int get_wlan_wep_key_length(unsigned char *mac_address);
```

## Description

*get\_wlan\_wep\_key\_length*은 무선랜 제품의 WEP 키 길이를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_wep\_key\_length*은 무선랜 제품의 WEP 키 길이를 반환하며 그 값은 아래와 같습니다.

1

64비트

2

128비트

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        int wep_type = get_wlan_wep_key_length(mac_address);
        if(wep_type == 1)
            printf("[WLAN] WEP Key : 64Bit.\r\n");
        else if(wep_type == 2)
            printf("[WLAN] WEP Key : 128Bit.\r\n");
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd, get\_wlan\_shared\_key.

## Remarks

### 4.6.13 get\_wlan\_wep\_key\_index

```
int get_wlan_wep_key_index(unsigned char *mac_address);
```

## Description

*get\_wlan\_wep\_key\_index*은 무선랜 제품이 사용하는 WEP 키 순번을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_wep\_key\_index*는 무선랜 제품이 사용하는 WEP 키 순번을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        int key_index = get_wlan_wep_key_index(mac_address);
        printf("[WLAN] WEP Key index %d is using now.\r\n", key_index);
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna,  
get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble,  
get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan,  
get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length,  
get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key,  
get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key,  
get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks



#### 4.6.14 get\_wlan\_wep\_key\_data\_type

```
int get_wlan_wep_key_data_type(unsigned char *mac_address, int wep_key_index);
```

##### Description

*get\_wlan\_wep\_key\_data\_type*은 *wep\_key\_index*로 지정된 WEP 키가 저장된 데이터 형식을 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int wep\_key\_index*

WEP 키 순번으로 0 ~ 3입니다.

##### Return values

*get\_wlan\_wep\_key\_data\_type*은 *wep\_key\_index*로 지정된 WEP 키가 저장된 데이터 형식을 반환하며 그 값은 아래와 같습니다.

0

16진수

1

ASCII

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

*is\_wlan*, *is\_wlan\_soft\_ap*, *get\_wlan\_type*, *get\_wlan\_channel*, *get\_wlan\_ssid*, *is\_wlan\_antenna*,  
*get\_wlan\_antenna*, *is\_wlan\_phy\_mode*, *get\_wlan\_phy\_mode*, *get\_wlan\_short\_preamble*,  
*get\_wlan\_short\_slog*, *get\_wlan\_cts\_protection*, *is\_wlan\_background\_scan*, *get\_wlan\_background\_scan*,  
*get\_wlan\_encryption\_type*, *get\_wlan\_authentication\_type*, *get\_wlan\_wep\_key\_length*,  
*get\_wlan\_wep\_key\_index*, *get\_wlan\_wep\_key\_data\_type*, *get\_wlan\_wep\_key*,  
*get\_wlan\_max\_wpa\_passphrase\_length*, *get\_wlan\_wpa\_passphrase*, *get\_wlan\_shared\_key*,  
*get\_wlan\_wpa\_enterprise*, *get\_wlan\_wpa\_enterprise\_id*, *get\_wlan\_wpa\_enterprise\_pwd*.

##### Remarks

#### 4.6.15 get\_wlan\_wep\_key

```
int get_wlan_wep_key(unsigned char *mac_address, char *out_wep_key);
```

## Description

*get\_wlan\_wep\_key*는 현재 무선랜 제품이 사용 중인 WEP 키를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_wep\_key*

WEP 키가 저장될 포인터입니다.

## Return values

*get\_wlan\_wep\_key*는 *out\_wep\_key*에 무선랜 제품이 사용중인 WEP 키를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int wep_type;
    int key_index;
    int key_type;
    char buf[32];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1)
    {
        wep_type = get_wlan_wep_key_length(mac_address);
        if(wep_type == 1)
            printf("[WLAN] WEP Key : 64Bit.\r\n");
        else if(wep_type == 2)
            printf("[WLAN] WEP Key : 128Bit.\r\n");

        key_index = get_wlan_wep_key_index(mac_address);

        key_type = get_wlan_wep_key_data_type(mac_address, key_index);
        if(key_type == 0)
            printf("[WLAN] WEP Key index %d is hex format\r\n", key_index);
        else if(key_type == 1)
            printf("[WLAN] WEP Key index %d is ASCII format\r\n", key_index);
    }
}
```



```

        memset(buf, 0x00, 32);
        get_wlan_wep_key(mac_address, buf);
        printf("[WLAN] WEP Key index %d is %s\r\n", key_index, buf);
    }
}

```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

### Remarks

*out\_wep\_key*에는 WEP 키 길이가 128비트, 저장된 데이터 형식이 16진수인 경우 NULL을 제외하고 최대 26바이트가 저장됩니다. 따라서 *out\_wep\_key*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.

## 4.6.16 get\_wlan\_max\_wpa\_passphrase\_length

```
int get_wlan_max_wpa_passphrase_length(unsigned char *mac_address);
```

### Description

*get\_wlan\_max\_wpa\_passphrase\_length*는 무선랜 제품에 저장가능한 WPA 암호문 최대길이를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_wlan\_max\_wpa\_passphrase\_length*는 무선랜 제품에 저장가능한 WPA 암호문 최대길이를 반환합니다. 그 값은 32 또는 64입니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna,

get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble,  
 get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan,  
 get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length,  
 get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key,  
 get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key,  
 get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

*get\_wlan\_max\_wpa\_passphrase\_length*의 반환 값은 NULL이 포함된 WPA 암호문의 최대 길이이므로 실제로 사용할 수 있는 WPA 암호문의 최대 길이는 *get\_wlan\_max\_wpa\_passphrase\_length*의 반환 값보다 1바이트 작습니다.

### 4.6.17 get\_wlan\_wpa\_passphrase

```
int get_wlan_wpa_passphrase(unsigned char *mac_address, char *out_wpa_passphrase);
```

## Description

*get\_wlan\_wpa\_passphrase*는 무선랜 제품의 WPA 암호문을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_wpa\_passphrase*

WPA 암호문이 저장될 포인터입니다.

## Return values

*get\_wlan\_wpa\_passphrase*는 *out\_wpa\_passphrase*에 무선랜 제품의 WPA 암호문을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wlan\_rsn*이 1을 반환하는 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    int max_len;
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 0)
```

```

{
    max_len = get_wlan_max_wpa_passphrase_length(mac_address);
    printf("[WLAN] Maximum WPA passphrase length : %d\r\n", max_len);

    memset(buf, 0x00, 65);
    get_wlan_wpa_passphrase(mac_address, buf);
    printf("[WLAN] WPA passphrase : %s.\r\n", buf);
}
}

```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, is\_wlan\_rsn, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

WPA 암호문은 NULL을 포함해서 최대 64바이트입니다. 따라서 *out\_wpa\_passphrase*는 최소 64바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.6.18 get\_wlan\_shared\_key

```
int get_wlan_shared_key(unsigned char *mac_address, char *out_shared_key);
```

## Description

*get\_wlan\_shared\_key*는 무선랜 제품의 보안 설정 중에서 [Shared Key]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_shared\_key*

[Shared Key]가 저장될 포인터입니다.

## Return values

*get\_wlan\_shared\_key*는 *out\_shared\_key*에 무선랜 제품의 [Shared Key]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wlan\_rsn*이 1을 반환하지 않는 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[64];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 1)
    {
        memset(buf, 0x00, 64);
        get_wlan_shared_key(mac_address, buf);
        printf("[WLAN] Security Settings - Shared Key : %s\r\n", buf);
    }
}
```

## See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, is\_wlan\_rsn, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

[Shared Key]는 NULL을 포함해서 최대 64바이트입니다. 따라서 *out\_shared\_key*는 최소 64바이트의 메모리 공간을 가지고 있어야 합니다.

### 4.6.19 get\_wlan\_wpa\_enterprise

```
int get_wlan_wpa_enterprise(unsigned char *mac_address);
```

## Description

*get\_wlan\_wpa\_enterprise*는 무선랜 제품의 보안 설정 중에서 [802.1X] 설정 값을 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_wlan\_wpa\_enterprise*는 무선랜 제품의 [802.1X] 설정 값을 반환하며 그 값은 아래와 같습니다.

0

사용안함

1

EAP TLS

2

EAP TTLS

3

PEAP

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wpa\_enterprise*가 1을 반환하지 않는 무선랜 제품인 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int _802_1x;
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 1)
    {
        if(is_wpa_enterprise(mac_address) == 1)
        {
            _802_1x = get_wlan_wpa_enterprise(mac_address);
            if(_802_1x == 0)
                printf("[WLAN] 802.1X : Disable.\r\n");
            else if(_802_1x == 1)
                printf("[WLAN] 802.1X : EAP TLS.\r\n");
            else if(_802_1x == 2)
                printf("[WLAN] 802.1X : EAP TTLS.\r\n");
            else if(_802_1x == 3)
                printf("[WLAN] 802.1X : PEAP.\r\n");
        }
    }
}
```

## See also



is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna,  
 get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble,  
 get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan,  
 is\_wlan\_rsn, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length,  
 get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key,  
 get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key,  
 is\_wpa\_enterprise, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

## Remarks

### 4.6.20 get\_wlan\_wpa\_enterprise\_id

```
int get_wlan_wpa_enterprise_id(unsigned char *mac_address, char *out_enterprise_id);
```

## Description

*get\_wlan\_wpa\_enterprise\_id*는 무선랜 제품의 보안 설정 중에서 802.1X [아이디]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_enterprise\_id*

802.1X [아이디]가 저장될 포인터입니다.

## Return values

*get\_wlan\_wpa\_enterprise\_id*는 *out\_enterprise\_id*에 802.1X [아이디]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wpa\_enterprise*가 1을 반환하지 않는 무선랜 제품인 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int _802_1x;
    char buf[33];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 1)
    {
```

```

        if(is_wpa_enterprise(mac_address) == 1)
        {
            _802_1x = get_wlan_wpa_enterprise(mac_address);
            if(_802_1x > 0)
            {
                memset(buf, 0x00, 33);
                get_wlan_wpa_enterprise_id(mac_address, buf);
                printf("[WLAN] 802.1X ID : %s.\r\n", buf);
            }
        }
    }
}

```

### See also

is\_wlan, is\_wlan\_soft\_ap, get\_wlan\_type, get\_wlan\_channel, get\_wlan\_ssid, is\_wlan\_antenna, get\_wlan\_antenna, is\_wlan\_phy\_mode, get\_wlan\_phy\_mode, get\_wlan\_short\_preamble, get\_wlan\_short\_slog, get\_wlan\_cts\_protection, is\_wlan\_background\_scan, get\_wlan\_background\_scan, is\_wlan\_rsn, get\_wlan\_encryption\_type, get\_wlan\_authentication\_type, get\_wlan\_wep\_key\_length, get\_wlan\_wep\_key\_index, get\_wlan\_wep\_key\_data\_type, get\_wlan\_wep\_key, get\_wlan\_max\_wpa\_passphrase\_length, get\_wlan\_wpa\_passphrase, get\_wlan\_shared\_key, is\_wpa\_enterprise, get\_wlan\_wpa\_enterprise, get\_wlan\_wpa\_enterprise\_id, get\_wlan\_wpa\_enterprise\_pwd.

### Remarks

802.1X [아이디]는 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_enterprise\_id*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.6.21 get\_wlan\_wpa\_enterprise\_pwd

```

int get_wlan_wpa_enterprise_pwd(unsigned char *mac_address,
                                char *out_enterprise_pwd);

```

### Description

*get\_wlan\_wpa\_enterprise\_pwd*는 무선랜 제품의 보안 설정 중에서 802.1X [비밀번호]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_enterprise\_pwd*

802.1X [비밀번호]가 저장될 포인터입니다.

### Return values



*get\_wlan\_wpa\_enterprise\_pwd*는 *out\_enterprise\_pwd*에 802.1X [비밀번호]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*is\_wpa\_enterprise*가 1을 반환하지 않는 무선랜 제품인 경우에는 -2를 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int _802_1x;
    char buf[33];
    unsigned char *mac_address = out_eztcp_info.mac_address;
    if(is_wlan(mac_address) == 1 && is_wlan_rsn(mac_address) == 1)
    {
        if(is_wpa_enterprise(mac_address) == 1)
        {
            _802_1x = get_wlan_wpa_enterprise(mac_address);
            if(_802_1x >= 2)
            {
                memset(buf, 0x00, 33);
                get_wlan_wpa_enterprise_pwd(mac_address, buf);
                printf("[WLAN] 802.1X Password : %s.\r\n", buf);
            }
        }
    }
}
```

## See also

*is\_wlan*, *is\_wlan\_soft\_ap*, *get\_wlan\_type*, *get\_wlan\_channel*, *get\_wlan\_ssid*, *is\_wlan\_antenna*, *get\_wlan\_antenna*, *is\_wlan\_phy\_mode*, *get\_wlan\_phy\_mode*, *get\_wlan\_short\_preamble*, *get\_wlan\_short\_slog*, *get\_wlan\_cts\_protection*, *is\_wlan\_background\_scan*, *get\_wlan\_background\_scan*, *is\_wlan\_rsn*, *get\_wlan\_encryption\_type*, *get\_wlan\_authentication\_type*, *get\_wlan\_wep\_key\_length*, *get\_wlan\_wep\_key\_index*, *get\_wlan\_wep\_key\_data\_type*, *get\_wlan\_wep\_key*, *get\_wlan\_max\_wpa\_passphrase\_length*, *get\_wlan\_wpa\_passphrase*, *get\_wlan\_shared\_key*, *is\_wpa\_enterprise*, *get\_wlan\_wpa\_enterprise*, *get\_wlan\_wpa\_enterprise\_id*, *get\_wlan\_wpa\_enterprise\_pwd*.

## Remarks

802.1X [비밀번호]는 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_enterprise\_pwd*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.



## 4.7 부가기능

### 4.7.1 get\_telnet

```
int get_telnet(unsigned char *mac_address);
```

#### Description

*get\_telnet*은 [텔넷] 기능의 선택 여부를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_telnet*은 [텔넷] 기능이 선택되어 있으면 1을 반환하고 그렇지않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*,  
*get\_tcp\_multi\_connection*, *get\_power\_management*, *get\_comment*

#### Remarks

### 4.7.2 get\_send\_mac\_address

```
int get_send_mac_address(unsigned char *mac_address);
```

#### Description

*get\_send\_mac\_address*는 [MAC 주소 전송] 기능의 선택 여부를 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

#### Return values

*get\_send\_mac\_address*는 [MAC 주소 전송] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

`get_telnet`, `is_send_mac_address`, `get_send_mac_address`, `get_ssl`, `get_ssh`, `get_ip4_address_search`, `get_remote_debug`, `get_tcp_multi_connection`, `get_power_management`, `get_comment`

## Remarks

### 4.7.3 get\_ssl

```
int get_ssl(unsigned char *mac_address);
```

## Description

*get\_ssl*은 [SSL 보안통신] 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_ssl*은 [SSL 보안통신] 기능이 선택되어 있으면 1을 반환하고 그렇지않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

`get_telnet`, `get_send_mac_address`, `is_ssl`, `get_ssl`, `get_ssh`, `get_ip4_address_search`, `get_remote_debug`, `get_tcp_multi_connection`, `get_power_management`, `get_comment`

## Remarks

### 4.7.4 get\_ssh

```
int get_ssh(unsigned char *mac_address);
```

## Description



*get\_ssh*는 [SSH 보안통신] 기능의 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_ssh*는 [SSH 보안통신] 기능이 선택되어 있으면 1을 반환하고 그렇지않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *is\_ssh*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*, *get\_tcp\_multi\_connection*, *get\_power\_management*, *get\_comment*

### Remarks

## 4.7.5 get\_ip4\_address\_search

```
int get_ip4_address_search(unsigned char *mac_address);
```

### Description

*get\_ip4\_address\_search*는 [IPv4 주소 검색] 기능의 선택 여부를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_ip4\_address\_search*는 [IPv4 주소 검색] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*,

get\_tcp\_multi\_connection, get\_power\_management, get\_comment

## Remarks

### 4.7.6 get\_remote\_debug

```
int get_remote_debug(unsigned char *mac_address);
```

## Description

*get\_remote\_debug*는 [디버깅 로그 보기] 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_remote\_debug*는 [디버깅 로그 보기] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

get\_telnet, get\_send\_mac\_address, get\_ssl, get\_ssh, get\_ip4\_address\_search, is\_remote\_debug, get\_remote\_debug, get\_tcp\_multi\_connection, get\_power\_management, get\_comment

## Remarks

### 4.7.7 get\_tcp\_multi\_connection

```
int get_tcp_multi_connection(unsigned char *mac_address);
```

## Description

*get\_tcp\_multi\_connection*는 [다중 접속] 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*get\_tcp\_multi\_connection*는 [다중 접속] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*, *is\_tcp\_multi\_connection*, *get\_tcp\_multi\_connection*, *get\_power\_management*, *get\_comment*

**Remarks****4.7.8 get\_power\_management**

```
int get_power_management(unsigned char *mac_address);
```

**Description**

*get\_power\_management*는 [전원 관리] 기능의 선택 여부를 반환합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*get\_power\_management*는 [전원 관리] 기능이 선택되어 있으면 1을 반환하고 그렇지 않은 경우 0을 반환합니다.

*is\_power\_management*가 1을 반환하지 않는 경우 -2를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*, *get\_tcp\_multi\_connection*, *is\_power\_management*, *get\_power\_management*, *get\_comment*

**Remarks**

### 4.7.9 get\_comment

```
int get_comment(unsigned char *mac_address, char *out_comment);
```

#### Description

*get\_comment*는 제품(ezTCP)의 [설명]을 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_comment*

제품(ezTCP)의 [설명]이 저장될 포인터입니다.

#### Return values

*get\_telnet*은 *out\_comment*에 제품(ezTCP)의 [설명]을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[65];
    memset(buf, 0x00, 65);
    get_comment(out_eztcp_info.mac_address, buf);
    printf("[ezTCP] Comment : %s.\r\n", buf);
}
```

#### See also

*get\_telnet*, *get\_send\_mac\_address*, *get\_ssl*, *get\_ssh*, *get\_ip4\_address\_search*, *get\_remote\_debug*,  
*get\_tcp\_multi\_connection*, *get\_power\_management*, *get\_comment*

#### Remarks

제품(ezTCP)의 [설명]은 NULL을 제외하고 최대 64바이트입니다. 따라서 *out\_comment*는 최소 65바이트의 메모리 공간을 가지고 있어야 합니다.

#### 4.7.10 get\_allowed\_ezmanager

```
int get_allowed_ezmanager(unsigned char *mac_address);
```

##### Description

*get\_allowed\_ezmanager*은 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 **[ezManager에도 적용]**의 선택 여부를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_allowed\_ezmanager*은 제품(ezTCP) 접근 제한 기능 중 **[ezManager에도 적용]**이 선택되어 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    if(get_allowed_ezmanager(mac_address) == 1)
        printf("[ezTCP Firewall] Apply To ezManager is enabled.\r\n");
    else
        printf("[ezTCP Firewall] Apply To ezManager is disabled.\r\n");
}
```

##### See also

*get\_allowed\_ezmanager*, *get\_allowed\_mac\_address*, *get\_allowed\_ip4\_address*,  
*get\_allowed\_ip4\_network\_mask\_type*, *get\_allowed\_ip6\_address*, *get\_allowed\_ip6\_subnet\_prefix\_length*

##### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.11 get\_allowed\_mac\_address

```
int get_allowed_mac_address(unsigned char *mac_address, int mac_index,
                           char *out_mac_address);
```

##### Description

*get\_allowed\_mac\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 MAC 주소만 접근 가능]이 선택된 경우 입력되어있는 MAC 주소를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int mac\_index*

입력되어있는 MAC 주소 6바이트 중에서 읽고자 하는 MAC 주소의 순번으로 0 ~ 5입니다.

*char \*out\_mac\_address*

*mac\_index*로 지정된 MAC 주소가 저장될 포인터입니다.

##### Return values

*get\_allowed\_mac\_address*은 제품(ezTCP) 접근 제한 기능 중 [다음의 MAC 주소만 접근 가능]에 입력되어있는 MAC 주소를 *out\_mac*에 저장하는데 성공하면 1을 반환합니다.

*mac\_index*가 지정된 범위를 벗어나면 -2를 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx;
    char buf[3];
    unsigned char *mac_address = out_eztcp_info.mac_address;

    printf("[ezTCP Firewall] Allowed MAC Address : ");
    for(idx = 0; idx < 6; idx++)
    {
        memset(buf, 0x00, 3);
        get_allowed_mac_address(mac_address, idx, buf);
        printf("%s ", buf);
    }
}
```



```
printf("\r\n");
}
```

### See also

get\_allowed\_ezmanager, get\_allowed\_mac\_address, get\_allowed\_ip4\_address,  
get\_allowed\_ip4\_network\_mask\_type, get\_allowed\_ip6\_address, get\_allowed\_ip6\_subnet\_prefix\_length

### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.12 get\_allowed\_ip4\_address

```
int get_allowed_ip4_address(unsigned char *mac_address, char *out_address);
```

### Description

*get\_allowed\_ip4\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 **[다음의 IP 주소 대역만 접근 가능]**의 선택된 경우 입력되어있는 **[IPv4 주소]**를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

IPv4 주소가 저장될 포인터입니다.

### Return values

*get\_allowed\_ip4\_address*은 **[다음의 IP 주소 대역만 접근 가능]**에 입력되어있는 **[IPv4 주소]**를 *out\_address*에 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int idx;
    int networkmask_type;
    char buf[16];
    unsigned char *mac_address = out_eztcp_info.mac_address;
```

```

printf("[ezTCP Firewall] Allowed IP Range\r\n");

memset(buf, 0x00, 16);
get_allowed_ip4_address(mac_address, buf);
printf("IPv4 Address : %s\r\n", buf);

networkmask_type = get_allowed_ip4_network_mask_type(mac_address);
switch(networkmask_type)
{
case 0:
    printf("Network Mask : 255.255.255.255\r\n");
    break;
case 1:
    printf("Network Mask : 255.255.255.0\r\n");
    break;
case 2:
    printf("Network Mask : 255.255.0.0\r\n");
    break;
case 3:
    printf("Network Mask : 255.0.0.0\r\n");
    break;
case 4:
    printf("Network Mask : 0.0.0.0\r\n");
    break;
}
}

```

### See also

get\_allowed\_ezmanager, get\_allowed\_mac\_address, get\_allowed\_ip4\_address,  
get\_allowed\_ip4\_network\_mask\_type, get\_allowed\_ip6\_address, get\_allowed\_ip6\_subnet\_prefix\_length

### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.13 get\_allowed\_ip4\_network\_mask\_type

```
int get_allowed_ip4_network_mask_type(unsigned char *mac_address);
```

### Description

*get\_allowed\_ip4\_network\_mask\_type*은 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 IP 주소 대역만 접근 가능]의 선택된 경우 입력되어있는 [넷 마스크(IPv4 주소 대역)]의

형태를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*get\_allowed\_ip4\_network\_mask\_type*은 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [넷 마스크(IPv4 주소 대역)]의 형태를 반환하며 그 값은 아래와 같습니다.

0

255.255.255.255

1

255.255.255.0

2

255.255.0.0

3

255.0.0.0

4

0.0.0.0

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int network_mask_type;
    char buf[16];
    unsigned char *mac_address = out_eztcp_info.mac_address;

    printf("[ezTCP Firewall] Allowed IP Range\r\n");

    memset(buf, 0x00, 16);
    get_allowed_ip4_address(mac_address, buf);
    printf("IPv4 Address : %s\r\n", buf);

    network_mask_type = get_allowed_ip4_network_mask_type(mac_address);
    switch(networkmask_type)
    {
```

```

    case 0:
        printf("Network Mask : 255.255.255.255\r\n");
        break;
    case 1:
        printf("Network Mask : 255.255.255.0\r\n");
        break;
    case 2:
        printf("Network Mask : 255.255.0.0\r\n");
        break;
    case 3:
        printf("Network Mask : 255.0.0.0\r\n");
        break;
    case 4:
        printf("Network Mask : 0.0.0.0\r\n");
        break;
    }
}

```

### See also

get\_allowed\_ezmanager, get\_allowed\_mac\_address, get\_allowed\_ip4\_address,  
get\_allowed\_ip4\_network\_mask\_type, get\_allowed\_ip6\_address, get\_allowed\_ip6\_subnet\_prefix\_length

### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.14 get\_allowed\_ip6\_address

```
int get_allowed_ip6_address(unsigned char *mac_address, char *out_address);
```

### Description

*get\_allowed\_ip6\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 IP 주소 대역만 접근 가능]의 선택된 경우 입력되어있는 [IPv6 주소]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

IPv6 주소가 저장될 포인터입니다.

## Return values

*get\_allowed\_ip6\_address*은 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [IPv6 주소]를 *out\_address*에 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    unsigned char *mac_address = out_eztcp_info.mac_address;

    printf("[ezTCP Firewall] Allowed IP Range\r\n");

    memset(buf, 0x00, 128);
    get_allowed_ip6_address(mac_address, buf);
    printf("IPv6 Address : %s\r\n", buf);

    memset(buf, 0x00, 128);
    get_allowed_ip6_subnet_prefix_length(mac_address, buf);
    printf("\tSubnet Prefix Length : %s\r\n", buf);
}
```

## See also

*get\_allowed\_ezmanager*, *get\_allowed\_mac\_address*, *get\_allowed\_ip4\_address*,  
*get\_allowed\_ip4\_network\_mask\_type*, *get\_allowed\_ip6\_address*, *get\_allowed\_ip6\_subnet\_prefix\_length*

## Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.7.15 get\_allowed\_ip6\_subnet\_prefix\_length

```
int get_allowed_ip6_subnet_prefix_length(unsigned char *mac_address,
                                         char *out_subnet_prefix_length);
```

## Description

*get\_allowed\_ip6\_subnet\_prefix\_length*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서

[다음의 IP 주소 대역만 접근 가능]이 선택된 경우 입력되어있는 [IPv6 주소]의 서브넷 접두사 길이를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_subnet\_prefix\_length*

서브넷 접두사 길이가 저장될 포인터입니다.

## Return values

*get\_allowed\_ip6\_subnet\_prefix\_length*는 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [IPv6 주소]의 서브넷 접두사 길이를 *out\_subnet\_prefix\_length*에 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[128];
    unsigned char *mac_address = out_eztcp_info.mac_address;

    printf("[ezTCP Firewall] Allowed IP Range\r\n");

    memset(buf, 0x00, 128);
    get_allowed_ip6_address(mac_address, buf);
    printf("IPv6 Address : %s\r\n", buf);

    memset(buf, 0x00, 128);
    get_allowed_ip6_subnet_prefix_length(mac_address, buf);
    printf("\tSubnet Prefix Length : %s\r\n", buf);
}
```

## See also

*get\_allowed\_ezmanager*, *get\_allowed\_mac\_address*, *get\_allowed\_ip4\_address*,  
*get\_allowed\_ip4\_network\_mask\_type*, *get\_allowed\_ip6\_address*, *get\_allowed\_ip6\_subnet\_prefix\_length*

## Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다

니다.

#### 4.7.16 get\_ip4\_change\_notification\_type

```
int get_ip4_change_notification_type(unsigned char *mac_address);
```

##### Description

*get\_ip4\_change\_notification\_type*은 [IPv4 주소 통보]기능의 [프로토콜]을 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

##### Return values

*get\_ip4\_change\_notification\_type*은 [IPv4 주소 통보]기능의 [프로토콜]을 반환하며 그 값은 아래와 같습니다.

0

사용안함

1

DDNS(dyndns.org)

2

TCP

3

UDP

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    printf("[Notify IPv4 Change] Protocol : ");

    int protocol_type = get_ip4_change_notification_type(mac_address);
    switch(protocol_type)
    {
```

```

    case 0:
        printf("Disable\r\n");
        break;

    case 1:
        printf("DDNS(dyndns.org)\r\n");
        break;

    case 2:
        printf("TCP\r\n");
        break;

    case 3:
        printf("UDP\r\n");
        break;

    }
}

```

### See also

get\_ip4\_change\_notification\_type,  
get\_ip4\_change\_notification\_interval,  
get\_ip4\_change\_notification\_peer\_address,  
get\_ip4\_change\_notification\_ddns\_pwd.

get\_ip4\_change\_notification\_data\_type,  
get\_ip4\_change\_notification\_peer\_port,  
get\_ip4\_change\_notification\_ddns\_id,

### Remarks

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.17 get\_ip4\_change\_notification\_data\_type

```
int get_ip4_change_notification_data_type(unsigned char *mac_address);
```

### Description

*get\_ip4\_change\_notification\_data\_type*은 [IPv4 주소 통보]기능의 [데이터 형식]을 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*get\_ip4\_change\_notification\_data\_type*은 [IPv4 주소 통보]기능의 [데이터 형식]을 반환하며 그 값은 아래와 같습니다.

0



ASCII

1

16진수

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    int data_type;
    int protocol_type = get_ip4_change_notification_type(mac_address);
    if(protocol_type == 2 || protocol_type == 3)
    {
        printf("[Notify IPv4 Change] Data Type : ");
        data_type = get_ip4_change_notification_data_type(mac_address);
        if(data_type == 0)
            printf("ASCII\r\n");
        else if(data_type == 1)
            printf("HEX\r\n");
    }
}
```

## See also

get\_ip4\_change\_notification\_type,  
get\_ip4\_change\_notification\_interval,  
get\_ip4\_change\_notification\_peer\_address,  
get\_ip4\_change\_notification\_ddns\_pwd.

get\_ip4\_change\_notification\_data\_type,  
get\_ip4\_change\_notification\_peer\_port,  
get\_ip4\_change\_notification\_ddns\_id,

## Remarks

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.7.18 get\_ip4\_change\_notification\_interval

```
int get_ip4_change_notification_interval(unsigned char *mac_address, char *out_interval);
```

## Description



*get\_ip4\_change\_notification\_interval*은 [IPv4 주소 통보]기능의 [통보 주기]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_interval*

[통보 주기]가 저장될 포인터입니다.

## Return values

*get\_ip4\_change\_notification\_interval*은 *out\_interval*에 IPv4 주소 통보 기능의 [통보 주기]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    char buf[8];
    int protocol_type = get_ip4_change_notification_type(mac_address);
    if(protocol_type == 2 || protocol_type == 3)
    {
        get_ip4_change_notification_interval(mac_address, buf);
        printf("[Notify IPv4 Change] Interval : %s\r\n", buf);
    }
}
```

## See also

*get\_ip4\_change\_notification\_type*,  
*get\_ip4\_change\_notification\_interval*,  
*get\_ip4\_change\_notification\_peer\_address*,  
*get\_ip4\_change\_notification\_ddns\_pwd*.

*get\_ip4\_change\_notification\_data\_type*,  
*get\_ip4\_change\_notification\_peer\_port*,  
*get\_ip4\_change\_notification\_ddns\_id*,

## Remarks

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.19 get\_ip4\_change\_notification\_peer\_port

```
int get_ip4_change_notification_peer_port(unsigned char *mac_address, char *out_port);
```

##### Description

*get\_ip4\_change\_notification\_peer\_port*은 [IPv4 주소 통보]기능의 [포트]를 반환합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_port*

[포트]가 저장될 포인터입니다.

##### Return values

*get\_ip4\_change\_notification\_peer\_port*은 *out\_port*에 [IPv4 주소 통보]기능의 [포트]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    char buf[8];
    int protocol_type = get_ip4_change_notification_type(mac_address);
    if(protocol_type == 2 || protocol_type == 3)
    {
        get_ip4_change_notification_peer_port(mac_address, buf);
        printf("[Notify IPv4 Change] Port Number : %s\r\n", buf);
    }
}
```

##### See also

get\_ip4\_change\_notification\_type,  
get\_ip4\_change\_notification\_interval,  
get\_ip4\_change\_notification\_peer\_address,  
get\_ip4\_change\_notification\_ddns\_pwd.

get\_ip4\_change\_notification\_data\_type,  
get\_ip4\_change\_notification\_peer\_port,  
get\_ip4\_change\_notification\_ddns\_id,

## Remarks

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.7.20 get\_ip4\_change\_notification\_peer\_address

```
int get_ip4_change_notification_peer_address(unsigned char *mac_address,
                                             char *out_address);
```

## Description

*get\_ip4\_change\_notification\_peer\_address*은 [IPv4 주소 통보]기능의 [호스트 이름] 또는 [통보할 주소]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_address*

[호스트 이름] 또는 [통보할 주소]가 저장될 포인터입니다.

## Return values

*get\_ip4\_change\_notification\_peer\_address*는 *out\_address*에 [호스트 이름] 또는 [통보할 주소]를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    char buf[64];
    int protocol_type = get_ip4_change_notification_type(mac_address);
    memset(buf, 0x00, 64);
    get_ip4_change_notification_peer_address(mac_address, buf);

    if(protocol == 1)
    {
        printf("[Notify IPv4 Change] Host Name (dyndns) : %s\r\n", buf);
    }
}
```

```

    }
    else if(protocol_type == 2 || protocol_type == 3)
    {
        printf("[Notify IPv4 Change] Host Name (custom) : %s\r\n", buf);
    }
}

```

## See also

get\_ip4\_change\_notification\_type,  
get\_ip4\_change\_notification\_interval,  
get\_ip4\_change\_notification\_peer\_address,  
get\_ip4\_change\_notification\_ddns\_pwd.

get\_ip4\_change\_notification\_data\_type,  
get\_ip4\_change\_notification\_peer\_port,  
get\_ip4\_change\_notification\_ddns\_id,

## Remarks

[IPv4 주소 통보]기능의 [프로토콜]이 1(DDNS(dyndns.org))인 경우 [호스트 이름]으로 사용되고 2(TCP) 또는 3(UDP)인 경우에만 [통보할 주소]로 사용됩니다.

[호스트 이름] 또는 [통보할 주소]는 NULL을 포함하여 최대 64바이트입니다. 따라서 *out\_address*는 최소 64바이트의 메모리 공간을 가지고 있어야 합니다.

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 4.7.21 get\_ip4\_change\_notification\_ddns\_id

```

int get_ip4_change_notification_ddns_id(unsigned char *mac_address, char
*out_ddns_id);

```

## Description

get\_ip4\_change\_notification\_ddns\_id는 IPv4 주소 통보 기능의 [DDNS 아이디]를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_ddns\_id*

[DDNS 아이디]가 저장될 포인터입니다.

## Return values

*get\_ip4\_change\_notification\_ddns\_id*는 *out\_ddns\_id*에 [DDNS 아이디]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples



```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    char buf[33];
    int protocol_type = get_ip4_change_notification_type(mac_address);
    if(protocol == 1)
    {
        memset(buf, 0x00, 33);
        get_ip4_change_notification_ddns_id(mac_address, buf);

        printf("[Notify IPv4 Change] DDNS ID : %s\r\n", buf);
    }
}

```

### See also

get\_ip4\_change\_notification\_type,  
get\_ip4\_change\_notification\_interval,  
get\_ip4\_change\_notification\_peer\_address,  
get\_ip4\_change\_notification\_ddns\_pwd.

get\_ip4\_change\_notification\_data\_type,  
get\_ip4\_change\_notification\_peer\_port,  
get\_ip4\_change\_notification\_ddns\_id,

### Remarks

[**DDNS 아이디**]는 NULL을 제외하고 최대 32바이트입니다. 따라서 *out\_ddns\_id*는 최소 33바이트의 메모리 공간을 가지고 있어야 합니다.

[**IPv4 주소 통보**]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 4.7.22 get\_ip4\_change\_notification\_ddns\_pwd

```

int get_ip4_change_notification_ddns_pwd(unsigned char *mac_address,
                                          char *out_ddns_pwd);

```

### Description

*get\_ip4\_change\_notification\_ddns\_pwd*는 [**IPv4 주소 통보**]기능의 [**DDNS 비밀번호**]를 반환합니다.

### Parameters

*unsigned char \*mac\_address*



*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
*char \*out\_ddns\_pwd*

[DDNS 비밀번호]가 저장될 포인터입니다.

### Return values

*get\_ip4\_change\_notification\_ddns\_pwd*는 *out\_ddns\_pwd*에 [DDNS 비밀번호]를 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    char buf[17];
    int protocol_type = get_ip4_change_notification_type(mac_address);
    if(protocol == 1)
    {
        memset(buf, 0x00, 17);
        get_ip4_change_notification_ddns_pwd(mac_address, buf);

        printf("[Notify IPv4 Change] DDNS Password : %s\r\n", buf);
    }
}
```

### See also

*get\_ip4\_change\_notification\_type*,  
*get\_ip4\_change\_notification\_interval*,  
*get\_ip4\_change\_notification\_peer\_address*,  
*get\_ip4\_change\_notification\_ddns\_pwd*.

*get\_ip4\_change\_notification\_data\_type*,  
*get\_ip4\_change\_notification\_peer\_port*,  
*get\_ip4\_change\_notification\_ddns\_id*,

### Remarks

[DDNS 비밀번호]는 NULL을 제외하고 최대 16바이트입니다. 따라서 *out\_ddns\_pwd*는 최소 17바이트의 메모리 공간을 가지고 있어야 합니다.

[IPv4 주소 통보]기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5 설정 값 변경

### 5.1 네트워크

#### 5.1.1 set\_arp

```
int set_arp(unsigned char *mac_address, int onoff);
```

##### Description

*set\_arp*는 [처음 수신된 패킷에서 IP 주소 얻기] 설정 값을 변경합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

*0*

기능 선택을 해제합니다.

*1*

기능을 선택합니다.

##### Return values

*set\_arp*는 설정 값 변경에 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

##### Examples

##### See also

##### Remarks

#### 5.1.2 set\_dhcp

```
int set_dhcp(unsigned char *mac_address, int onoff);
```

##### Description

*get\_dhcp*는 [자동으로 IP 주소 받기(DHCP)] 설정 값을 변경합니다.



**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

*0*

기능 선택을 해제합니다.

*1*

기능을 선택합니다.

**Return values***set\_dhcp*는 설정 값 변경에 성공하면 1을 반환합니다.*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.**Examples****See also***set\_dhcp\_dns*, *set\_pppoe***Remarks**

[자동으로 IP 주소 받기(DHCP)]와 [아이디로 IP 주소 받기(PPPoE)]는 동시에 설정할 수 없습니다. 따라서 *set\_dhcp*의 함수 입력값인 *onoff*에 1을 입력하는 경우 [아이디로 IP 주소 받기(PPPoE)]는 자동으로 선택이 해제됩니다.

**5.1.3 set\_dhcp\_dns**

```
int set_dhcp_dns(unsigned char *mac_address, int onoff);
```

**Description***set\_dhcp\_dns*는 [자동으로 DNS 서버 주소 받기] 설정 값을 변경합니다.**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

*0*

기능 선택을 해제합니다.

1

기능을 선택합니다.

### Return values

*set\_dhcp\_dns*는 설정 값 변경에 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

*set\_dhcp*

### Remarks

[**자동으로 DNS 서버 주소 받기**]가 설정된 경우 제품(ezTCP)은 DHCP 서버로부터 할당 받은 DNS 서버 IP 주소를 사용합니다. 그렇지 않은 경우 사용자가 입력한 DNS 서버 IP 주소를 사용합니다.

#### 5.1.4 set\_pppoe

```
int set_pppoe(unsigned char *mac_address, int onoff);
```

### Description

*set\_pppoe*는 [**아이디로 IP 주소 받기(PPPoE)**] 설정 값을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

### Return values

*set\_pppoe*는 설정 값 변경에 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples



**See also**

set\_pppoe\_id, set\_pppoe\_pwd, set\_dhcp

**Remarks**

[아이디로 IP 주소 받기(PPPoE)]와 [자동으로 IP 주소 받기(DHCP)]는 동시에 설정할 수 없습니다. 따라서 *set\_pppoe*의 입력값인 *onoff*에 1을 입력하는 경우 [자동으로 IP 주소 받기(DHCP)]는 자동으로 선택이 해제됩니다.

**5.1.5 set\_ip4\_local\_address**

```
int set_ip4_local_address(unsigned char *mac_address, char *ip4_local_address);
```

**Description**

*set\_ip4\_local\_address*는 제품(ezTCP)의 IPv4 [제품 IP 주소] 설정 값을 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip4\_local\_address*

IPv4 [제품 IP 주소]를 변경할 IPv4 주소가 저장되어있는 포인터입니다.

**Return values**

*set\_ip4\_local\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

제품(ezTCP)이 사용할 수 없는 IP 주소인 경우에는 -2를 반환합니다.

**Examples**

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

char *new_ip4_addr = "10.1.0.2";

if(ret == EZTCP_SUCCESS)
{
    int res = set_ip4_local_address(out_eztcp_info.mac_address, new_ip4_addr);
}
```

```

    if(res == 1)
        printf("Success.\r\n");
    else if(res == -1)
        printf("There is no product information of the requested MAC address\r\n");
    else if(res == -2)
        printf("You entered an incorrect IPv4 address.\r\n");
}

```

### See also

set\_ip4\_local\_address, set\_ip4\_subnet\_mask, set\_ip4\_gateway\_address, set\_ip4\_dns\_address.

### Remarks

*set\_ip4\_local\_address* 입력 값인 *ip4\_local\_address*에는 IPv4의 점으로 구분된 십진수 표기법으로 된 IPv4 주소를 입력해야 합니다.

#### 5.1.6 set\_ip4\_subnet\_mask

```
int set_ip4_subnet_mask(unsigned char *mac_address, char *ip4_subnet_mask);
```

### Description

*set\_ip4\_subnet\_mask*는 제품(ezTCP)의 IPv4 [서브넷 마스크] 설정 값을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip4\_subnet\_mask*

IPv4 [서브넷 마스크]를 변경할 서브넷 마스크가 저장되어있는 포인터입니다.

### Return values

*set\_ip4\_subnet\_mask*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

제품(ezTCP)이 사용할 수 없는 서브넷 마스크인 경우에는 -2를 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
```



```

int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

char *new_subnet_mask = "255.255.255.0";

if(ret == EZTCP_SUCCESS)
{
    int res = set_ip4_subnet_mask(out_eztcp_info.mac_address, new_subnet_mask);
    if(res == 1)
        printf("Success.\r\n");
    else if(res == -1)
        printf("There is no product information of the requested MAC address\r\n");
    else if(res == -2)
        printf("You entered an incorrect subnet mask.\r\n");
}

```

### See also

set\_ip4\_local\_address, set\_ip4\_subnet\_mask, set\_ip4\_gateway\_address, set\_ip4\_dns\_address.

### Remarks

*set\_ip4\_subnet\_mask* 입력 값인 *ip4\_subnet\_mask*에는 IPv4의 점으로 구분된 십진수 표기법으로 된 서브넷마스크를 입력해야 합니다.

## 5.1.7 set\_ip4\_gateway\_address

```
int set_ip4_gateway_address(unsigned char *mac_address, char *ip4_gateway_addr);
```

### Description

*set\_ip4\_gateway\_address*는 제품(ezTCP)의 IPv4 [게이트웨이 IP 주소] 설정 값을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip4\_gateway\_addr*

IPv4 [게이트웨이 IP 주소]를 변경할 게이트웨이 IPv4 주소가 저장되어있는 포인터입니다.

### Return values

*set\_ip4\_gateway\_address*는 설정 값 변경에 성공하면 1을 사용합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

char *new_gateway = "10.1.0.254";

if(ret == EZTCP_SUCCESS)
{
    int res = set_ip4_gateway_address(out_eztcp_info.mac_address, new_gateway);
    if(res == 1)
        printf("Success.\r\n");
    else if(res == -1)
        printf("There is no product information of the requested MAC address\r\n");
}
```

## See also

set\_ip4\_local\_address, set\_ip4\_subnet\_mask, set\_ip4\_gateway\_address, set\_ip4\_dns\_address.

## Remarks

*set\_ip4\_gateway\_address* 입력 값인 *ip4\_gateway\_addr*에는 IPv4의 점으로 구분된 십진수 표기법으로 된 게이트웨이 IPv4 주소를 입력해야 합니다. 사용할 수 없는 게이트웨이 IPv4 주소가 입력되면 **[게이트웨이 IP 주소]**는 0으로 변경됩니다.

### 5.1.8 set\_ip4\_dns\_address

```
int set_ip4_dns_address(unsigned char *mac_address, char *ip4_dns_addr);
```

## Description

*set\_ip4\_dns\_address*는 제품(ezTCP)의 IPv4 **[DNS 서버 IP 주소]** 설정 값을 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip4\_dns\_addr*

IPv4 **[DNS 서버 IP 주소]**를 변경할 DNS 서버 IPv4 주소가 저장되어있는 포인터입니다.

## Return values

*set\_ip4\_dns\_address*는 설정 값 변경에 성공하면 1을 사용합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

char *new_dns = "10.1.0.200";

if(ret == EZTCP_SUCCESS)
{
    int res = set_ip4_dns_address(out_eztcp_info.mac_address, new_dns);
    if(res == 1)
        printf("Success.\r\n");
    else if(res == -1)
        printf("There is no product information of the requested MAC address\r\n");
}
```

## See also

set\_ip4\_local\_address, set\_ip4\_subnet\_mask, set\_ip4\_gateway\_address, set\_ip4\_dns\_address.

## Remarks

*set\_ip4\_dns\_address* 입력 값인 *ip4\_dns\_addr*에는 IPv4의 점으로 구분된 십진수 표기법으로 된 DNS 서버 IPv4 주소를 입력해야 합니다. 사용할 수 없는 DNS 서버 IPv4 주소가 입력되면 **[DNS 서버 IP 주소]**는 0으로 변경됩니다.

### 5.1.9 set\_pppoe\_id

```
int set_pppoe_id(unsigned char *mac_address, char *pppoe_id);
```

## Description

*set\_pppoe\_id*는 제품(ezTCP)의 **[PPPoE 아이디]** 설정 값을 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*pppoe\_id*

**[PPPoE 아이디]**를 변경할 PPPoE 아이디가 저장되어있는 포인터입니다.

## Return values

*set\_pppoe\_id*는 설정 값 변경에 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

set\_pppoe, set\_pppoe\_pwd

### Remarks

[PPPoE 아이디]는 NULL을 제외하고 최대 32바이트입니다.

#### 5.1.10 set\_pppoe\_pwd

```
int set_pppoe_pwd(unsigned char *mac_address, char *pppoe_pwd);
```

### Description

*set\_pppoe\_pwd*는 제품(ezTCP)의 [PPPoE 비밀번호] 설정 값을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*pppoe\_pwd*

[PPPoE 비밀번호]를 변경할 PPPoE 비밀번호가 저장되어있는 포인터입니다.

### Return values

*set\_pppoe\_pwd*는 설정 값 변경에 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

### See also

get\_pppoe, get\_pppoe\_id

### Remarks

[PPPoE 비밀번호]는 NULL을 제외하고 최대 16바이트입니다.

#### 5.1.11 set\_ip6

```
int set_ip6(unsigned char *mac_address, int onoff);
```

### Description





**set\_ip6**는 [IPv6] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_ip6*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*onoff*에 1을 입력하는 경우 [SSL 보안통신] 기능이 선택되어있으면 -3을 반환합니다.

## Examples

## See also

*is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.

## Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

### 5.1.12 set\_ip6\_gua\_type

```
int set_ip6_gua_type(unsigned char *mac_address, int gua_type);
```

## Description

*set\_ip6\_eui\_type*는 IPv6 [제품 IP 주소] 설정 방식을 변경합니다.



## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int gua\_type*

IPv6 [제품 IP 주소] 설정 방식을 나타내며 아래와 같은 값을 가질 수 있습니다.

0

[자동으로 IP 주소 받기]

1

[고정된 IP 주소 사용]

## Return values

*get\_ip6\_gua\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*gua\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

### See also

*is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.

### Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

#### 5.1.13 set\_ip6\_eui\_type

```
int set_ip6_eui_type(unsigned char *mac_address, int eui_type);
```

### Description

*set\_ip6\_eui\_type*은 Interface ID 생성 방식을 변경합니다. Interface ID 생성 방식은 IPv6 [제품 IP 주소] 설정 방식이 [자동으로 IP 주소 받기]인 경우에 사용됩니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int eui\_type*

Interface ID 생성 방식을 나타내며 아래와 같은 값을 가질 수 있습니다.

*0*

[MAC 주소]

*1*

[Random]

**Return values***get\_ip6\_eui\_type*의 반환 값은 다음과 같습니다.*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1**mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.*-2**is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.*-3**eui\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.**Examples****See also***is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.**Remarks**

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

**5.1.14 set\_ip6\_local\_address**

```
int set_local_ip6(unsigned char *mac_address, char *ip6_local_address);
```

**Description***set\_ip6\_local\_address*는 제품(ezTCP)의 IPv6 [제품 IP 주소]를 변경합니다.**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip6\_local\_address*

IPv6 [제품 IP 주소]를 변경할 IPv6 주소가 저장되어있는 포인터입니다.

## Return values

*set\_ip6\_local\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

잘못된 IPv6 주소형식인 경우에는 -3을 반환합니다.

## Examples

## See also

*is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.

## Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

### 5.1.15 set\_ip6\_subnet\_prefix\_length

```
int set_ip6_subnet_prefix_length(unsigned char *mac_address,
                                int ip6_subnet_prefix_length);
```

## Description

*set\_ip6\_subnet\_prefix\_length*는 제품(ezTCP)의 IPv6 서브넷 접두사 길이를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int ip6\_subnet\_prefix\_length*

변경할 IPv6 서브넷 접두사 길이입니다. 사용 가능한 범위는 0 ~ 128입니다.

### Return values

*set\_ip6\_subnet\_prefix\_length*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

입력 가능한 IPv6 서브넷 접두사 길이가 아닌 경우에는 -3을 반환합니다.

### Examples

### See also

*is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.

### Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

#### 5.1.16 set\_ip6\_gateway\_address

```
int set_ip6_gateway_address(unsigned char *mac_address, char *ip6_gateway_address);
```

### Description

*set\_ip6\_gateway\_address*는 제품(ezTCP)의 IPv6 [게이트웨이 IP 주소]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip6\_gateway\_address*

IPv6 [게이트웨이 IP 주소]를 변경할 IPv6 주소가 저장되어있는 포인터입니다.

### Return values

*set\_ip6\_gateway\_address*의 반환 값은 다음과 같습니다.

1



설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

잘못된 IPv6 주소형식인 경우에는 -3를 반환합니다.

## Examples

## See also

*is\_ip6*, *set\_ip6\_gua\_type*, *set\_ip6\_eui\_type*, *set\_ip6\_local\_address*, *set\_ip6\_subnet\_prefix\_length*, *set\_ip6\_gateway\_address*, *set\_ip6\_dns\_address*.

## Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

### 5.1.17 *set\_ip6\_dns\_address*

```
int set_ip6_dns_address(unsigned char *mac_address, char *ip6_dns_address);
```

## Description

*set\_ip6\_dns\_address*는 제품(ezTCP)의 IPv6 [DNS 서버 IP 주소]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip6\_dns\_address*

IPv6 [DNS 서버 IP 주소]를 변경할 IPv6 주소가 저장되어있는 포인터입니다.

## Return values

*set\_ip6\_dns\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ip6*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.



-3

잘못된 IPv6 주소형식인 경우에는 -3를 반환합니다.

## Examples

### See also

is\_ip6, set\_ip6\_gua\_type, set\_ip6\_eui\_type, set\_ip6\_local\_address, set\_ip6\_subnet\_prefix\_length, set\_ip6\_gateway\_address, set\_ip6\_dns\_address.

### Remarks

IPv6에 대한 보다 자세한 내용은 당사 홈페이지에서 “IPv6 가이드” 문서를 참조하시기 바랍니다.

## 5.2 시리얼 포트

### 5.2.1 set\_uart\_serial\_type

```
int set_uart_serial_type(unsigned char *mac_address, int uart_index, int serial_type);
```

### Description

*set\_uart\_serial\_type*은 *uart\_index*로 지정된 시리얼 포트의 [시리얼 종류]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int serial\_type*

[시리얼 종류]를 나타내며 아래와 같은 값을 가질 수 있습니다.

0	RS-232
1	RS-485
2	RS-422

### Return values

*set\_uart\_serial\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

제품(ezTCP)이 CSE-M53 또는 CSE-M53N인 경우 [시리얼 종류]를 RS-485 또는 RS-422로 변경할 때 제품(ezTCP)에 설정된 [시리얼 통신속도]가 230,400bps보다 큰 경우에는 -2를 반환합니다.

## Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *set\_uart\_parity*, *set\_uart\_databit*, *set\_uart\_stopbit*, *set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

### Remarks

*is\_uart\_rs232*가 1을 반환하는 경우에만 RS-232를 사용할 수 있습니다.

*is\_uart\_rs485*가 1을 반환하는 경우에만 RS-485를 사용할 수 있습니다.

*is\_uart\_rs422*이 1을 반환하는 경우에만 RS-422를 사용할 수 있습니다.

## 5.2.2 set\_uart\_ttl

```
int set_uart_ttl(unsigned char *mac_address, int uart_index, int onoff);
```

### Description

*set\_uart\_ttl*은 *uart\_index*로 지정된 시리얼 포트의 [TTL] 출력 기능의 선택 여부를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.





**Return values**

*set\_uart\_ttl*의 반환 값은 다음과 같습니다.

*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*-2*

*is\_uart\_ttl*이 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

**Examples****See also**

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *set\_uart\_parity*, *set\_uart\_databit*, *set\_uart\_stopbit*, *set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

**Remarks**

*is\_uart\_ttl*이 1을 반환하는 경우에만 사용할 수 있습니다.

**5.2.3 set\_uart\_baudrate**

```
int set_uart_baudrate(unsigned char *mac_address, int uart_index, int baudrate);
```

**Description**

*set\_uart\_baudrate*는 *uart\_index*로 지정된 시리얼 포트의 [시리얼 통신속도]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int baudrate*

변경할 시리얼 통신속도 입니다.

**Return values**

*set\_uart\_baudrate*의 반환 값은 다음과 같습니다.

*1*

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

[시리얼 통신속도]를 변경할 수 없는 경우 -2를 반환합니다.

-3

*baudrate*에 입력된 값의 범위가 *get\_uart\_min\_baudrate*과 *get\_uart\_max\_baudrate*의 반환값을 벗어나는 경우 -3을 반환합니다.

## Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *set\_uart\_parity*, *set\_uart\_databit*, *set\_uart\_stopbit*, *set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

### Remarks

*uart\_index*로 지정된 시리얼 포트의 [시리얼 통신속도]는 다른 설정 값에 따라 자동으로 변경될 수 있습니다.

## 5.2.4 set\_uart\_parity

```
int set_uart_parity(unsigned char *mac_address, int uart_index, int parity);
```

### Description

*set\_uart\_parity*는 *uart\_index*로 지정된 시리얼 포트의 [패리티]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int parity*

[패리티] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

NONE

1

EVEN

2

ODD

3  
MARK

4  
SPACE

### Return values

*set\_uart\_parity*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_7\_and\_8\_databit\_only*가 1을 반환하는 제품(ezTCP)인 경우 [패리티]를 NONE으로 변경할 때 제품(ezTCP)에 설정된 [데이터 비트]가 7인 경우에는 -2를 반환합니다.

### Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*,  
*get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*,  
*is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *set\_uart\_stopbit*, *set\_uart\_flow\_control*,  
*set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

### Remarks

*is\_uart\_7\_and\_8\_databit\_only*가 1을 반환하는 제품(ezTCP)은 [데이터 비트]를 7비트로 사용하기 위해선 [패리티]를 NONE 이외의 값으로 설정해야 합니다.

## 5.2.5 set\_uart\_databit

```
int set_uart_databit(unsigned char *mac_address, int uart_index, int databit);
```

### Description

*get\_uart\_databit*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 비트]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int databit*

[데이터 비트] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0	5비트
1	6비트
2	7비트
3	8비트

### Return values

*set\_uart\_databit*의 반환 값은 다음과 같습니다.

- 1  
설정 값 변경에 성공하면 1을 반환합니다.
- 1  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.
- 2  
*is\_uart\_7\_and\_8\_databit\_only*가 1을 반환하는 제품(ezTCP)인 경우 [데이터 비트]를 7비트로 변경할 때 제품(ezTCP)에 설정된 [패리티]가 NONE인 경우에는 -2를 반환합니다.

### Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*, *get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*, *is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *set\_uart\_stopbit*, *set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

### Remarks

*is\_uart\_7\_and\_8\_databit\_only*가 1을 반환하는 제품(ezTCP)은 [데이터 비트]를 7비트로 사용하기 위해선 [패리티]를 NONE 이외의 값으로 설정해야 합니다.

## 5.2.6 set\_uart\_stopbit

```
int set_uart_stopbit(unsigned char *mac_address, int uart_index, int stopbit);
```

### Description

*set\_uart\_stopbit*는 *uart\_index*로 지정된 시리얼 포트의 [정지 비트]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int stopbit*

[정지 비트] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

1비트

1

1.5비트

2

2비트

## Return values

*set\_uart\_stopbit*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_one5\_stopbit*가 1을 반환하지 않는 제품(ezTCP)인 경우 [정지 비트]를 1.5비트로 변경하려는 경우에는 -2를 반환합니다.

## Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*,  
*get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*,  
*is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *is\_uart\_one5\_stopbit*, *set\_uart\_stopbit*,  
*set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

## Remarks

### 5.2.7 set\_uart\_flow\_control

```
int set_uart_flow_control(unsigned char *mac_address, int uart_index, int flow_control);
```

## Description



*set\_uart\_flow\_control*는 *uart\_index*로 지정된 시리얼 포트의 [흐름 제어]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int flow\_control*

[흐름 제어] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0	NONE
1	RTS/CTS
2	XON/XOFF

## Return values

*set\_uart\_flow\_control*의 반환 값은 다음과 같습니다.

1	설정 값 변경에 성공하면 1을 반환합니다.
-1	<i>mac_address</i> 을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.
-2	제품(ezTCP)이 CSE-M53 또는 CSE-H53인 경우 [흐름 제어]를 XON/XOFF로 변경할 때 제품(ezTCP)에 설정된 [시리얼 통신속도]가 230,400bps보다 큰 경우에는 -2를 반환합니다.
-3	<i>is_uart_xonxoff</i> 가 1을 반환하지 않는 제품(ezTCP)인 경우 [흐름 제어]를 XON/XOFF로 변경하려는 경우에는 -3을 반환합니다.

## Examples

### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*,  
*get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*,  
*is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *is\_uart\_one5\_stopbit*, *set\_uart\_stopbit*,  
*is\_uart\_xonxoff*, *set\_uart\_flow\_control*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

## Remarks

### 5.2.8 set\_uart\_dtrdsr

```
int set_uart_dtrdsr(unsigned char *mac_address, int uart_index, int onoff);
```

#### Description

*set\_uart\_dtrdsr*은 *uart\_index*로 지정된 시리얼 포트의 [DTR/DSR] 설정 값을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

#### Return values

*set\_uart\_dtrdsr*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_dtrdsr*이 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*,  
*get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*,  
*is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *is\_uart\_one5\_stopbit*, *set\_uart\_stopbit*,  
*set\_uart\_flow\_control*, *is\_uart\_dtrdsr*, *set\_uart\_dtrdsr*, *set\_uart\_tx\_delay*.

#### Remarks

*is\_uart\_dtrdsr*이 1을 반환하는 경우에만 사용할 수 있습니다.

### 5.2.9 set\_uart\_tx\_delay

```
int set_uart_tx_delay(unsigned char *mac_address, int uart_index, int tx_delay);
```

#### Description

*set\_uart\_tx\_delay*는 *uart\_index*로 지정된 시리얼 포트의 [데이터 전송 간격]을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int tx\_delay*

변경할 데이터 전송 간격입니다.

#### Return values

*set\_uart\_tx\_delay*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_tx\_delay*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*tx\_delay*에 입력된 값의 범위가 0 ~ 25가 아니면 -3을 반환합니다.

#### Examples

#### See also

*is\_uart\_rs232*, *is\_uart\_rs485*, *is\_uart\_rs422*, *set\_uart\_serial\_type*, *is\_uart\_ttl*, *set\_uart\_ttl*,  
*get\_uart\_min\_baudrate*, *get\_uart\_max\_baudrate*, *set\_uart\_baudrate*, *is\_uart\_8\_databit\_only*,  
*is\_uart\_7\_and\_8\_databit\_only*, *set\_uart\_parity*, *set\_uart\_databit*, *is\_uart\_one5\_stopbit*, *set\_uart\_stopbit*,  
*set\_uart\_flow\_control*, *is\_uart\_dtrdsr*, *set\_uart\_dtrdsr*, *is\_uart\_tx\_delay*, *set\_uart\_tx\_delay*.

#### Remarks

*is\_uart\_tx\_delay*가 1을 반환하는 경우에만 사용할 수 있습니다.



## 5.3 TCP/IP

### 5.3.1 set\_uart\_communication\_mode

```
int set_uart_communication_mode(unsigned char *mac_address, int uart_index,
                                int communication_mode);
```

#### Description

*set\_uart\_communication\_mode*는 *uart\_index*로 지정된 시리얼 포트의 **[통신모드]**를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int communication\_mode*

**[통신모드]** 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

- |   |                 |
|---|-----------------|
| 0 | T2S – TCP 서버    |
| 1 | ATC – AT 명령     |
| 2 | COD – TCP 클라이언트 |
| 3 | U2S – UDP       |
| 4 | 시리얼 Modbus/TCP  |

#### Return values

*set\_uart\_communication\_mode*의 반환 값은 다음과 같습니다.

- |    |  |
|----|--|
| 1  | 설정 값 변경에 성공하면 1을 반환합니다.  |
| -1 | <i>mac_address</i> 을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.                                   |
| -2 | <b>[통신모드]</b> 를 T2S – TCP 서버가 아닌 값으로 변경할 때 제품(ezTCP)에 <b>[다중 접속]</b> 기능이 설정된 경우에는 -2를 반환합니다. |

-3

[통신모드]를 U2S - UDP로 변경할 때 제품(ezTCP)에 [SSL 보안통신] 기능이 설정된 경우에는 -3을 반환합니다.

-4

[통신모드]를 T2S - TCP 서버 이외의 값으로 변경할 때 제품(ezTCP)에 [SSH 보안통신] 기능이 설정된 경우에는 -4를 반환합니다.

## Examples

### See also

is\_uart\_tcp\_server, is\_uart\_at\_command, is\_uart\_tcp\_client, is\_uart\_udp, is\_uart\_serial\_modbus, set\_uart\_communication\_mode, set\_uart\_peer\_address, set\_uart\_peer\_port, set\_uart\_local\_port, set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval, set\_uart\_separator\_length, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay, set\_uart\_rfc2217, set\_uart\_protocol.

### Remarks

*is\_uart\_tcp\_server*가 1을 반환하는 경우에만 T2S - TCP 서버를 사용할 수 있습니다.

*is\_uart\_at\_command*가 1을 반환하는 경우에만 ATC - AT 명령을 사용할 수 있습니다.

*is\_uart\_tcp\_client*가 1을 반환하는 경우에만 T2S - TCP 클라이언트를 사용할 수 있습니다.

*is\_uart\_udp*가 1을 반환하는 경우에만 U2S - UDP를 사용할 수 있습니다.

*is\_uart\_serial\_modbus*가 1을 반환하는 경우에만 시리얼 Modbus/TCP를 사용할 수 있습니다.

## 5.3.2 set\_uart\_peer\_address

```
int set_uart_peer_address(unsigned char *mac_address, int uart_index,
                          char *peer_address);
```

### Description

*set\_uart\_peer\_address*는 *uart\_index*로 지정된 시리얼 포트의 [통신할 주소]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*char \*peer\_address*

통신할 주소가 저장되어있는 포인터입니다.

IPv6 기능이 선택되어있으면 *peer\_address*에 IPv6 주소형식을 입력해야 하고 그렇지 않으면 IPv4의 점으로 구분된 십진수 표기법으로 된 IP 주소 또는 호스트 이름을 입력해야 합니다.

## Return values

*set\_uart\_peer\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

IPv6 기능이 선택되어있는 경우 *peer\_address*에 잘못된 IPv6 주소형식을 입력하면 -2를 반환합니다.

## Examples

```
char *ipv6_peer_address = "2001:DB8::9";
char *ipv4_peer_address = "192.168.0.5";

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int res;
    unsigned char *mac_address = out_eztcp_info.mac_address;

    if(get_ip6(mac_address) == 1)
        res = set_uart_peer_address(mac_address, 0, ipv6_peer_address);
    else
        res = set_uart_peer_address(mac_address, 0, ipv4_peer_address);

    if(res == 1)
        printf("Success\r\n");
    else if(res == -1)
        printf("There is no product information of the requested MAC address\r\n");
    else if(res == -2)
        printf("You entered an incorrect IPV6 address.\r\n");
}
```

## See also

is\_uart\_tcp\_server, is\_uart\_at\_command, is\_uart\_tcp\_client, is\_uart\_udp, is\_uart\_serial\_modbus, set\_uart\_communication\_mode, set\_uart\_peer\_address, set\_uart\_peer\_port, set\_uart\_local\_port, set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval, set\_uart\_separator\_length, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay, set\_uart\_rfc2217, set\_uart\_protocol.

**Remarks**

[통신할 주소]는 NULL을 제외하고 최대 64바이트입니다.

**5.3.3 set\_uart\_peer\_port**

```
int set_uart_peer_port(unsigned char *mac_address, int uart_index, int peer_port);
```

**Description**

*set\_uart\_peer\_port*는 *uart\_index*로 지정된 시리얼 포트의 [통신할 포트]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int peer\_port*

변경할 [통신할 포트]입니다.

**Return values**

*set\_uart\_peer\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*,  
*set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*,  
*set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*,  
*set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*,  
*set\_uart\_rfc2217*, *set\_uart\_protocol*.

**Remarks****5.3.4 set\_uart\_local\_port**

```
int set_uart_local_port(unsigned char *mac_address, int uart_index, int local_port);
```

## Description

*set\_uart\_local\_port*는 *uart\_index*로 지정된 시리얼 포트의 **[제품 로컬포트]**를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int local\_port*

변경할 **[제품 로컬포트]**입니다.

## Return values

*set\_uart\_local\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*local\_port*에 입력한 제품 로컬포트 번호가 이미 사용 중인 경우 -2를 반환합니다.

-3

**[텔넷]** 기능이 선택되어있을 때 *local\_port*에 23번을 입력하는 경우 -3을 반환합니다.

-4

I/O 제품의 Modbus/TCP 접속 방법이 **[수동 접속]**이고 *local\_port*에 입력한 제품 로컬포트 번호가 **Modbus/TCP [제품 로컬포트]**로 이미 사용 중인 경우 -4를 반환합니다.

-5

I/O 제품 **[웹(HTTP)]** 제어방식 기능이 선택되어있을 때 *local\_port*에 입력한 제품 로컬포트 번호가 I/O 제품 **[웹(HTTP) 포트]** 번호로 이미 사용 중인 경우 -5를 반환합니다.

-6

*local\_port*에 입력한 제품 로컬포트 번호가 **50,005**번이면 -6을 반환합니다.

-7

*local\_port*에 입력한 제품 로컬포트 번호가 **50,006**번이면 -7을 반환합니다.

-8

*local\_port*에 입력한 제품 로컬포트 번호가 **50,007**번이면 -8을 반환합니다.

## Examples

## See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*,  
*set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*,

set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval,  
set\_uart\_separator\_length, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay,  
set\_uart\_rfc2217, set\_uart\_protocol.

## Remarks

50,005번, 50,006번 그리고 50,007번은 제품(ezTCP) 검색 등을 위해서 라이브러리 내부에서 사용하고 있는 포트번호 이므로 제품(ezTCP)의 설정 값으로 사용할 수 없습니다.

### 5.3.5 set\_uart\_cod\_tcp\_server

```
int set_uart_cod_tcp_server(unsigned char *mac_address, int uart_index, int onoff);
```

## Description

*set\_uart\_cod\_tcp\_server*은 *uart\_index*로 지정된 시리얼 포트의 [TCP 서버] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_uart\_cod\_tcp\_server*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

is\_uart\_tcp\_server, is\_uart\_at\_command, is\_uart\_tcp\_client, is\_uart\_udp, is\_uart\_serial\_modbus,

set\_uart\_communication\_mode, set\_uart\_peer\_address, set\_uart\_peer\_port, set\_uart\_local\_port,  
 set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval,  
 set\_uart\_separator\_length, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay,  
 set\_uart\_rfc2217, set\_uart\_protocol.

## Remarks

*get\_uart\_communication\_mode*이 2(COD - TCP 클라이언트)를 반환하는 경우에만 사용할 수 있습니다.

### 5.3.6 set\_uart\_watermark

```
int set_uart_watermark(unsigned char *mac_address, int uart_index, int watermark);
```

## Description

*set\_uart\_watermark*는 *uart\_index*로 지정된 시리얼 포트의 [접속전 데이터 크기] 또는 [패킷 블록 설정(바이트)]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int watermark*

변경할 접속전 데이터 크기 또는 패킷 블록 설정(바이트) 값입니다.

## Return values

*set\_uart\_watermark*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

is\_uart\_tcp\_server, is\_uart\_at\_command, is\_uart\_tcp\_client, is\_uart\_udp, is\_uart\_serial\_modbus,  
 set\_uart\_communication\_mode, set\_uart\_peer\_address, set\_uart\_peer\_port, set\_uart\_local\_port,  
 set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval,  
 set\_uart\_separator\_length, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay,  
 set\_uart\_rfc2217, set\_uart\_protocol.

**Remarks**

[통신모드]가 0 (T2S – TCP 서버), 1 (ATC – AT 명령), 2 (COD – TCP 클라이언트)인 경우 [접속전 데이터 크기]로 사용됩니다.

[통신모드]가 3 (U2S – UDP)인 경우 [패킷 블록 설정 (바이트)]로 사용됩니다.

[접속전 데이터 크기] 또는 [패킷 블록 설정(바이트)]는 다른 설정 값에 따라 자동으로 변경될 수 있습니다.

**5.3.7 set\_uart\_timeout**

```
int set_uart_timeout(unsigned char *mac_address, int uart_index, int timeout);
```

**Description**

*set\_uart\_timeout*은 *uart\_index*로 지정된 시리얼 포트의 [접속종료 대기시간(초)]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int timeout*

변경할 접속종료 대기시간(초)입니다.

**Return values**

*set\_uart\_timeout*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*, *set\_uart\_rfc2217*, *set\_uart\_protocol*.

**Remarks**



[통신모드]가 0 (T2S – TCP 서버), 1 (ATC – AT 명령), 2 (COD – TCP 클라이언트)인 경우에만 사용할 수 있습니다.

[접속종료 대기시간(초)]는 다른 설정 값에 따라 자동으로 변경될 수 있습니다.

### 5.3.8 set\_uart\_data\_frame\_interval

```
int set_uart_data_frame_interval(unsigned char *mac_address, int uart_index,
                                int data_frame_interval);
```

#### Description

*set\_uart\_data\_frame\_interval*은 *uart\_index*로 지정된 시리얼 포트의 [데이터 프레임 간격 (10ms)]를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int data\_frame\_interval*

변경할 데이터 프레임 간격(10ms)입니다.

#### Return values

*set\_uart\_data\_frame\_interval*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_data\_frame\_interval*이 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *is\_uart\_data\_frame\_interval*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*, *set\_uart\_rfc2217*, *set\_uart\_protocol*.

#### Remarks



[통신모드]가 4 (시리얼 Modbus/TCP)인 경우에는 사용할 수 없습니다.

### 5.3.9 set\_uart\_separator\_length

```
int set_uart_separator_length(unsigned char *mac_address, int uart_index,
                             int separator_length);
```

#### Description

*set\_uart\_separator\_length*는 *uart\_index*로 지정된 시리얼 포트의 [구분자 길이]를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int separator\_length*

변경할 구분자 길이로 0 ~ 4입니다.

#### Return values

*set\_uart\_separator\_length*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_separator*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *is\_uart\_separator*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*, *set\_uart\_rfc2217*, *set\_uart\_protocol*.

#### Remarks

### 5.3.10 set\_uart\_separator\_type

```
int set_uart_separator_type(unsigned char *mac_address, int uart_index,
                           int separator_type);
```

#### Description

*set\_uart\_separator\_type*은 *uart\_index*로 지정된 시리얼 포트의 [구분자 동작방식]을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int separator\_type*

[구분자 동작방식] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

구분자까지 전송

1

구분자+1바이트 전송

2

구분자+2바이트 전송

#### Return values

*set\_uart\_separator\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_separator*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*separator\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

#### Examples

#### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*,  
*set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*,

set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval,  
 set\_uart\_separator\_length, is\_uart\_separator, set\_uart\_separator\_type, set\_uart\_separator,  
 set\_uart\_tcp\_nodelay, set\_uart\_rfc2217, set\_uart\_protocol.

## Remarks

### 5.3.11 set\_uart\_separator

```
int set_uart_separator(unsigned char *mac_address, int uart_index, int separator_index,
                      char *separator);
```

## Description

*set\_uart\_separator*는 *uart\_index*로 지정된 시리얼 포트의 [구분자(16진수)]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int separator\_index*

구분자의 순번으로 0부터 시작합니다.

*char \*separator*

*separator\_index*로 지정된 구분자를 변경할 16진수 문자열입니다.

## Return values

*set\_uart\_separator*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_separator*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*separator\_index*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

```
char *separator = "3A";
struct eztcp_info out_eztcp_info;
```

```

int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    int res;
    unsigned char *mac_address = out_eztcp_info.mac_address;

    if(is_uart_separaot(mac_address) == 1)
    {
        res = set_uart_separator(mac_address, 0, 0, separator);

        if(res == 1)
            printf("Success\r\n");
        else if(res == -1)
            printf("There is no product information of the requested MAC
address\r\n");
        else if(res == -2)
            printf("This product can't use separator function.\r\n");
        else if(res == -3)
            printf("You entered an invalid separator.\r\n");
    }
}
}

```

### See also

is\_uart\_tcp\_server, is\_uart\_at\_command, is\_uart\_tcp\_client, is\_uart\_udp, is\_uart\_serial\_modbus, set\_uart\_communication\_mode, set\_uart\_peer\_address, set\_uart\_peer\_port, set\_uart\_local\_port, set\_uart\_cod\_tcp\_server, set\_uart\_watermark, set\_uart\_timeout, set\_uart\_data\_frame\_interval, set\_uart\_separator\_length, is\_uart\_separator, set\_uart\_separator\_type, set\_uart\_separator, set\_uart\_tcp\_nodelay, set\_uart\_rfc2217, set\_uart\_protocol.

### Remarks

*separator*는 16진수 문자열로 최대 2자입니다.

### 5.3.12 set\_uart\_tcp\_nodelay

```
int set_uart_tcp_nodelay(unsigned char *mac_address, int uart_index, int onoff);
```

### Description

*set\_uart\_tcp\_nodelay*는 *uart\_index*로 지정된 시리얼 포트의 [전송지연 기능 사용안함] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_uart\_tcp\_nodelay*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_tcp\_nodelay*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *is\_uart\_tcp\_nodelay*, *set\_uart\_tcp\_nodelay*, *set\_uart\_rfc2217*, *set\_uart\_protocol*.

## Remarks

### 5.3.13 set\_uart\_rfc2217

```
int set_uart_rfc2217(unsigned char *mac_address, int uart_index, int onoff);
```

## Description

*set\_uart\_rfc2217*은 *uart\_index*로 지정된 시리얼 포트의 [시리얼 포트 설정/상태 전송 (RFC2217)] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_uart\_rfc2217*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_rfc2217*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

[다중 접속]이 선택되어 있을 때 *onoff*에 1을 입력하면 -3을 반환합니다.

-4

[SSL 보안통신] 또는 [SSH 보안통신]이 선택되어 있을 때 *onoff*에 1을 입력하면 -4를 반환합니다.

## Examples

## See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*, *is\_uart\_rfc2217*, *set\_uart\_rfc2217*, *set\_uart\_protocol*.

## Remarks

### 5.3.14 set\_uart\_protocol

```
int set_uart_protocol(unsigned char *mac_address, int uart_index, int uart_protocol);
```

#### Description

*set\_uart\_protocol*은 *uart\_index*로 지정된 시리얼 포트가 사용 중인 [프로토콜]을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

*int uart\_protocol*

[프로토콜] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

TCP

1

TCP + TELNET

2

TCP + SSL

#### Return values

*set\_uart\_protocol*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_uart\_protocol*이 1을 반환하지 않는 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_uart\_tcp\_server*, *is\_uart\_at\_command*, *is\_uart\_tcp\_client*, *is\_uart\_udp*, *is\_uart\_serial\_modbus*, *set\_uart\_communication\_mode*, *set\_uart\_peer\_address*, *set\_uart\_peer\_port*, *set\_uart\_local\_port*, *set\_uart\_cod\_tcp\_server*, *set\_uart\_watermark*, *set\_uart\_timeout*, *set\_uart\_data\_frame\_interval*, *set\_uart\_separator\_length*, *set\_uart\_separator\_type*, *set\_uart\_separator*, *set\_uart\_tcp\_nodelay*, *set\_uart\_rfc2217*, *is\_uart\_protocol*, *set\_uart\_protocol*.



**Remarks**

현재 CSE-T16, CSE-T32, CSE-T48만 사용 가능하며 추후 지원가능 모델은 추가 될 수 있습니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.4 CSC-HR2****5.4.1 set\_csc\_hr2\_communication\_mode**

```
int set_csc_hr2_communication_mode(unsigned char *mac_address,
                                   int communication_mode);
```

**Description**

*set\_csc\_hr2\_communication\_mode*는 CSC-HR2의 [통신모드]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int communication\_mode*

[통신모드] 종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

자동

1

EZU-100 펌웨어 변경

**Return values**

*set\_csc\_hr2\_communication\_mode*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

**Examples****See also**

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,

```
set_csc_hr2_checkpoint,      set_csc_hr2_first_server_address,      set_csc_hr2_first_server_port,
set_csc_hr2_second_server_address, set_csc_hr2_second_server_port.
```

## Remarks

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

### 5.4.2 set\_csc\_hr2\_id

```
int set_csc_hr2_id(unsigned char *mac_address, char *csc_hr2_id);
```

## Description

*set\_csc\_hr2\_id*는 CSC-HR2의 [CSC-HR2 아이디]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*csc\_hr2\_id*

변경할 CSC-HR2 아이디입니다.

## Return values

*set\_csc\_hr2\_id*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

```
is_csc_hr2,      set_csc_hr2_communication_mode,      set_csc_hr2_id,      set_csc_hr2_network_timeout,
set_csc_hr2_network_threshold,      set_csc_hr2_server_timeout,      set_csc_hr2_server_threshold,
set_csc_hr2_checkpoint,      set_csc_hr2_first_server_address,      set_csc_hr2_first_server_port,
set_csc_hr2_second_server_address, set_csc_hr2_second_server_port.
```

## Remarks

[CSC-HR2 아이디]는 NULL을 제외하고 최대 16바이트입니다. 보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

### 5.4.3 set\_csc\_hr2\_network\_timeout

```
int set_csc_hr2_network_timeout(unsigned char *mac_address, int timeout);
```

#### Description

*set\_csc\_hr2\_network\_timeout*은 CSC-HR2의 [절체 타임아웃]을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int timeout*

변경할 절체 타임아웃입니다.

#### Return values

*set\_csc\_hr2\_network\_timeout*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

#### Remarks

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

### 5.4.4 set\_csc\_hr2\_network\_threshold

```
int set_csc_hr2_network_threshold(unsigned char *mac_address, int threshold);
```

#### Description

*set\_csc\_hr2\_network\_threshold*는 CSC-HR2의 [절체 바이트 수]를 변경합니다.



**Parameters***unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
*int threshold*

변경할 절체 바이트 수입니다.

**Return values**

*set\_csc\_hr2\_network\_threshold*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

**Examples****See also**

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

**Remarks**

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

**5.4.5 set\_csc\_hr2\_server\_timeout**

```
int set_csc_hr2_server_timeout(unsigned char *mac_address, int timeout);
```

**Description**

*set\_csc\_hr2\_server\_timeout*은 CSC-HR2의 [서버변경 타임아웃]을 변경합니다.

**Parameters***unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
*int timeout*

변경할 서버변경 타임아웃입니다.

**Return values**

*set\_csc\_hr2\_server\_timeout*의 반환 값은 다음과 같습니다.

*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

*-2*

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

**Examples****See also**

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

**Remarks**

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

**5.4.6 set\_csc\_hr2\_server\_threshold**

```
int set_csc_hr2_server_threshold(unsigned char *mac_address, int threshold);
```

**Description**

*set\_csc\_hr2\_server\_threshold*는 CSC-HR2의 [서버변경 바이트 수]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int threshold*

변경할 서버변경 바이트 수입니다.

**Return values**

*set\_csc\_hr2\_server\_threshold*의 반환 값은 다음과 같습니다.

*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.



-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

### See also

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

### Remarks

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

## 5.4.7 set\_csc\_hr2\_checkport

```
int set_csc_hr2_checkport(unsigned char *mac_address, int check_port);
```

### Description

*set\_csc\_hr2\_check\_port*는 CSC-HR2의 [통신품질 점검포트]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int check\_port*

변경할 통신품질 점검포트 번호입니다.

### Return values

*set\_csc\_hr2\_checkport*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

### See also



is\_csc\_hr2, set\_csc\_hr2\_communication\_mode, set\_csc\_hr2\_id, set\_csc\_hr2\_network\_timeout,  
 set\_csc\_hr2\_network\_threshold, set\_csc\_hr2\_server\_timeout, set\_csc\_hr2\_server\_threshold,  
 set\_csc\_hr2\_checkpoint, set\_csc\_hr2\_first\_server\_address, set\_csc\_hr2\_first\_server\_port,  
 set\_csc\_hr2\_second\_server\_address, set\_csc\_hr2\_second\_server\_port.

## Remarks

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

### 5.4.8 set\_csc\_hr2\_first\_server\_address

```
int set_csc_hr2_first_server_address(unsigned char *mac_address, char *server_address);
```

## Description

*set\_csc\_hr2\_first\_server\_address*는 CSC-HR2의 [첫 번째 서버 주소]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*server\_address*

첫 번째 서버 주소가 저장되어있는 포인터입니다.

## Return values

*set\_csc\_hr2\_first\_server\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

is\_csc\_hr2, set\_csc\_hr2\_communication\_mode, set\_csc\_hr2\_id, set\_csc\_hr2\_network\_timeout,  
 set\_csc\_hr2\_network\_threshold, set\_csc\_hr2\_server\_timeout, set\_csc\_hr2\_server\_threshold,  
 set\_csc\_hr2\_checkpoint, set\_csc\_hr2\_first\_server\_address, set\_csc\_hr2\_first\_server\_port,  
 set\_csc\_hr2\_second\_server\_address, set\_csc\_hr2\_second\_server\_port.

## Remarks

[첫 번째 서버 주소]는 NULL을 제외하고 최대 64바이트입니다. 보다 자세한 내용은 CSC-

HR2 제품 사용설명서를 참조하시기 바랍니다.

### 5.4.9 set\_csc\_hr2\_first\_server\_port

```
int set_csc_hr2_first_server_port(unsigned char *mac_address, int server_port);
```

#### Description

*set\_csc\_hr2\_first\_server\_port*는 CSC-HR2가 통신할 첫 번째 서버의 [포트번호]를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int server\_port*

변경할 첫 번째 서버의 포트번호입니다.

#### Return values

*set\_csc\_hr2\_first\_server\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkpoint*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

#### Remarks

*is\_csc\_hr2*가 1을 반환하는 경우에만 사용할 수 있습니다. 보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.



#### 5.4.10 set\_csc\_hr2\_second\_server\_address

```
int set_csc_hr2_second_server_address(unsigned char *mac_address,
                                     char *server_address);
```

##### Description

*set\_csc\_hr2\_second\_server\_address*는 CSC-HR2의 [두 번째 서버 주소]를 변경합니다.

##### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*server\_address*

두 번째 서버 주소가 저장되어있는 포인터입니다.

##### Return values

*set\_csc\_hr2\_second\_server\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

##### Examples

##### See also

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

##### Remarks

[두 번째 서버 주소]는 NULL을 제외하고 최대 64바이트입니다. 보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

#### 5.4.11 set\_csc\_hr2\_second\_server\_port

```
int set_csc_hr2_second_server_port(unsigned char *mac_address, int server_port);
```

## Description

*set\_csc\_hr2\_second\_server\_port*는 CSC-HR2가 통신할 두 번째 서버의 [포트번호]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int server\_port*

변경할 두 번째 서버의 포트번호입니다.

## Return values

*set\_csc\_hr2\_second\_server\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_csc\_hr2*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

*is\_csc\_hr2*, *set\_csc\_hr2\_communication\_mode*, *set\_csc\_hr2\_id*, *set\_csc\_hr2\_network\_timeout*,  
*set\_csc\_hr2\_network\_threshold*, *set\_csc\_hr2\_server\_timeout*, *set\_csc\_hr2\_server\_threshold*,  
*set\_csc\_hr2\_checkport*, *set\_csc\_hr2\_first\_server\_address*, *set\_csc\_hr2\_first\_server\_port*,  
*set\_csc\_hr2\_second\_server\_address*, *set\_csc\_hr2\_second\_server\_port*.

## Remarks

보다 자세한 내용은 CSC-HR2 제품 사용설명서를 참조하시기 바랍니다.

## 5.5 I/O 제품

### 5.5.1 set\_io\_http

```
int set_io_http(unsigned char *mac_address, int onoff);
```

## Description

*set\_io\_http*는 I/O 제품 [웹(HTTP)] 제어방식 기능 선택 여부를 변경합니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

*0*

기능 선택을 해제합니다.

*1*

기능을 선택합니다.

**Return values***set\_io\_http*의 반환 값은 다음과 같습니다.*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1**mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.*-2**is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.**Examples****See also**

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

**Remarks****5.5.2 set\_io\_http\_port**

```
int set_io_http_port(unsigned char *mac_address, int http_port);
```

**Description***set\_io\_http\_port*는 I/O 제품 [웹(HTTP) 포트] 번호를 변경합니다.**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int http\_port*

변경할 웹(HTTP) 포트번호입니다.

## Return values

*set\_io\_http\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.3 set\_io\_html\_size

```
int set_io_html_size(unsigned char *mac_address, int html_size);
```

## Description

*set\_io\_html\_size*는 I/O 제품 [웹(HTTP)페이지 크기]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int html\_size*

[웹(HTTP)페이지 크기]를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

	80KB
1	
	96KB
2	
	112KB

## Return values

*set\_io\_html\_size*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 제품인 경우 -2를 반환합니다.

-3

*html\_size*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.4 set\_io\_modbus

```
int set_io_modbus(unsigned char *mac_address, int onoff);
```

## Description

*set\_io\_modbus*는 I/O 제품 [Modbus/TCP] 제어방식 기능 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.



*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_io\_modbus*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.5 *set\_io\_input\_notification*

```
int set_io_input_notification(unsigned char *mac_address, int onoff);
```

## Description

*set\_io\_input\_notification*은 I/O 제품의 [입력포트 변경 알림] 기능 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_io\_input\_notification*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.6 *set\_io\_output\_automatic\_initialize*

```
int set_io_output_automatic_initialize(unsigned char *mac_address, int onoff);
```

## Description

*set\_io\_output\_automatic\_initialize*는 I/O 제품의 [출력포트 상태 초기화(Modbus/TCP 접속 종료 시 출력포트[초기상태] 값으로 상태 변경)] 기능 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

### Return values

*set\_io\_output\_automatic\_initialize*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_io\_output\_automatic\_initialize*가 1을 반환하지 않는 경우 -3을 반환합니다.

### Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *is\_io\_output\_automatic\_initialize*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5.5.7 set\_io\_modbus\_type

```
int set_io_modbus_type(unsigned char *mac_address, int modbus_type);
```

### Description

*set\_io\_modbus\_type*은 I/O 제품의 [마스터/슬레이브]를 변경합니다.

### Parameters





*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int modbus\_type*

[마스터/슬레이브]종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

슬레이브

1

마스터

## Return values

*set\_io\_modbus\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*modbus\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.8 set\_io\_unit\_id

```
int set_io_unit_id(unsigned char *mac_address, int unit_id);
```

## Description

*set\_io\_unit\_id*는 I/O 제품의 [유니트 아이디]를 변경합니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int unit\_id*

변경할 유닛 아이디입니다.

**Return values***set\_io\_unit\_id*의 반환 값은 다음과 같습니다.*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1**mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.*-2**is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.**Examples****See also**

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

**Remarks****5.5.9 set\_io\_input\_address**

```
int set_io_input_address(unsigned char *mac_address, int input_address);
```

**Description***set\_io\_input\_address*는 I/O 제품의 [입력포트 주소]를 변경합니다.**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*int input\_address*

변경할 입력포트 주소입니다.

## Return values

*set\_io\_input\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*input\_address*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.10 set\_io\_output\_address

```
int set_io_output_address(unsigned char *mac_address, int output_address);
```

## Description

*set\_io\_output\_address*는 I/O 제품의 [출력포트 주소]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int input\_address*

변경할 출력포트 주소입니다.

## Return values



*set\_io\_output\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*output\_address*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.11 *set\_io\_poll\_interval*

```
int set_io_poll_interval(unsigned char *mac_address, int poll_interval);
```

## Description

*set\_io\_poll\_interval*은 I/O 제품의 마스터 [통신 주기]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int poll\_interval*

변경할 통신 주기입니다.

## Return values

*set\_io\_poll\_interval*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

#### 5.5.12 *set\_io\_slave\_control\_type*

```
int set_io_slave_control_type(unsigned char *mac_address, int slave_control_type);
```

### Description

*set\_io\_slave\_control\_type*은 I/O 제품의 [슬레이브 출력포트 제어방식]을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int slave\_control\_type*

[슬레이브 출력포트 제어방식]종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

FC 16 (동시제어)

1

FC 05 (개별제어)

### Return values

*set\_io\_slave\_control\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*slave\_control\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

### 5.5.13 *set\_io\_master\_control\_type*

```
int set_io_master_control_type(unsigned char *mac_address, int master_control_type);
```

### Description

*set\_io\_master\_control\_type*은 I/O 제품의 [마스터 출력포트 제어방식]을 반환합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int master\_control\_type*

[마스터 출력포트 제어방식]종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

AND

1

OR

### Return values

*set\_io\_master\_control\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*master\_control\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

### 5.5.14 set\_io\_tcp\_type

```
int set_io_tcp_type(unsigned char *mac_address, int tcp_type);
```

## Description

*set\_io\_tcp\_type*은 I/O 제품의 Modbus/TCP 접속 방법을 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int master\_control\_type*

[마스터 출력포트 제어방식]종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

수동접속

1

능동접속

## Return values



*set\_io\_tcp\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*tcp\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

보다 자세한 내용은 제품 사용설명서를 참조하시기 바랍니다.

### 5.5.15 *set\_io\_multi\_connection\_number*

```
int set_io_multi_connection_number(unsigned char *mac_address, int number);
```

## Description

*set\_io\_multi\_connection\_number*은 I/O 제품이 Modbus/TCP 수동접속인 경우 동시 접속 가능한 Modbus/TCP 클라이언트의 개수를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int number*

변경할 동시 접속 가능한 Modbus/TCP 클라이언트의 개수로 1 ~ 8입니다.

## Return values

*set\_io\_multi\_connection\_number*의 반환 값은 다음과 같습니다.





1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*number*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

#### 5.5.16 *set\_io\_modbus\_peer\_address*

```
int set_io_modbus_peer_address(unsigned char *mac_address, char *peer_address);
```

### Description

*set\_io\_modbus\_peer\_address*는 I/O 제품이 Modbus/TCP 능동접속인 경우 [통신할 주소]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*peer\_address*

통신할 주소가 저장되어있는 포인터입니다. 통신할 주소에는 IPv4의 점으로 구분된 십진 수 표기법으로 된 IP 주소 또는 호스트 이름을 입력해야 합니다.

### Return values

*set\_io\_modbus\_peer\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

[통신할 주소]는 NULL을 제외하고 최대 64바이트입니다.

## 5.5.17 set\_io\_modbus\_peer\_port

```
int set_io_modbus_peer_port(unsigned char *mac_address, int peer_port);
```

### Description

*set\_io\_modbus\_peer\_port*는 I/O 제품이 Modbus/TCP 능동접속인 경우 [통신할 포트]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int peer\_port*

변경할 통신할 포트입니다.

### Return values

*set\_io\_modbus\_peer\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.18 *set\_io\_modbus\_local\_port*

```
int set_io_modbus_local_port(unsigned char *mac_address, int local_port);
```

## Description

*set\_io\_modbus\_local\_port*는 I/O 제품이 Modbus/TCP 수동접속인 경우 Modbus/TCP [제품 로컬포트]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int peer\_port*

변경할 제품 로컬포트입니다.

## Return values

*set\_io\_modbus\_local\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples



**See also**

is\_io, set\_io\_http, set\_io\_http\_port, set\_io\_html\_size, set\_io\_modbus, set\_io\_input\_notification, set\_io\_output\_automatic\_initialize, set\_io\_modbus\_type, set\_io\_unit\_id, set\_io\_input\_address, set\_io\_output\_address, set\_io\_poll\_interval, set\_io\_slave\_control\_type, set\_io\_master\_control\_type, set\_io\_tcp\_type, set\_io\_multi\_connection\_number, set\_io\_modbus\_peer\_address, set\_io\_modbus\_peer\_port, set\_io\_modbus\_local\_port, set\_io\_event\_notification, set\_event\_notification\_email, set\_io\_event\_notification\_port, set\_io\_input\_valid\_time, set\_io\_macro, set\_io\_port\_macro, set\_io\_macro\_text, set\_io\_output\_delay, set\_io\_output\_initial\_state, set\_io\_comment.

**Remarks****5.5.19 set\_io\_event\_notification**

```
int set_io_event_notification(unsigned char *mac_address, int onoff)
```

**Description**

*set\_io\_event\_notification*은 I/O 제품의 [입력 또는 출력포트 변경 알림(전자메일)] 기능 선택 여부를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

**Return values**

*set\_io\_event\_notification*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_event\_notification*이 1을 반환하지 않는 경우 -3을 반환합니다.

## Examples

### See also

is\_io, set\_io\_http, set\_io\_http\_port, set\_io\_html\_size, set\_io\_modbus, set\_io\_input\_notification, set\_io\_output\_automatic\_initialize, set\_io\_modbus\_type, set\_io\_unit\_id, set\_io\_input\_address, set\_io\_output\_address, set\_io\_poll\_interval, set\_io\_slave\_control\_type, set\_io\_master\_control\_type, set\_io\_tcp\_type, set\_io\_multi\_connection\_number, set\_io\_modbus\_peer\_address, set\_io\_modbus\_peer\_port, set\_io\_modbus\_local\_port, is\_event\_notification, set\_io\_event\_notification, set\_event\_notification\_email, set\_io\_event\_notification\_port, set\_io\_input\_valid\_time, set\_io\_macro, set\_io\_port\_macro, set\_io\_macro\_text, set\_io\_output\_delay, set\_io\_output\_initial\_state, set\_io\_comment.

### Remarks

#### 5.5.20 set\_event\_notification\_email

```
int set_event_notification_email(unsigned char *mac_address, char *email);
```

### Description

*set\_event\_notification\_email*은 I/O 제품의 입력 또는 출력포트에 변경이 발생한 경우 알림을 발송할 [전자메일 주소]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*email*

전자메일 주소가 저장되어있는 포인터입니다.

### Return values

*set\_event\_notification\_email*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_event\_notification*이 1을 반환하지 않는 경우 -2을 반환합니다.

## Examples

### See also

is\_io, set\_io\_http, set\_io\_http\_port, set\_io\_html\_size, set\_io\_modbus, set\_io\_input\_notification,

set\_io\_output\_automatic\_initialize, set\_io\_modbus\_type, set\_io\_unit\_id, set\_io\_input\_address, set\_io\_output\_address, set\_io\_poll\_interval, set\_io\_slave\_control\_type, set\_io\_master\_control\_type, set\_io\_tcp\_type, set\_io\_multi\_connection\_number, set\_io\_modbus\_peer\_address, set\_io\_modbus\_peer\_port, set\_io\_modbus\_local\_port, is\_event\_notification, set\_io\_event\_notification, set\_event\_notification\_email, set\_io\_event\_notification\_port, set\_io\_input\_valid\_time, set\_io\_macro, set\_io\_port\_macro, set\_io\_macro\_text, set\_io\_output\_delay, set\_io\_output\_initial\_state, set\_io\_comment.

## Remarks

[전자메일 주소]는 NULL을 포함하여 최대 64바이트입니다.

### 5.5.21 set\_io\_event\_notification\_port

```
int set_io_event_notification_port(unsigned char *mac_address, int port_flag,
                                   int port_index, int onoff);
```

## Description

*set\_io\_event\_notification\_port*는 I/O 제품에 설치된 입력 또는 출력포트의 [입력 또는 출력포트 변경 알림(전자메일)] 기능 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소입니다.

*int port\_flag*

*port\_flag*가 0이면 입력포트를 의미하고 1이면 출력포트를 의미합니다.

*int port\_index*

입력 또는 출력포트의 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_io\_event\_notification\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_event\_notification*이 1을 반환하지 않는 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *is\_event\_notification*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.22 *set\_io\_input\_valid\_time*

```
int set_io_input_valid_time(unsigned char *mac_address, int port_index,
                           int input_valid_time);
```

## Description

*set\_io\_input\_valid\_time*은 *port\_index*로 지정된 입력포트의 [신호유지 시간(밀리초)]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

입력포트의 순번으로 0부터 시작합니다.

*int input\_valid\_time*

*port\_index*로 지정된 입력포트의 신호유지 시간(밀리초)입니다.

## Return values

*set\_io\_input\_valid\_time*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1



*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*input\_valid\_time*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.23 set\_io\_macro

```
int set_io_macro(unsigned char *mac_address, int onoff);
```

## Description

*set\_io\_macro*는 I/O 제품의 [매크로 기능] 선택 여부를 변경합니다. *get\_io\_macro\_type*이 0을 반환하는 경우에만 사용할 수 있습니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_io\_macro*의 반환 값은 다음과 같습니다.

1





설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*get\_io\_macro\_type*이 0을 반환하지 않는 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.24 set\_io\_port\_macro

```
int set_io_port_macro(unsigned char *mac_address, int port_index, int onoff);
```

## Description

*set\_io\_port\_macro*는 *port\_index*로 지정된 출력포트의 [매크로 기능] 선택 여부를 변경합니다. *get\_io\_macro\_type*이 1을 반환하는 경우에만 사용할 수 있습니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

**Return values**

*set\_io\_macro\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*get\_io\_macro\_type*이 1을 반환하지 않는 경우 -3을 반환합니다.

-4

*port\_index*에 유효하지 않은 값을 입력한 경우 -4를 반환합니다.

**Examples****See also**

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

**Remarks****5.5.25 set\_io\_macro\_text**

```
int set_io_macro_text(unsigned char *mac_address, int port_index, char *macro_text);
```

**Description**

*set\_io\_macro\_text*는 *port\_index*로 지정된 출력포트의 매크로 문자열을 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*char \*macro\_text*

*port\_index*로 지정된 출력포트의 매크로 문자열입니다.

## Return values

*set\_io\_macro\_text*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*macro\_text*에 입력한 매크로 문자열이 문법에 맞지 않으면 -3을 반환합니다.

-4

*port\_index*에 유효하지 않은 값을 입력한 경우 -4를 반환합니다.

## Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*,  
*set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*,  
*set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*,  
*set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*,  
*set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*,  
*set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*,  
*set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

매크로 문자열은 NULL을 제외하고 최대 32바이트입니다. 매크로 문자열 길이는 **IO\_SCRIPT\_LEN**에 정의 되어 있습니다.

## 5.5.26 set\_io\_output\_delay

```
int set_io_output_delay(unsigned char *mac_address, int port_index, int output_delay);
```

### Description

*set\_io\_output\_delay*는 *port\_index*로 지정된 출력포트의 [출력지연(밀리초)]을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*int output\_delay*

*port\_index*로 지정된 출력포트의 출력지연(밀리초)입니다.

## Return values

*set\_io\_output\_delay*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*port\_index*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

### 5.5.27 set\_io\_output\_initial\_state

```
int set_io_output_initial_state(unsigned char *mac_address, int port_index, int onoff);
```

## Description

*set\_io\_output\_initial\_state*는 *port\_index*로 지정된 출력포트의 [초기상태]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_index*

출력포트의 순번으로 0부터 시작합니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

### Return values

*set\_io\_output\_initial\_state*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*port\_index*에 유효하지 않은 값을 입력한 경우 -3를 반환합니다.

### Examples

### See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

### Remarks

#### 5.5.28 set\_io\_comment

```
int set_io_comment(unsigned char *mac_address, int port_flag, int port_index,
                  char *comment);
```

### Description

*set\_io\_comment*는 I/O 제품에 설치된 입력 또는 출력포트의 [I/O 포트 설명]을 변경합니다.

### Parameters

*unsigned char \*mac\_address*



*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int port\_flag*

*port\_flag*가 0이면 입력포트를 의미하고 1이면 출력포트를 의미합니다.

*int port\_index*

입력 또는 출력포트의 순번으로 0부터 시작합니다.

*char \*comment*

*port\_index*로 지정된 입력 또는 출력포트의 I/O 포트 설명이 저장된 포인터입니다.

## Return values

*set\_io\_comment*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_io*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*port\_index*에 유효하지 않은 값을 입력한 경우 -3를 반환합니다.

## Examples

## See also

*is\_io*, *set\_io\_http*, *set\_io\_http\_port*, *set\_io\_html\_size*, *set\_io\_modbus*, *set\_io\_input\_notification*, *set\_io\_output\_automatic\_initialize*, *set\_io\_modbus\_type*, *set\_io\_unit\_id*, *set\_io\_input\_address*, *set\_io\_output\_address*, *set\_io\_poll\_interval*, *set\_io\_slave\_control\_type*, *set\_io\_master\_control\_type*, *set\_io\_tcp\_type*, *set\_io\_multi\_connection\_number*, *set\_io\_modbus\_peer\_address*, *set\_io\_modbus\_peer\_port*, *set\_io\_modbus\_local\_port*, *set\_io\_event\_notification*, *set\_event\_notification\_email*, *set\_io\_event\_notification\_port*, *set\_io\_input\_valid\_time*, *set\_io\_macro*, *set\_io\_port\_macro*, *set\_io\_macro\_text*, *set\_io\_output\_delay*, *set\_io\_output\_initial\_state*, *set\_io\_comment*.

## Remarks

[I/O 포트 설명]은 NULL을 제외하고 최대 16바이트입니다. [I/O 포트 설명]길이는 *IO\_COMMENT\_LEN*에 정의 되어 있습니다.

## 5.6 무선랜

### 5.6.1 set\_wlan\_type

```
int set_wlan_type(unsigned char *mac_address, int wlan_type);
```

## Description

*set\_wlan\_type*은 무선랜 제품의 [무선랜 종류]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int wlan\_type*

[무선랜 종류]를 나타내며 아래와 같은 값을 가질 수 있습니다.

0

애드혹

1

인프라스트럭처

2

Soft AP

## Return values

*set\_wlan\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*wlan\_type*에 유효하지 않은 값을 입력한 경우 -3를 반환합니다.

-4

*wlan\_type*에 **Soft AP(2)**를 입력한 경우 *is\_wlan\_soft\_ap*가 1을 반환하지 않으면 -4를 반환합니다.

## Examples

## See also

*is\_wlan*, *is\_wlan\_soft\_ap*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

## Remarks



### 5.6.2 set\_wlan\_channel

```
int set_wlan_channel(unsigned char *mac_address, int channel);
```

#### Description

*set\_wlan\_channel*은 무선랜 제품의 [채널]을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int channel*

변경할 무선랜 채널번호입니다.

#### Return values

*set\_wlan\_channel*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

#### Examples

#### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*,  
*set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*,  
*set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*,  
*set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*,  
*set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*,  
*set\_wlan\_wpa\_enterprise\_pwd*.

#### Remarks

무선랜 제품의 [채널]은 [무선랜 종류]가 애드혹 또는 Soft-AP인 경우에만 사용됩니다.

### 5.6.3 set\_wlan\_ssid

```
int set_wlan_ssid(unsigned char *mac_address, char *ssid);
```

#### Description





*set\_wlan\_ssid*는 무선랜 제품의 [SSID]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ssid*

무선랜 제품의 SSID가 저장되어 있는 포인터입니다.

### Return values

*set\_wlan\_ssid*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

### Examples

### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*,  
*set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*,  
*set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*,  
*set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*,  
*set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*,  
*set\_wlan\_wpa\_enterprise\_pwd*.

### Remarks

[SSID]는 NULL을 제외하고 최대 32바이트입니다.

#### 5.6.4 set\_wlan\_antenna

```
int set_wlan_antenna(unsigned char *mac_address, int antenna_type);
```

### Description

*set\_wlan\_antenna*는 무선랜 제품의 [안테나]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int antenna\_type*

무선랜 제품의 [안테나] 설정 값을 나타내며 아래와 같은 값을 가질 수 있습니다.

0

내장안테나

1

외장안테나

### Return values

*set\_wlan\_antenna*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_antenna*가 1을 반환하지 않는 경우 -3을 반환합니다.

### Examples

### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *is\_wlan\_antenna*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

### Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5.6.5 set\_wlan\_phy\_mode

```
int set_wlan_phy_mode(unsigned char *mac_address, int phy_mode);
```

### Description

*set\_wlan\_phy\_mode*는 무선랜 제품의 [무선 고급설정]들 중에서 [Phy Mode]를 변경합니다.

### Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int phy\_mode*

무선랜 제품의 [Phy Mode] 설정 값을 나타내며 아래와 같은 값을 가질 수 있습니다.

1	802.11
2	802.11b
3	802.11b/g

## Return values

*set\_wlan\_phy\_mode*의 반환 값은 다음과 같습니다.

- 1  
설정 값 변경에 성공하면 1을 반환합니다.
- 1  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.
- 2  
*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.
- 3  
*is\_wlan\_phy\_mode*가 1을 반환하지 않는 경우 -3를 반환합니다.
- 4  
*phy\_mode*에 유효하지 않은 값을 입력한 경우 -4를 반환합니다.

## Examples

### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *is\_wlan\_phy\_mode*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

### Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.6 set\_wlan\_short\_preamble

```
int set_wlan_short_preamble(unsigned char *mac_address, int onoff);
```

#### Description

*set\_wlan\_short\_preamble*은 무선랜 제품의 [무선 고급설정]들 중에서 [Short Preamble] 선택 여부를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

#### Return values

*set\_wlan\_short\_preamble*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_phy\_mode*가 1을 반환하지 않는 경우 -3를 반환합니다.

#### Examples

#### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *is\_wlan\_phy\_mode*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

#### Remarks



[무선 고급설정]들 중에서 [Phy Mode]가 2(802.11b) 또는 3(802.11b/g)인 경우에만 사용됩니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.7 set\_wlan\_short\_slot

```
int set_wlan_short_slot(unsigned char *mac_address, int onoff);
```

#### Description

*set\_wlan\_short\_slot*은 무선랜 제품의 [무선 고급설정]들 중에서 [Short Slot] 선택 여부를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

#### Return values

*set\_wlan\_short\_slot*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_phy\_mode*가 1을 반환하지 않는 경우 -3를 반환합니다.

#### Examples

#### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *is\_wlan\_phy\_mode*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*,

set\_wlan\_wpa\_enterprise\_pwd.

## Remarks

[무선 고급설정]들 중에서 [Phy Mode]가 3(802.11b/g)인 경우에만 사용할 수 있습니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.8 set\_wlan\_cts\_protection

```
int set_wlan_cts_protection(unsigned char *mac_address, int onoff);
```

## Description

*set\_wlan\_cts\_protection*은 무선랜 제품의 [무선 고급설정]들 중에서 [CTS Protection] 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_wlan\_cts\_protection*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_phy\_mode*가 1을 반환하지 않는 경우 -3를 반환합니다.

## Examples

## See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *is\_wlan\_phy\_mode*,

set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection,  
 set\_wlan\_background\_scan, set\_wlan\_encryption\_type, set\_wlan\_authentication\_type,  
 set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key,  
 set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id,  
 set\_wlan\_wpa\_enterprise\_pwd.

## Remarks

[무선 고급설정]들 중에서 [Phy Mode]가 3(802.11b/g)인 경우에만 사용할 수 있습니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.9 set\_wlan\_background\_scan

```
int set_wlan_background_scan(unsigned char *mac_address, int onoff);
```

## Description

*set\_wlan\_background\_scan*은 무선랜 제품의 [무선 고급설정]들 중에서 [Background Scan] 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_wlan\_background\_scan*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_background\_scan*이 1을 반환하지 않는 경우 -3를 반환합니다.

## Examples



## See also

is\_wlan, set\_wlan\_type, set\_wlan\_channel, set\_wlan\_ssid, set\_wlan\_antenna, set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection, is\_wlan\_background\_scan, set\_wlan\_background\_scan, set\_wlan\_encryption\_type, set\_wlan\_authentication\_type, set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key, set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id, set\_wlan\_wpa\_enterprise\_pwd.

## Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.10 set\_wlan\_encryption\_type

```
int set_wlan_encryption_type(unsigned char *mac_address, int encryption_type);
```

## Description

*set\_wlan\_encryption\_type*은 무선랜 제품의 [암호화 방식]을 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int encryption\_type*

무선랜 제품의 [암호화 방식]을 나타내며 아래와 같은 값을 가질 수 있습니다.

0

없음

1

WEP

2

WPA

## Return values

*set\_wlan\_encryption\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3



*is\_wlan\_rsn*이 0을 반환하지 않는 경우 -3을 반환합니다.

-4

*encryption\_type*에 유효하지 않은 값을 입력한 경우 -4를 반환합니다.

-5

[무선랜 종류]가 애드훅(0)인 경우 *encryption\_type*에 WPA(2)를 입력하면 -5를 반환합니다.

## Examples

### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *is\_wlan\_rsn*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

### Remarks

#### 5.6.11 *set\_wlan\_authentication\_type*

```
int set_wlan_authentication_type(unsigned char *mac_address, int authentication_type);
```

### Description

*set\_wlan\_authentication\_type*은 무선랜 제품의 [인증 방식] 또는 WPA [인증 방식/암호화 방법]을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소입니다.

*int encryption\_type*

무선랜 제품의 [인증 방식] 또는 WPA [인증 방식/암호화 방법]을 나타내며 *get\_wlan\_encryption\_type*의 반환 값에 따라서 *authentication\_type*에 입력 가능한 값과 의미는 달라집니다.

*get\_wlan\_encryption\_type*이 0(없음)을 반환하는 경우에는 아래와 같은 값을 가질 수 있습니다.

1

개방 모드

*get\_wlan\_encryption\_type*이 1(WEP)을 반환하는 경우에는 아래와 같은 값을 가질 수 있습니다.

- 1  
개방 모드
- 2  
공유 모드
- 3  
자동 모드

*get\_wlan\_encryption\_type*이 **2(WPA)**을 반환하는 경우에는 아래와 같은 값을 가질 수 있습니다.

- 0  
WPA PSK – TKIP
- 1  
WPA PSK – AES
- 2  
WPA PSK – TKIP/AES
- 3  
WPA2 PSK – TKIP
- 4  
WPA2 PSK – AES
- 5  
WPA2 PSK- TKIP/AES

### Return values

*set\_wlan\_authentication\_type*의 반환 값은 다음과 같습니다.

- 1  
설정 값 변경에 성공하면 1을 반환합니다.
  - 1  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.
  - 2  
*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.
  - 3  
*is\_wlan\_rsn*이 0을 반환하지 않는 경우 -3을 반환합니다.
  - 4  
무선랜 제품의 [암호화 방식]이 **0(없음)** 또는 **1(WEP)**일 때 *authentication\_type*에 유효하지 않은 값을 입력한 경우 -4를 반환합니다.
  - 5  
무선랜 제품의 [암호화 방식]이 **2(WPA)**일 때 *authentication\_type*에 유효하지 않은 값을 입력한 경우 -5를 반환합니다.
- is\_wlan\_authentication\_type*이 0을 반환하면 *authentication\_type*에 **0(WPA PSK – TKIP)**

또는 4(WPA2 PSK – AES)만 입력할 수 있습니다.

## Examples

## See also

is\_wlan, set\_wlan\_type, set\_wlan\_channel, set\_wlan\_ssid, set\_wlan\_antenna, set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection, set\_wlan\_background\_scan, is\_wlan\_rsn, set\_wlan\_encryption\_type, is\_wlan\_authentication\_type, set\_wlan\_authentication\_type, set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key, set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id, set\_wlan\_wpa\_enterprise\_pwd.

## Remarks

### 5.6.12 set\_wlan\_wep\_key\_length

```
int set_wlan_wep_key_length(unsigned char *mac_address, int wep_key_length);
```

## Description

*set\_wlan\_wep\_key\_length*은 무선랜 제품의 WEP 키 길이를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int wep\_key\_length*

무선랜 제품의 WEP 키 길이를 나타내며 아래와 같은 값을 가질 수 있습니다.

- |   |       |
|---|-------|
| 1 | 64비트  |
| 2 | 128비트 |

## Return values

*set\_wlan\_wep\_key\_length*의 반환 값은 다음과 같습니다.

- |    |  |
|----|--|
| 1  | 설정 값 변경에 성공하면 1을 반환합니다.                                    |
| -1 | <i>mac_address</i> 을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다. |
| -2 | <i>is_wlan</i> 이 1을 반환하지 않는 경우 -2를 반환합니다.                  |

-3

*wep\_key\_length*에 유효하지 않은 값을 입력한 경우 -3를 반환합니다.

## Examples

## See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*,  
*set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*,  
*set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*,  
*set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*,  
*set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*,  
*set\_wlan\_wpa\_enterprise\_pwd*.

## Remarks

### 5.6.13 *set\_wlan\_wep\_key\_index*

```
int set_wlan_wep_key_index(unsigned char *mac_address, int wep_key_index);
```

## Description

*set\_wlan\_wep\_key\_index*은 무선랜 제품이 사용하는 WEP 키 순번을 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int wep\_key\_index*

변경할 WEP 키 순번으로 0 ~ 3입니다.

## Return values

*set\_wlan\_wep\_key\_index*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*wep\_key\_index*에 유효하지 않은 값을 입력한 경우 -3를 반환합니다.

## Examples

### See also

is\_wlan, set\_wlan\_type, set\_wlan\_channel, set\_wlan\_ssid, set\_wlan\_antenna, set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection, set\_wlan\_background\_scan, set\_wlan\_encryption\_type, set\_wlan\_authentication\_type, set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key, set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id, set\_wlan\_wpa\_enterprise\_pwd.

### Remarks

#### 5.6.14 set\_wlan\_wep\_key\_data\_type

```
int set_wlan_wep_key_data_type(unsigned char *mac_address, int wep_key_data_type);
```

### Description

*set\_wlan\_wep\_key\_data\_type*은 WEP 키가 저장될 데이터 형식을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int wep\_key\_data\_type*

WEP 키가 저장될 데이터 형식을 나타내며 아래와 같은 값을 가질 수 있습니다.

0  
16진수  
1  
ASCII

### Return values

*set\_wlan\_wep\_key\_data\_type*의 반환 값은 다음과 같습니다.

1  
설정 값 변경에 성공하면 1을 반환합니다.  
-1  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.  
-2  
*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.  
-3  
*wep\_key\_data\_type*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

## Examples

### See also

is\_wlan, set\_wlan\_type, set\_wlan\_channel, set\_wlan\_ssid, set\_wlan\_antenna, set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection, set\_wlan\_background\_scan, set\_wlan\_encryption\_type, set\_wlan\_authentication\_type, set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key, set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id, set\_wlan\_wpa\_enterprise\_pwd.

### Remarks

#### 5.6.15 set\_wlan\_wep\_key

```
int set_wlan_wep_key(unsigned char *mac_address, char *wep_key);
```

### Description

*set\_wlan\_wep\_key*는 무선랜 제품이 사용할 WEP 키를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*wep\_key*

무선랜 제품이 사용할 WEP 키가 저장되어 있는 포인터입니다.

### Return values

*set\_wlan\_wep\_key*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*wep\_key*에 입력한 WEP 키 문자열 길이가 유효하지 않은 경우 -3을 반환합니다.

-4

*wep\_key*에 입력한 WEP 키 문자열 내용이 유효하지 않은 경우 -4를 반환합니다.

## Examples



```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    int wep_key_length = 1; // 64bit
    int wep_key_index = 1; // index #1
    int wep_key_data_type = 1; // ASCII
    char *wep_key = "12345";

    if(is_wlan(mac_address) == 1)
    {
        set_wlan_wep_key_length(mac_address, wep_key_length);
        set_wlan_wep_key_index(mac_address, wep_key_index);
        set_wlan_wep_key_data_type(mac_address, wep_key_data_type);
        set_wlan_wep_key(mac_address, wep_key);
    }
}

```

### See also

is\_wlan, set\_wlan\_type, set\_wlan\_channel, set\_wlan\_ssid, set\_wlan\_antenna, set\_wlan\_phy\_mode, set\_wlan\_short\_preamble, set\_wlan\_short\_slot, set\_wlan\_cts\_protection, set\_wlan\_background\_scan, set\_wlan\_encryption\_type, set\_wlan\_authentication\_type, set\_wlan\_wep\_key\_length, set\_wlan\_wep\_key\_index, set\_wlan\_wep\_key\_data\_type, set\_wlan\_wep\_key, set\_wlan\_wpa\_passphrase, set\_wlan\_shared\_key, set\_wlan\_wpa\_enterprise, set\_wlan\_wpa\_enterprise\_id, set\_wlan\_wpa\_enterprise\_pwd.

### Remarks

WEP 키 길이가 **64비트**이면 *wep\_key*에 **10자리의 16진수 코드( 0~9, A~F, a~f )** 또는 **5자리의 ASCII코드**를 입력해야합니다.

WEP 키 길이가 **128비트**이면 *wep\_key*에 **26자리의 16진수 코드( 0~9, A~F, a~f )** 또는 **13자리의 ASCII코드**를 입력해야합니다.

## 5.6.16 set\_wlan\_wpa\_passphrase

```
int set_wlan_wpa_passphrase(unsigned char *mac_address, char *wpa_pp);
```

### Description

*set\_wlan\_wpa\_passphrase*는 무선랜 제품의 WPA 암호문을 변경합니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.*char \*wpa\_pp*

무선랜 제품이 사용할 WPA 암호문이 저장되어 있는 포인터입니다.

**Return values***set\_wlan\_wpa\_passphrase*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wlan\_rsn*이 0을 반환하지 않는 경우 -3을 반환합니다.**Examples****See also**

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*,  
*set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*,  
*set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*,  
*set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*,  
*set\_wlan\_shared\_key*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*,  
*set\_wlan\_wpa\_enterprise\_pwd*.

**Remarks**

WPA 암호문은 NULL을 포함해서 최대 64바이트입니다.

**5.6.17 set\_wlan\_shared\_key**

```
int set_wlan_shared_key(unsigned char *mac_address, char *shared_key);
```

**Description***set\_wlan\_shared\_key*는 무선랜 제품의 보안 설정 중에서 [Shared Key]를 변경합니다.**Parameters***unsigned char \*mac\_address*



*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.  
`char *shared_key`

Shared Key값이 저장되어 있는 포인터입니다.

### Return values

`set_wlan_shared_key`의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

`mac_address`을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

`is_wlan`이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

`is_wlan_rsn`이 1을 반환하지 않는 경우 -3을 반환합니다.

-4

[무선랜 종류]가 애드혹(0) 또는 Soft AP(2)인 경우 입력한 Shared Key값을 WEP 키 값으로 사용할 수 없으면 -4를 반환합니다.

### Examples

### See also

`is_wlan`, `set_wlan_type`, `set_wlan_channel`, `set_wlan_ssid`, `set_wlan_antenna`, `set_wlan_phy_mode`,  
`set_wlan_short_preamble`, `set_wlan_short_slot`, `set_wlan_cts_protection`, `set_wlan_background_scan`,  
`set_wlan_encryption_type`, `set_wlan_authentication_type`, `set_wlan_wep_key_length`,  
`set_wlan_wep_key_index`, `set_wlan_wep_key_data_type`, `set_wlan_wep_key`, `set_wlan_wpa_passphrase`,  
`set_wlan_shared_key`, `set_wlan_wpa_enterprise`, `set_wlan_wpa_enterprise_id`,  
`set_wlan_wpa_enterprise_pwd`.

### Remarks

[Shared Key]는 NULL을 포함해서 최대 64바이트입니다. 보다 자세한 내용은 해당 제품 사용 설명서를 참조하시기 바랍니다.

## 5.6.18 set\_wlan\_wpa\_enterprise

```
int set_wlan_wpa_enterprise(unsigned char *mac_address, int enterprise_type);
```

### Description

`set_wlan_wpa_enterprise`는 무선랜 제품의 [보안 설정] 중에서 [802.1X] 설정 값을 변경합니다.

### Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int enterprise\_type*

[802.1X]종류를 나타내며 아래와 같은 값을 가질 수 있습니다.

0	사용안함
1	EAP TLS
2	EAP TTLS
3	PEAP

## Return values

*set\_wlan\_wpa\_enterprise*의 반환 값은 다음과 같습니다.

1	설정 값 변경에 성공하면 1을 반환합니다.
-1	<i>mac_address</i> 을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.
-2	<i>is_wlan</i> 이 1을 반환하지 않는 경우 -2를 반환합니다.
-3	<i>is_wpa_enterprise</i> 가 1을 반환하지 않는 경우 -3을 반환합니다.

## Examples

## See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *is\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

## Remarks

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.6.19 set\_wlan\_wpa\_enterprise\_id

```
int set_wlan_wpa_enterprise_id(unsigned char *mac_address, char *enterprise_id);
```

#### Description

*set\_wlan\_wpa\_enterprise\_id*는 무선랜 제품의 [보안 설정] 중에서 802.1X [아이디]를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*enterprise\_id*

802.1X 아이디가 저장되어 있는 포인터입니다.

#### Return values

*set\_wlan\_wpa\_enterprise\_id*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wpa\_enterprise*가 1을 반환하지 않는 경우 -3을 반환합니다.

#### Examples

#### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *is\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

#### Remarks

802.1X [아이디]는 NULL을 제외하고 최대 32바이트입니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5.6.20 set\_wlan\_wpa\_enterprise\_pwd

```
int set_wlan_wpa_enterprise_pwd(unsigned char *mac_address, char *enterprise_pwd);
```

### Description

*set\_wlan\_wpa\_enterprise\_pwd*는 무선랜 제품의 [보안 설정] 중에서 802.1X [비밀번호]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*enterprise\_pwd*

802.1X 비밀번호가 저장되어 있는 포인터입니다.

### Return values

*set\_wlan\_wpa\_enterprise\_pwd*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_wlan*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

*is\_wpa\_enterprise*가 1을 반환하지 않는 경우 -3을 반환합니다.

### Examples

### See also

*is\_wlan*, *set\_wlan\_type*, *set\_wlan\_channel*, *set\_wlan\_ssid*, *set\_wlan\_antenna*, *set\_wlan\_phy\_mode*, *set\_wlan\_short\_preamble*, *set\_wlan\_short\_slot*, *set\_wlan\_cts\_protection*, *set\_wlan\_background\_scan*, *set\_wlan\_encryption\_type*, *set\_wlan\_authentication\_type*, *set\_wlan\_wep\_key\_length*, *set\_wlan\_wep\_key\_index*, *set\_wlan\_wep\_key\_data\_type*, *set\_wlan\_wep\_key*, *set\_wlan\_wpa\_passphrase*, *set\_wlan\_shared\_key*, *is\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise*, *set\_wlan\_wpa\_enterprise\_id*, *set\_wlan\_wpa\_enterprise\_pwd*.

### Remarks

무선랜 제품의 [보안 설정] 중에서 [802.1X] 설정 값이 2(EAP TTLS) 또는 3(PEAP)를 반환하는 경우에만 사용됩니다.

802.1X [비밀번호]는 NULL을 제외하고 최대 32바이트입니다.

보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5.7 부가기능

### 5.7.1 set\_telnet

```
int set_telnet(unsigned char *mac_address, int onoff);
```

#### Description

*set\_telnet*은 [텔넷] 기능의 선택 여부를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

#### Return values

*set\_telnet*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

*set\_telnet*, *set\_send\_mac\_address*, *set\_ssl*, *set\_ssh*, *set\_ip4\_address\_search*, *set\_remote\_debug*, *set\_tcp\_multi\_connection*, *set\_power\_management*, *set\_comment*.

#### Remarks

### 5.7.2 set\_send\_mac\_address

```
int set_send_mac_address(unsigned char *mac_address, int onoff);
```

## Description

*set\_send\_mac\_address*는 [MAC 주소 전송] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_send\_mac\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_send\_mac\_address*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

[SSL 보안통신]과 [MAC 주소 전송]을 동시에 사용하지 못하는 제품(ezTCP)은 두 기능을 동시에 설정하는 경우 -3을 반환합니다.

-4

[SSH 보안통신]이 설정되어있는 경우 -4를 반환합니다.

## Examples

### See also

*set\_telnet*, *is\_send\_mac\_address*, *set\_send\_mac\_address*, *set\_ssl*, *set\_ssh*, *set\_ip4\_address\_search*, *set\_remote\_debug*, *set\_tcp\_multi\_connection*, *set\_power\_management*, *set\_comment*.

## Remarks

### 5.7.3 set\_ssl

```
int set_ssl(unsigned char *mac_address, int onoff);
```

## Description

*set\_ssl*은 [SSL 보안통신] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_ssl*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ssl*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

[SSH 보안통신]이 설정되어있는 경우 -3를 반환합니다.

-4

[SSL 보안통신]과 [MAC 주소 전송]을 동시에 사용하지 못하는 제품(ezTCP)은 두 기능을 동시에 설정하는 경우 -4을 반환합니다.

-5

RS-485 또는 RS-422과 [SSL 보안통신]을 동시에 사용하지 못하는 제품(ezTCP)인 경우 -5를 반환합니다.

-6

시리얼 포트의 [통신모드]가 U2S - UDP인 경우 -6를 반환합니다.

-7

IPv6 기능이 선택된 경우 -7을 반환합니다.

-8

시리얼 포트의 [시리얼 포트 설정/상태 전송(RFC2217)]가 선택된 경우 -8을 반환합니다.

## Examples

## See also



set\_telnet, set\_send\_mac\_address, is\_ssl, set\_ssl, set\_ssh, set\_ip4\_address\_search, set\_remote\_debug, set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

## Remarks

### 5.7.4 set\_ssh

```
int set_ssh(unsigned char *mac_address, int onoff);
```

## Description

*set\_ssh*는 [SSH 보안통신] 기능의 선택 여부를 반환합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_ssh*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_ssh*가 1을 반환하지 않는 경우 -2를 반환합니다.

-3

[SSL 보안통신]이 설정되어있는 경우 -3를 반환합니다.

-4

시리얼 포트의 [시리얼 종류]가 RS-232가 아닌 경우 -4을 반환합니다.

-5

시리얼 포트의 [통신모드]가 T2S - TCP 서버가 아닌 경우 -5를 반환합니다.

-6

[MAC 주소 전송]이 선택된 경우 -6를 반환합니다.



-7

시리얼 포트의 [시리얼 포트 설정/상태 전송(RFC2217)]가 선택된 경우 -7을 반환합니다.

## Examples

## See also

set\_telnet, set\_send\_mac\_address, set\_ssl, is\_ssh, set\_ssh, set\_ip4\_address\_search, set\_remote\_debug, set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

## Remarks

### 5.7.5 set\_ip4\_address\_search

```
int set_ip4_address_search(unsigned char *mac_address, int onoff);
```

## Description

*set\_ip4\_address\_search*는 [IPv4 주소 검색] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

set\_ip4\_address\_search의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also



set\_telnet, set\_send\_mac\_address, set\_ssl, set\_ssh, set\_ip4\_address\_search, set\_remote\_debug,  
set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

## Remarks

### 5.7.6 set\_remote\_debug

```
int set_remote_debug(unsigned char *mac_address, int onoff);
```

## Description

*set\_remote\_debug*는 [디버깅 로그 보기] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_remote\_debug*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_remote\_debug*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

set\_telnet, set\_send\_mac\_address, set\_ssl, set\_ssh, set\_ip4\_address\_search, is\_remote\_debug,  
set\_remote\_debug, set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

## Remarks



### 5.7.7 set\_tcp\_multi\_connection

```
int set_tcp_multi_connection(unsigned char *mac_address, int onoff);
```

#### Description

*set\_tcp\_multi\_connection*는 [다중 접속] 기능의 선택 여부를 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

#### Return values

*set\_tcp\_multi\_connection*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_tcp\_multi\_connection*이 1을 반환하지 않는 경우 -2를 반환합니다.

-3

시리얼 포트의 [통신모드]가 T2S – TCP 서버가 아닌 경우 -3을 반환합니다.

-4

[MAC 주소 전송]이 선택된 경우 -4를 반환합니다.

-5

[SSL 보안통신]이 설정되어있는 경우 -5를 반환합니다.

-6

[SSH 보안통신]이 설정되어있는 경우 -6을 반환합니다.

#### Examples

#### See also

*set\_telnet*, *set\_send\_mac\_address*, *set\_ssl*, *set\_ssh*, *set\_ip4\_address\_search*, *set\_remote\_debug*,

is\_tcp\_multi\_connection, set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

## Remarks

### 5.7.8 set\_power\_management

```
int set_power_management(unsigned char *mac_address, int onoff);
```

## Description

*set\_power\_management*는 [전원 관리] 기능의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1

기능을 선택합니다.

## Return values

*set\_power\_management*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*is\_power\_management*가 1을 반환하지 않는 경우 -2를 반환합니다.

## Examples

## See also

set\_telnet, set\_send\_mac\_address, set\_ssl, set\_ssh, set\_ip4\_address\_search, set\_remote\_debug, set\_tcp\_multi\_connection, is\_power\_management, set\_power\_management, set\_comment.

## Remarks



### 5.7.9 set\_comment

```
int set_comment(unsigned char *mac_address, char *comment);
```

#### Description

*set\_comment*는 제품(ezTCP)의 **[설명]**을 변경합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*comment*

제품(ezTCP)의 **[설명]**이 저장되어 있는 포인터입니다.

#### Return values

*set\_comment*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

#### See also

set\_telnet, set\_send\_mac\_address, set\_ssl, set\_ssh, set\_ip4\_address\_search, set\_remote\_debug, set\_tcp\_multi\_connection, set\_power\_management, set\_comment.

#### Remarks

제품(ezTCP)의 **[설명]**은 NULL을 제외하고 최대 64바이트입니다.

### 5.7.10 set\_allowed\_mac\_address

```
int set_allowed_mac_address(unsigned char *mac_address, int index,  
                             char *in_mac_address);
```

#### Description

*set\_allowed\_mac\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 **[다음의 MAC 주소만 접근 가능]**에 입력되어있는 MAC 주소를 변경합니다.

#### Parameters



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int index*

MAC 주소 6바이트 중에서 변경할 MAC 주소의 순번으로 0 ~ 5입니다.

### Return values

*set\_allowed\_mac\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*index*에 유효하지 않은 값을 입력한 경우 -2를 반환합니다.

-3

*in\_mac\_address*에 유효하지 않은 값을 입력한 경우 -3을 반환합니다.

### Examples

### See also

*set\_allowed\_mac\_address*, *set\_allowed\_ip4\_address*, *set\_allowed\_ip4\_network\_mask\_type*,  
*set\_allowed\_ezmanager*, *set\_allowed\_ip6\_address*, *set\_allowed\_ip6\_subnet\_prefix\_length*.

### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 5.7.11 set\_allowed\_ip4\_address

```
int set_allowed_ip4_address(unsigned char *mac_address, char *ip4_address);
```

### Description

*set\_allowed\_ip4\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [IPv4 주소]를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip4\_address*

IPv4 주소가 저장되어 있는 포인터입니다.

**Return values**

*set\_allowed\_ip4\_address*의 반환 값은 다음과 같습니다.

*1*

설정 값 변경에 성공하면 1을 반환합니다.

*-1*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*set\_allowed\_mac\_address*, *set\_allowed\_ip4\_address*, *set\_allowed\_ip4\_network\_mask\_type*,  
*set\_allowed\_ezmanager*, *set\_allowed\_ip6\_address*, *set\_allowed\_ip6\_subnet\_prefix\_length*.

**Remarks**

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.7.12 set\_allowed\_ip4\_network\_mask\_type**

```
int set_allowed_ip4_network_mask_type(unsigned char *mac_address,
                                       int network_mask_type);
```

**Description**

*set\_allowed\_ip4\_network\_mask\_type*은 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [넷 마스크(IPv4 주소 대역)]의 형태를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int network\_mask\_type*

[넷 마스크(IPv4 주소 대역)] 설정 값을 나타내며 아래와 같은 값을 가질 수 있습니다.

*0*

255.255.255.255

*1*

255.255.255.0

*2*

255.255.0.0

*3*

255.0.0.0  
4  
0.0.0.0

## Return values

*set\_allowed\_ip4\_network\_mask\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*network\_mask\_type*에 유효하지 않는 값을 입력한 경우 -2를 반환합니다.

## Examples

## See also

*set\_allowed\_mac\_address*, *set\_allowed\_ip4\_address*, *set\_allowed\_ip4\_network\_mask\_type*,  
*set\_allowed\_ezmanager*, *set\_allowed\_ip6\_address*, *set\_allowed\_ip6\_subnet\_prefix\_length*.

## Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.7.13 set\_allowed\_ezmanager

```
int set_allowed_ezmanager(unsigned char *mac_address, int onoff);
```

## Description

*set\_allowed\_ezmanager*은 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [ezManager에도 적용]의 선택 여부를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int onoff*

해당 기능의 선택 여부를 결정하며 아래와 같은 값을 가질 수 있습니다.

0

기능 선택을 해제합니다.

1



기능을 선택합니다.

### Return values

*set\_allowed\_ezmanager*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

*set\_allowed\_mac\_address*, *set\_allowed\_ip4\_address*, *set\_allowed\_ip4\_network\_mask\_type*,  
*set\_allowed\_ezmanager*, *set\_allowed\_ip6\_address*, *set\_allowed\_ip6\_subnet\_prefix\_length*.

### Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 5.7.14 set\_allowed\_ip6\_address

```
int set_allowed_ip6_address(unsigned char *mac_address, char *ip6_address);
```

### Description

*set\_allowed\_ip6\_address*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 **[다음의 IP 주소 대역만 접근 가능]**에 입력되어있는 **[IPv6 주소]**를 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ip6\_address*

IPv6 주소가 저장되어 있는 포인터입니다.

### Return values

*set\_allowed\_ip6\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*ip6\_address*에 잘못된 형식의 IPv6 주소를 입력한 경우에는 -2를 반환합니다.

## Examples

## See also

*set\_allowed\_mac\_address*, *set\_allowed\_ip4\_address*, *set\_allowed\_ip4\_network\_mask\_type*,  
*set\_allowed\_ezmanager*, *set\_allowed\_ip6\_address*, *set\_allowed\_ip6\_subnet\_prefix\_length*.

## Remarks

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.7.15 *set\_allowed\_ip6\_subnet\_prefix\_length*

```
int set_allowed_ip6_subnet_prefix_length(unsigned char *mac_address,
                                         int subnet_prefix_length);
```

## Description

*set\_allowed\_ip6\_subnet\_prefix\_length*는 여러 가지 제품(ezTCP) 접근 제한 기능 중에서 [다음의 IP 주소 대역만 접근 가능]에 입력되어있는 [서브넷 접두사 길이]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int subnet\_prefix\_length*

변경할 서브넷 접두사 길이로 0 ~ 128입니다.

## Return values

*set\_allowed\_ip6\_subnet\_prefix\_length*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*subnet\_prefix\_length*에 유효하지 않는 값을 입력한 경우 -2를 반환합니다.

## Examples



**See also**

set\_allowed\_mac\_address, set\_allowed\_ip4\_address, set\_allowed\_ip4\_network\_mask\_type,  
set\_allowed\_ezmanager, set\_allowed\_ip6\_address, set\_allowed\_ip6\_subnet\_prefix\_length.

**Remarks**

제품(ezTCP) 접근 제한에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.7.16 set\_ip4\_change\_notification\_type**

```
int set_ip4_change_notification_type(unsigned char *mac_address, int notification_type);
```

**Description**

*set\_ip4\_change\_notification\_type*은 IPv4 주소 통보 기능의 [프로토콜]을 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int notification\_type*

[프로토콜] 설정 값을 나타내며 아래와 같은 값을 가질 수 있습니다.

0

사용안함

1

DDNS(dyndns.org)

2

TCP

3

UDP

**Return values**

*set\_ip4\_change\_notification\_type*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

*notification\_type*에 유효하지 않는 값을 입력한 경우 -2를 반환합니다.

## Examples

### See also

set\_ip4\_change\_notification\_type,  
set\_ip4\_change\_notification\_interval,  
set\_ip4\_change\_notification\_peer\_address,  
set\_ip4\_change\_notification\_ddns\_pwd.

set\_ip4\_change\_notification\_data\_type,  
set\_ip4\_change\_notification\_peer\_port,  
set\_ip4\_change\_notification\_ddns\_id,

### Remarks

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

#### 5.7.17 set\_ip4\_change\_notification\_data\_type

```
int set_ip4_change_notification_data_type(unsigned char *mac_address, int data_type);
```

### Description

*set\_ip4\_change\_notification\_data\_type*은 IPv4 주소 통보 기능의 [데이터 형식]을 변경합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int notification\_type*

[데이터 형식] 설정 값을 나타내며 아래와 같은 값을 가질 수 있습니다.

0  
ASCII  
1  
16진수

### Return values

*set\_ip4\_change\_notification\_data\_type*의 반환 값은 다음과 같습니다.

1  
설정 값 변경에 성공하면 1을 반환합니다.  
-1  
*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.  
-2  
*data\_type*에 유효하지 않는 값을 입력한 경우 -2를 반환합니다.

## Examples



**See also**

set\_ip4\_change\_notification\_type,  
set\_ip4\_change\_notification\_interval,  
set\_ip4\_change\_notification\_peer\_address,  
set\_ip4\_change\_notification\_ddns\_pwd.

set\_ip4\_change\_notification\_data\_type,  
set\_ip4\_change\_notification\_peer\_port,  
set\_ip4\_change\_notification\_ddns\_id,

**Remarks**

IPv4 주소 통보 기능의 [데이터 형식]은 [프로토콜]이 2(TCP) 또는 3(UDP)인 경우에만 사용됩니다.

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.7.18 set\_ip4\_change\_notification\_interval**

```
int set_ip4_change_notification_interval(unsigned char *mac_address, int interval);
```

**Description**

*set\_ip4\_change\_notification\_interval*은 IPv4 주소 통보 기능의 [통보 주기]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int interval*

변경할 통보 주기입니다.

**Return values**

*set\_ip4\_change\_notification\_interval*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

-2

[프로토콜]이 2(TCP) 또는 3(UDP)이 아니면 -2를 반환합니다.

**Examples****See also**

set\_ip4\_change\_notification\_type,  
set\_ip4\_change\_notification\_interval,  
set\_ip4\_change\_notification\_peer\_address,  
set\_ip4\_change\_notification\_ddns\_pwd.

set\_ip4\_change\_notification\_data\_type,  
set\_ip4\_change\_notification\_peer\_port,  
set\_ip4\_change\_notification\_ddns\_id,

**Remarks**

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.7.19 set\_ip4\_change\_notification\_peer\_port**

```
int set_ip4_change_notification_peer_port(unsigned char *mac_address, int peer_port);
```

**Description**

*set\_ip4\_change\_notification\_peer\_port*은 IPv4 주소 통보 기능의 [포트]를 변경합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int peer\_port*

변경할 IPv4 주소 통보 기능의 포트번호입니다.

**Return values**

*set\_ip4\_change\_notification\_peer\_port*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

*set\_ip4\_change\_notification\_type,*  
*set\_ip4\_change\_notification\_interval,*  
*set\_ip4\_change\_notification\_peer\_address,*  
*set\_ip4\_change\_notification\_ddns\_pwd.*

*set\_ip4\_change\_notification\_data\_type,*  
*set\_ip4\_change\_notification\_peer\_port,*  
*set\_ip4\_change\_notification\_ddns\_id,*

**Remarks**

IPv4 주소 통보 기능의 [포트]는 [프로토콜]이 2(TCP) 또는 3(UDP)인 경우에만 사용됩니다.

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

**5.7.20 set\_ip4\_change\_notification\_peer\_address**

```
int set_ip4_change_notification_peer_address(unsigned char *mac_address,
```

```
char *peer_address);
```

## Description

*set\_ip4\_change\_notification\_peer\_address*은 IPv4 주소 통보 기능의 [호스트 이름] 또는 [통보할 주소]를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*peer\_address*

호스트 이름 또는 통보할 주소가 저장되어 있는 포인터입니다.

## Return values

*set\_ip4\_change\_notification\_peer\_address*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*set\_ip4\_change\_notification\_type*,  
*set\_ip4\_change\_notification\_interval*,  
*set\_ip4\_change\_notification\_peer\_address*,  
*set\_ip4\_change\_notification\_ddns\_pwd*.

*set\_ip4\_change\_notification\_data\_type*,  
*set\_ip4\_change\_notification\_peer\_port*,  
*set\_ip4\_change\_notification\_ddns\_id*,

## Remarks

IPv4 주소 통보 기능의 [프로토콜]이 1(DDNS(dyndng.org))인 경우 [호스트 이름]을 변경합니다.

IPv4 주소 통보 기능의 [프로토콜]이 2(TCP) 또는 3(UDP)인 경우에는 [통보할 주소]를 변경합니다.

[호스트 이름] 또는 [통보할 주소]는 NULL을 포함하여 최대 64바이트입니다.

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.7.21 set\_ip4\_change\_notification\_ddns\_id

```
int set_ip4_change_notification_ddns_id(unsigned char *mac_address, char *ddns_id);
```

## Description



*set\_ip4\_change\_notification\_ddns\_id*는 IPv4 주소 통보 기능의 **[DDNS 아이디]**를 변경합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ddns\_id*

DDNS 아이디가 저장되어 있는 포인터입니다.

## Return values

*set\_ip4\_change\_notification\_ddns\_id*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*set\_ip4\_change\_notification\_type*,  
*set\_ip4\_change\_notification\_interval*,  
*set\_ip4\_change\_notification\_peer\_address*,  
*set\_ip4\_change\_notification\_ddns\_pwd*.

*set\_ip4\_change\_notification\_data\_type*,  
*set\_ip4\_change\_notification\_peer\_port*,  
*set\_ip4\_change\_notification\_ddns\_id*,

## Remarks

IPv4 주소 통보 기능의 **[DDNS 아이디]**는 **[프로토콜]**이 1(DDNS(dyndns.org)))인 경우에만 사용 됩니다.

**[DDNS 아이디]**는 NULL을 제외하고 최대 32바이트입니다.

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

### 5.7.22 set\_ip4\_change\_notification\_ddns\_pwd

```
int set_ip4_change_notification_ddns_pwd(unsigned char *mac_address,
                                         char *ddns_pwd);
```

## Description

*set\_ip4\_change\_notification\_ddns\_pwd*는 IPv4 주소 통보 기능의 **[DDNS 비밀번호]**를 변경합니다.

## Parameters





*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*ddns\_pwd*

DDNS 비밀번호가 저장되어 있는 포인터입니다.

## Return values

*set\_ip4\_change\_notification\_ddns\_pwd*의 반환 값은 다음과 같습니다.

1

설정 값 변경에 성공하면 1을 반환합니다.

-1

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

*set\_ip4\_change\_notification\_type,*  
*set\_ip4\_change\_notification\_interval,*  
*set\_ip4\_change\_notification\_peer\_address,*  
*set\_ip4\_change\_notification\_ddns\_pwd.*

*set\_ip4\_change\_notification\_data\_type,*  
*set\_ip4\_change\_notification\_peer\_port,*  
*set\_ip4\_change\_notification\_ddns\_id,*

## Remarks

IPv4 주소 통보 기능의 **[DDNS 비밀번호]**는 **[프로토콜]**이 1(DDNS(dyndns.org))인 경우에만 사용됩니다.

**[DDNS 비밀번호]**는 NULL을 제외하고 최대 16바이트입니다.

IP 주소 통보기능에 대한 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 6 설정 값 저장

### 6.1 EzTCP\_Write

```
int EzTCP_Write(unsigned char *mac_address, char *password, int udp_port_number,
                int *error);
```

#### Description

제품(ezTCP)의 MAC주소를 사용하여 설정 값을 제품(ezTCP)에 저장합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*password*

제품(ezTCP)에 비밀번호가 설정된 경우 현재 제품(ezTCP)에 설정되어있는 비밀번호를 입력해야 합니다.

*int udp\_port\_number*

*EzTCP\_Search* 또는 *EzTCP\_Read*로 제품(ezTCP)을 검색할 때 사용한 UDP 포트 번호 입니다. 0을 입력하면 기본값인 50005번, 50007번을 사용합니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

#### Return values

*EzTCP\_Write*의 반환 값은 다음과 같습니다.

*EZTCP\_SUCCESS*

제품(ezTCP)에 설정 값을 성공적으로 저장하면 EZTCP\_SUCCESS를 반환합니다.

*EZTCP\_ERR*

*EzTCP\_Write* 실행 중 오류가 발생한 경우 EZTCP\_ERR을 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.

*EZTCP\_ERR\_NO\_INFO*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 EZTCP\_ERR\_NO\_INFO를 *error*에 저장합니다.

*EZTCP\_ERR\_PWD\_LENGTH*

*password*에 입력한 비밀번호의 문자열 길이가 유효하지 않으면 EZTCP\_ERR\_PWD\_LENGTH를 *error*에 저장합니다. 비밀번호는 4 ~ 8자입니다.

*EZTCP\_ERR\_LOCAL\_IP*

제품(ezTCP)에 사용할 수 없는 IP주소를 저장하려는 경우 EZTCP\_ERR\_LOCAL\_IP를

*error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_0**

제품 로컬포트 번호를 중복해서 저장하려는 경우 EZTCP\_ERR\_LOCAL\_PORT\_0를 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_1**

[텔넷] 기능이 선택되어있을 때 제품 로컬포트 번호에 23번을 저장하려는 경우 EZTCP\_ERR\_LOCAL\_PORT\_1을 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_2**

I/O 제품의 Modbus/TCP 접속 방법이 [수동 접속]이고 저장하려는 제품 로컬포트 번호가 Modbus/TCP [제품 로컬포트]로 이미 사용 중인 경우 EZTCP\_ERR\_LOCAL\_PORT\_2를 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_3**

I/O 제품 [웹(HTTP)] 제어방식 기능이 선택되어있을 때 저장하려는 제품 로컬포트 번호가 I/O 제품 [웹(HTTP) 포트] 번호로 이미 사용 중인 경우 EZTCP\_ERR\_LOCAL\_PORT\_3를 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_4**

저장하려는 제품 로컬포트 번호가 50005번이면 EZTCP\_ERR\_LOCAL\_PORT\_4을 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_5**

저장하려는 제품 로컬포트 번호가 50006번이면 EZTCP\_ERR\_LOCAL\_PORT\_5을 *error*에 저장합니다.

#### **EZTCP\_ERR\_LOCAL\_PORT\_6**

저장하려는 제품 로컬포트 번호가 50007번이면 EZTCP\_ERR\_LOCAL\_PORT\_6을 *error*에 저장합니다.

#### **EZTCP\_ERR\_IO\_SCRIPT**

I/O 제품에 문법에 맞지 않은 매크로를 저장하려는 경우 EZTCP\_ERR\_IO\_SCRIPT을 *error*에 저장합니다.

#### **EZTCP\_ERR\_IO\_ADDR**

I/O 제품의 [입력포트 주소] 또는 [출력포트 주소]에 유효하지 않은 값을 저장하려는 경우 EZTCP\_ERR\_IO\_ADDR을 *error*에 저장합니다.

#### **EZTCP\_ERR\_RES**

제품(ezTCP)으로부터 응답을 수신하지 못하면 EZTCP\_ERR\_RES를 *error*에 저장합니다.

#### **EZTCP\_ERR\_PWD**

제품(ezTCP)에 설정된 비밀번호와 입력한 비밀번호가 일치하지 않으면 EZTCP\_ERR\_PWD를 *error*에 저장합니다.

#### **EZTCP\_ERR\_PRODUCT\_MISMATCH**

제품종류가 일치하지 않는 경우 EZTCP\_ERR\_PRODUCT\_MISMATCH를 *error*에 저장합니다.

#### **EZTCP\_ERR\_FLASH\_NOSPACE**

제품(ezTCP)에 설정값을 저장할 충분한 메모리 공간이 없는 경우 EZTCP\_ERR\_FLASH\_NOSPACE를 *error*에 저장합니다.

#### **EZTCP\_ERR\_UNKNOWN**

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 EZTCP\_ERR\_UNKNOWN을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    // change some variables by using set_XXXXX functions.

    int err = 0;
    int res = EzTCP_write(mac_address, NULL, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        printf("The requested operation has been successfully completed!\r\n");
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("writing has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
#ifdef __LINUX
            printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
            LPVOID lpMsgBuf;

            FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                (LPSTR)&lpMsgBuf, 0, NULL);

            printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
        }
    }
}
```

**See also**

EzTCP\_Write, EzTCP\_Initialize.

**Remarks****6.2 EzTCP\_Initialize**

```
int EzTCP_Initialize(unsigned char *mac_address, char *password, int udp_port_number,
                    int *error);
```

**Description**

제품(ezTCP)의 MAC주소를 사용하여 제품(ezTCP)의 설정 값을 초기화합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*password*

제품(ezTCP)에 비밀번호가 설정된 경우 현재 제품(ezTCP)에 설정되어있는 비밀번호를 입력해야 합니다.

*int udp\_port\_number*

*EzTCP\_Search* 또는 *EzTCP\_Read*로 제품(ezTCP)을 검색할 때 사용한 UDP 포트 번호 입니다. 0을 입력하면 기본값인 50005번, 50007번을 사용합니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

**Return values**

*EzTCP\_Initialize*의 반환 값은 다음과 같습니다.

**EZTCP\_SUCCESS**

제품(ezTCP)의 설정 값을 성공적으로 초기화하면 EZTCP\_SUCCESS를 반환합니다.

**EZTCP\_ERR**

*EzTCP\_Initialize* 실행 중 오류가 발생한 경우 EZTCP\_ERR를 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.

**EZTCP\_ERR\_NO\_INFO**

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 EZTCP\_ERR\_NO\_INFO를 *error*에 저장합니다.

**EZTCP\_ERR\_PWD\_LENGTH**

*password*에 입력한 비밀번호의 문자열 길이가 유효하지 않으면 EZTCP\_ERR\_PWD\_LENGTH를 *error*에 저장합니다. 비밀번호는 4 ~ 8자입니다.

#### **EZTCP\_ERR\_RES**

제품(ezTCP)으로부터 응답을 수신하지 못하면 EZTCP\_ERR\_RES를 *error*에 저장합니다.

#### **EZTCP\_ERR\_PWD**

제품(ezTCP)에 설정된 비밀번호와 입력한 비밀번호가 일치하지 않으면 EZTCP\_ERR\_PWD를 *error*에 저장합니다.

#### **EZTCP\_ERR\_PRODUCT\_MISMATCH**

제품종류가 일치하지 않는 경우 EZTCP\_ERR\_PRODUCT\_MISMATCH를 *error*에 저장합니다.

#### **EZTCP\_ERR\_FLASH\_NOSPACE**

제품(ezTCP)에 설정값을 저장할 충분한 메모리 공간이 없는 경우 EZTCP\_ERR\_FLASH\_NOSPACE를 *error*에 저장합니다.

#### **EZTCP\_ERR\_UNKNOWN**

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 EZTCP\_ERR\_UNKNOWN을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

### **Examples**

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    unsigned char *mac_address = out_eztcp_info.mac_address;

    int err = 0;
    int res = EzTCP_Initialize(mac_address, NULL, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        printf("The requested operation has been successfully completed!\r\n");
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("Initializing has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
```

```
        {
#ifdef __LINUX
            printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
            LPVOID lpMsgBuf;

            FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                (LPSTR)&lpMsgBuf, 0, NULL);

            printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
        }
    }
}
```

### See also

EzTCP\_Write, EzTCP\_Initialize.

### Remarks

## 7 비밀번호 관리

### 7.1 EzTCP\_ChangePassword

```
int EzTCP_ChangePassword(unsigned char *mac_address, char *current_pwd,  
                          char *new_pwd, int udp_port_number, int *error);
```

#### Description

제품(ezTCP)에 비밀번호를 설정, 변경 또는 삭제할 수 있습니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*current\_pwd*

제품(ezTCP)에 비밀번호가 설정된 경우 현재 제품(ezTCP)에 설정되어있는 비밀번호를 입력해야 합니다. *current\_pwd*는 제품(ezTCP)의 비밀번호를 변경 또는 삭제할 때 사용됩니다.

현재 제품(ezTCP)에 비밀번호가 설정되어 있지 않으면 *current\_pwd*는 NULL이거나 길이가 0이어야 합니다.

*char \*new\_pwd*

제품(ezTCP)에 저장할 비밀번호를 입력해야 합니다.

*new\_pwd*는 제품(ezTCP)에 비밀번호를 처음 설정하거나 이미 설정된 비밀번호를 변경할 때 사용됩니다.

제품(ezTCP)에 설정된 비밀번호를 삭제할 때에는 *new\_pwd*는 NULL이거나 길이가 0이어야 합니다.

*int udp\_port\_number*

*EzTCP\_Search* 또는 *EzTCP\_Read*로 제품(ezTCP)을 검색할 때 사용한 UDP 포트 번호 입니다. 0을 입력하면 기본값인 50005번, 50007번을 사용합니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

#### Return values

*EzTCP\_ChangePassword*의 반환 값은 다음과 같습니다.

*EZTCP\_SUCCESS*

제품(ezTCP)의 설정 값을 성공적으로 초기화하면 EZTCP\_SUCCESS를 반환합니다.

*EZTCP\_ERR*

*EzTCP\_ChangePassword* 실행 중 오류가 발생한 경우 EZTCP\_ERR를 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.



**EZTCP\_ERR\_NO\_INFO**

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 EZTCP\_ERR\_NO\_INFO를 *error*에 저장합니다.

**EZTCP\_ERR\_PWD\_LENGTH**

*current\_pwd* 또는 *new\_pwd*에 입력한 비밀번호의 문자열 길이가 유효하지 않으면 EZTCP\_ERR\_PWD\_LENGTH를 *error*에 저장합니다. 비밀번호는 4 ~ 8자입니다.

**EZTCP\_ERR\_RES**

제품(ezTCP)으로부터 응답을 수신하지 못하면 EZTCP\_ERR\_RES를 *error*에 저장합니다.

**EZTCP\_ERR\_PWD**

제품(ezTCP)에 설정된 비밀번호와 입력한 비밀번호가 일치하지 않으면 EZTCP\_ERR\_PWD를 *error*에 저장합니다.

**EZTCP\_ERR\_PRODUCT\_MISMATCH**

제품종류가 일치하지 않는 경우 EZTCP\_ERR\_PRODUCT\_MISMATCH를 *error*에 저장합니다.

**EZTCP\_ERR\_FLASH\_NOSPACE**

제품(ezTCP)에 설정값을 저장할 충분한 메모리 공간이 없는 경우 EZTCP\_ERR\_FLASH\_NOSPACE를 *error*에 저장합니다.

**EZTCP\_ERR\_UNKNOWN**

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 EZTCP\_ERR\_UNKNOWN을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

**Examples**

```
//-----
// example of setting new password
//
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char *new_pwd = "bbbb";
    unsigned char *mac_address = out_eztcp_info.mac_address;
    int err = 0;

    int res = EzTCP_ChangePassword(mac_address, NULL, new_pwd,
    out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
```

```

    {
        printf("The requested operation has been successfully completed!\r\n");
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("Changing the password has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
#ifdef __LINUX
            printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
            LPVOID lpMsgBuf;

            FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                (LPSTR)&lpMsgBuf, 0, NULL);

            printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
        }
    }
}

//-----

//-----
// example of changing current password
//
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char *cur_pwd = "bbbb"
    char *new_pwd = "aaaa";
    unsigned char *mac_address = out_eztcp_info.mac_address;
    int err = 0;

    int res = EzTCP_ChangePassword(mac_address, cur_pwd, new_pwd,
        out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {

```

```

        printf("The requested operation has been successfully completed!\r\n");
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("Changing the password has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
#ifdef __LINUX
            printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
            LPVOID lpMsgBuf;

            FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                (LPSTR)&lpMsgBuf, 0, NULL);

            printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
        }
    }
}

//-----

//-----
// example of deleting current password
//
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char *cur_pwd = "aaaa"
    unsigned char *mac_address = out_eztcp_info.mac_address;
    int err = 0;

    int res = EzTCP_ChangePassword(mac_address, cur_pwd, NULL,
        out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        printf("The requested operation has been successfully completed!\r\n");
    }
}

```

```

else if(res == EZTCP_ERR)
{
    if(err < 0)
    {
        printf("Changing the password has failed.[Error : %d]\r\n", err);
    }
    else if(err > 0)
    {
#ifdef __LINUX
        printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
        LPVOID lpMsgBuf;
        FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
            FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
            (LPSTR)&lpMsgBuf, 0, NULL);
        printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
    }
}

//-----

```

## See also

## Remarks



## 8 현재상태 보기

### 8.1 EzTCP\_Status

```
int EzTCP_Status(unsigned char *mac_address, int udp_port_number, int *error);
```

#### Description

제품(ezTCP) 상태정보를 읽을 수 있습니다. 상태정보는 문자열로 되어있으며 그 길이는 제품의 종류에 따라서 다를 수 있습니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int udp\_port\_number*

*EzTCP\_Search* 또는 *EzTCP\_Read*로 제품(ezTCP)을 검색할 때 사용한 UDP 포트 번호 입니다. 0을 입력하면 기본값인 50005번, 50007번을 사용합니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

#### Return values

*EzTCP\_Status*의 반환 값은 다음과 같습니다.

##### *EZTCP\_SUCCESS*

제품(ezTCP)의 상태정보를 성공적으로 읽으면 EZTCP\_SUCCESS를 반환합니다.

##### *EZTCP\_ERR*

*EzTCP\_Status*실행 중 오류가 발생한 경우 EZTCP\_ERR을 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.

##### *EZTCP\_ERR\_NO\_INFO*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 EZTCP\_ERR\_NO\_INFO를 *error*에 저장합니다.

##### *EZTCP\_ERR\_RES*

제품(ezTCP)으로부터 응답을 수신하지 못하면 EZTCP\_ERR\_RES를 *error*에 저장합니다.

##### *EZTCP\_ERR\_UNKNOWN*

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 EZTCP\_ERR\_UNKNOWN을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

## Examples

### See also

EzTCP\_Status,      get\_status\_string\_length,      get\_status\_string,      get\_tcpip4\_session\_count,  
 get\_tcpip6\_session\_count,      get\_tcpip4\_session\_info,      get\_tcpip6\_session\_info,      EzTCP\_CloseTcpIp4,  
 EzTCP\_CloseTcpIp6.

### Remarks

## 8.2 get\_status\_string\_length

```
int get_status_string_length(void);
```

### Description

제품(ezTCP) 상태정보 문자열의 길이를 반환합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

### Return values

*get\_status\_string\_length*는 제품(ezTCP) 상태정보 문자열의 길이를 반환합니다.

## Examples

### See also

EzTCP\_Status,      get\_status\_string\_length,      get\_status\_string,      get\_tcpip4\_session\_count,  
 get\_tcpip6\_session\_count,      get\_tcpip4\_session\_info,      get\_tcpip6\_session\_info,      EzTCP\_CloseTcpIp4,  
 EzTCP\_CloseTcpIp6.

### Remarks

## 8.3 get\_status\_string

```
void get_status_string(char *buf);
```

### Description

제품(ezTCP) 상태정보 문자열을 *buf*에 저장합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

**Parameters***char \*buf*

제품(ezTCP) 상태정보가 저장될 포인터입니다.

**Return values****Examples**

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char *buf;
    int err = 0;
    int status_len = 0;

    int res = EzTCP_Status(out_eztcp_info.mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        status_len = get_status_string_length();
        if(status_len > 0)
        {
            buf = (char*)malloc(status_len + 1);
            memset(buf, 0x00, status_len + 1);
            get_status_string(buf);
            printf("[Status]\r\n%s\r\n", buf);
            free(buf);
        }
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("The requested operation has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
            #if defined(__LINUX)
                printf("Error[%d] %s\r\n", err, strerror(err));
            #elif defined(__WINDOWS)

```

```

        LPVOID lpMsgBuf;

        FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPSTR)&lpMsgBuf, 0, NULL);

        printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);

    #endif

    }

}

```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count, get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4, EzTCP\_CloseTcpIp6.

### Remarks

*buf*는 *get\_status\_string\_length*의 반환 값보다 큰 메모리 공간을 가지고 있어야 합니다.

## 8.4 get\_tcpip4\_session\_count

```
int get_tcpip4_session_count(void);
```

### Description

제품(ezTCP) 상태정보 중에서 제품이 가지고 있는 TCP/IP 버전4 접속 개 수를 반환합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

### Return values

*get\_tcpip4\_session\_count*는 제품(ezTCP)이 가지고 있는 TCP/IP 버전4 접속 개 수를 반환합니다.

### Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;

```



```

int tcpip4_count;

int res = EzTCP_Status(out_eztcp_info.mac_address, out_eztcp_info.search_port, &err);

if(res == EZTCP_SUCCESS)
{
    tcpip4_count = get_tcpip4_session_count();
    for(idx = 0; idx < tcpip4_count; idx++)
    {
        if(get_tcpip4_session_info(idx, buf) == 1)
        {
            memset(buf, 0x00, 256);
            printf("[TCP/IP4] SESSION #%d\r\n%s\r\n", idx, buf);
        }
    }
}
else if(res == EZTCP_ERR)
{
    if(err < 0)
    {
        printf("The requested operation has failed.[Error : %d]\r\n", err);
    }
    else if(err > 0)
    {
#ifdef __LINUX
        printf("Error[%d] %s\r\n", err, strerror(err));
#elif defined(__WINDOWS)
        LPVOID lpMsgBuf;

        FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
            FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
            (LPSTR)&lpMsgBuf, 0, NULL);

        printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
#endif
    }
}
}

```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count,  
 get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4,  
 EzTCP\_CloseTcpIp6.

### Remarks



## 8.5 get\_tcpip6\_session\_count

```
int get_tcpip6_session_count(void);
```

### Description

제품(ezTCP) 상태정보 중에서 제품이 가지고 있는 TCP/IP 버전6 접속 개 수를 반환합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

### Return values

*get\_tcpip6\_session\_count*는 제품(ezTCP)이 가지고 있는 TCP/IP 버전6 접속 개 수를 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;
    int tcpip6_count;

    int res = EzTCP_Status(out_eztcp_info.mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        tcpip6_count = get_tcpip6_session_count();

        for(idx = 0; idx < tcpip6_count; idx++)
        {
            if(get_tcpip6_session_info(idx, buf) == 1)
            {
                memset(buf, 0x00, 256);
                printf("[TCP/IP6] SESSION #%d\r\n%s\r\n", idx, buf);
            }
        }
    }
    else if(res == EZTCP_ERR)
```

```

    {
        if(err < 0)
        {
            printf("The requested operation has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
            #if defined(__LINUX)
                printf("Error[%d] %s\r\n", err, strerror(err));
            #elif defined(__WINDOWS)
                LPVOID lpMsgBuf;

                FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                    FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                    (LPSTR)&lpMsgBuf, 0, NULL);

                printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
            #endif
        }
    }
}

```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count,  
get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4,  
EzTCP\_CloseTcpIp6.

### Remarks

## 8.6 get\_tcpip4\_session\_info

```
int get_tcpip4_session_info(int session_number, char *out_session_info);
```

### Description

제품(ezTCP) 상태정보 중에서 *session\_number*가 가리키는 TCP/IP 버전4 접속정보를 *out\_session\_info*에 저장합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

*int session\_number*

TCP/IP 버전4 접속정보 순번으로 0부터 시작하며 *get\_tcpip4\_session\_count*의 반환 값보다 크거나 같을 수 없습니다.

*char \*out\_session\_info*

TCP/IP 버전4 접속정보가 저장될 포인터입니다. *out\_session\_info*의 크기는 256바이트 이

상이 되어야합니다.

## Return values

*get\_tcpip4\_session\_info*는 *out\_session\_info*에 TCP/IP4 버전4 접속정보를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;
    int tcpip4_count;

    int res = EzTCP_Status(out_eztcp_info.mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        tcpip4_count = get_tcpip4_session_count();
        for(idx = 0; idx < tcpip4_count; idx++)
        {
            if(get_tcpip4_session_info(idx, buf) == 1)
            {
                memset(buf, 0x00, 256);
                printf("[TCP/IP4] SESSION %d\r\n%s\r\n", idx, buf);
            }
        }
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("The requested operation has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
            #if defined(__LINUX)
                printf("Error[%d] %s\r\n", err, strerror(err));
            #endif
        }
    }
}
```

```

#elif defined(__WINDOWS)
    LPVOID lpMsgBuf;

    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPSTR)&lpMsgBuf, 0, NULL);

    printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);

#endif
    }
}
}

```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count, get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4, EzTCP\_CloseTcpIp6.

### Remarks

*out\_session\_info*의 크기는 256바이트 이상이 되어야합니다.

## 8.7 get\_tcpip6\_session\_info

```
int get_tcpip6_session_info(int session_number, char *out_session_info);
```

### Description

제품(ezTCP) 상태정보 중에서 *session\_number*가 가리키는 TCP/IP 버전6 접속정보를 *out\_session\_info*에 저장합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

*int session\_number*

TCP/IP 버전6 접속정보 순번으로 0부터 시작하며 *get\_tcpip6\_session\_count*의 반환 값보다 크거나 같을 수 없습니다.

*char \*out\_session\_info*

TCP/IP 버전6 접속정보가 저장될 포인터입니다. *out\_session\_info*의 크기는 256바이트 이상이 되어야합니다.

### Return values

*get\_tcpip6\_session\_info*는 *out\_session\_info*에 TCP/IP4 버전6 접속정보를 저장하는데 성공하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

### Examples

```
struct eztcp_info out_eztcp_info;
```



```

int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;
    int tcpip6_count;

    int res = EzTCP_Status(out_eztcp_info.mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        tcpip6_count = get_tcpip6_session_count();

        for(idx = 0; idx < tcpip6_count; idx++)
        {
            if(get_tcpip6_session_info(idx, buf) == 1)
            {
                memset(buf, 0x00, 256);
                printf("[TCP/IP6] SESSION #d\r\n%s\r\n", idx, buf);
            }
        }
    }
    else if(res == EZTCP_ERR)
    {
        if(err < 0)
        {
            printf("The requested operation has failed.[Error : %d]\r\n", err);
        }
        else if(err > 0)
        {
            #if defined(__LINUX)
                printf("Error[%d] %s\r\n", err, strerror(err));
            #elif defined(__WINDOWS)
                LPVOID lpMsgBuf;

                FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
                    FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err, MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
                    (LPSTR)&lpMsgBuf, 0, NULL);

                printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
            #endif
        }
    }
}

```

```
}
}
```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count,  
get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4,  
EzTCP\_CloseTcpIp6.

### Remarks

*out\_session\_info*의 크기는 256바이트 이상이 되어야합니다.

## 8.8 EzTCP\_CloseTcpIp4

```
int EzTCP_CloseTcpIp4(unsigned char *mac_address, int session_number,  
                      char *password, int *error);
```

### Description

제품(ezTCP)의 TCP/IP 버전4 접속 중에서 *session\_number*가 가리키는 접속의 상태가 접속완료인 경우 그 접속상태를 접속종료로 변경합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용하십시오.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int session\_number*

TCP/IP 버전4 접속정보 순번으로 0부터 시작하며 *get\_tcpip4\_session\_count*의 반환 값보다 크거나 같을 수 없습니다.

*char \*password*

제품(ezTCP)에 비밀번호가 설정된 경우 현재 설정되어있는 비밀번호가 저장된 포인터입니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

### Return values

*EzTCP\_CloseTcpIp4*의 반환 값은 다음과 같습니다.

*EZTCP\_SUCCESS*

제품(ezTCP)에 설정 값을 성공적으로 저장하면 *EZTCP\_SUCCESS*를 반환합니다.

*EZTCP\_ERR*



*EzTCP\_CloseTcpIp4* 실행 중 오류가 발생한 경우 EZTCP\_ERR을 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.

#### *EZTCP\_ERR\_NO\_INFO*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 EZTCP\_ERR\_NO\_INFO를 *error*에 저장합니다.

#### *EZTCP\_ERR\_PWD\_LENGTH*

*password*에 입력한 비밀번호의 문자열 길이가 유효하지 않으면 EZTCP\_ERR\_PWD\_LENGTH를 *error*에 저장합니다. 비밀번호는 4 ~ 8자입니다.

#### *EZTCP\_ERR\_RES*

제품(ezTCP)으로부터 응답을 수신하지 못하면 EZTCP\_ERR\_RES를 *error*에 저장합니다.

#### *EZTCP\_ERR\_PWD*

제품(ezTCP)에 설정된 비밀번호와 입력한 비밀번호가 일치하지 않으면 EZTCP\_ERR\_PWD를 *error*에 저장합니다.

#### *EZTCP\_ERR\_UNKNOWN*

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 EZTCP\_ERR\_UNKNOWN을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

## Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;
    int tcpip4_count;
    unsigned char *mac_address = out_eztcp_info.mac_address;

    int res = EzTCP_Status(mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        tcpip4_count = get_tcpip4_session_count();

        err = 0;

        res = EzTCP_CloseTcpIp4(mac_address, 0, NULL, &err);
    }
}
```



```

        if(res == EZTCP_SUCCESS)
        {
            printf("Success.\r\n");
        }
        else if(res == EZTCP_ERR)
        {
            if(err < 0)
            {
                printf("The requested operation has failed.[Error : %d]\r\n", err);
            }
            else if(err > 0)
            {
                #if defined(__LINUX)
                    printf("Error[%d] %s\r\n", err, strerror(err));
                #elif defined(__WINDOWS)
                    LPVOID lpMsgBuf;

                    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER |
                        FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err,
                        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), (LPSTR)&lpMsgBuf, 0, NULL);
                    printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
                #endif
            }
        }
    }
}

```

### See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count,  
get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4,  
EzTCP\_CloseTcpIp6.

### Remarks

## 8.9 EzTCP\_CloseTcpIp6

```

int EzTCP_CloseTcpIp6(unsigned char *mac_address, int session_number,
    char *password, int *error);

```

### Description

제품(ezTCP)의 TCP/IP 버전6 접속 중에서 *session\_number*가 가리키는 접속의 상태가 접속완료인 경우 그 접속상태를 접속종료로 변경합니다. *EzTCP\_Status* 실행이 성공한 경우에 사용

하십시오.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int session\_number*

TCP/IP 버전6 접속정보 순번으로 0부터 시작하며 *get\_tcpip6\_session\_count*의 반환 값보다 크거나 같을 수 없습니다.

*char \*password*

제품(ezTCP)에 비밀번호가 설정된 경우 현재 설정되어있는 비밀번호가 저장된 포인터입니다.

*int \*error*

함수 실행이 실패한 경우 오류 값이 저장됩니다.

## Return values

*EzTCP\_CloseTcpIp6*의 반환 값은 다음과 같습니다.

### *EZTCP\_SUCCESS*

제품(ezTCP)에 설정 값을 성공적으로 저장하면 *EZTCP\_SUCCESS*를 반환합니다.

### *EZTCP\_ERR*

*EzTCP\_CloseTcpIp6*실행 중 오류가 발생한 경우 *EZTCP\_ERR*를 반환합니다. 이 경우 *error*에 오류 번호를 저장합니다.

오류 번호가 0보다 작은 경우에는 아래와 같은 값을 가질 수 있습니다.

#### *EZTCP\_ERR\_NO\_INFO*

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 *EZTCP\_ERR\_NO\_INFO*를 *error*에 저장합니다.

#### *EZTCP\_ERR\_PWD\_LENGTH*

*password*에 입력한 비밀번호의 문자열 길이가 유효하지 않으면 *EZTCP\_ERR\_PWD\_LENGTH*를 *error*에 저장합니다. 비밀번호는 4 ~ 8자입니다.

#### *EZTCP\_ERR\_RES*

제품(ezTCP)으로부터 응답을 수신하지 못하면 *EZTCP\_ERR\_RES*를 *error*에 저장합니다.

#### *EZTCP\_ERR\_PWD*

제품(ezTCP)에 설정된 비밀번호와 입력한 비밀번호가 일치하지 않으면 *EZTCP\_ERR\_PWD*를 *error*에 저장합니다.

#### *EZTCP\_ERR\_UNKNOWN*

제품(ezTCP)으로부터 알 수 없는 오류번호를 수신한 경우 *EZTCP\_ERR\_UNKNOWN*을 *error*에 저장합니다.

오류번호가 0보다 큰 경우에는 함수 실행 중 사용자 시스템에서 발생한 오류번호입니다.

## Examples

```

struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[256];
    int err = 0;
    int idx;
    int tcpip6_count;
    unsigned char *mac_address = out_eztcp_info.mac_address;

    int res = EzTCP_Status(mac_address, out_eztcp_info.search_port, &err);

    if(res == EZTCP_SUCCESS)
    {
        tcpip6_count = get_tcpip6_session_count();

        err = 0;

        res = EzTCP_CloseTcpIp6(mac_address, 0, NULL, &err);
        if(res == EZTCP_SUCCESS)
        {
            printf("Success.\r\n");
        }
        else if(res == EZTCP_ERR)
        {
            if(err < 0)
            {
                printf("The requested operation has failed.[Error : %d]\r\n", err);
            }
            else if(err > 0)
            {
                #if defined(__LINUX)
                    printf("Error[%d] %s\r\n", err, strerror(err));
                #elif defined(__WINDOWS)
                    LPVOID lpMsgBuf;
                    FormatMessage(FORMAT_MESSAGE_ALLOCATE_BUFFER |
                        FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_IGNORE_INSERTS, NULL, err,
                        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), (LPSTR)&lpMsgBuf, 0, NULL);
                    printf("Error[%d] %s\r\n", err, (char*)lpMsgBuf);
                #endif
            }
        }
    }
}

```

```

    }
}
}
}

```

## See also

EzTCP\_Status, get\_status\_string\_length, get\_status\_string, get\_tcpip4\_session\_count, get\_tcpip6\_session\_count, get\_tcpip4\_session\_info, get\_tcpip6\_session\_info, EzTCP\_CloseTcpIp4, EzTCP\_CloseTcpIp6.

## Remarks

## 9 기타

### 9.1 get\_version

```
int get_version(unsigned char *mac_address, char *out_version);
```

#### Description

*get\_version*은 제품(ezTCP)의 펌웨어 버전을 반환합니다.

#### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*char \*out\_version*

펌웨어 버전이 저장될 포인터입니다.

#### Return values

*get\_version*은 *out\_version*에 제품(ezTCP)의 펌웨어 버전을 저장하는데 성공하면 1을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

#### Examples

```
struct eztcp_info out_eztcp_info;
int ret = get_eztcp_info_by_index(0, &out_eztcp_info);

if(ret == EZTCP_SUCCESS)
{
    char buf[8];
    memset(buf, 0x00, 8);
    if(get_version(out_eztcp_info.mac_address, buf) == 1)
    {
        printf("Firmware version : %s\r\n", buf);
    }
}
```

#### See also

#### Remarks

펌웨어 버전은 1.1A와 같이 숫자로 된 주 번호 1자리, 부 번호 1자리 그리고 영문으로 된 개정번호 1자리를 사용합니다.

## 9.2 is\_ip6

```
int is_ip6(unsigned char *mac_address);
```

### Description

*is\_ip6*는 제품(ezTCP)이 TCP/IPv6 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_ip6*는 제품(ezTCP)이 TCP/IPv6 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_ip6*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.3 is\_uart\_rs232

```
int is_uart_rs232(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_rs232*는 *uart\_index*로 지정된 시리얼 포트가 [시리얼 종류]로 RS-232를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values



*is\_uart\_rs232*는 제품(ezTCP)이 RS-232를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

## Remarks

## 9.4 is\_uart\_rs485

```
int is_uart_rs485(unsigned char *mac_address, int uart_index);
```

## Description

*is\_uart\_rs485*는 *uart\_index*로 지정된 시리얼 포트가 [시리얼 종류]로 RS-485를 사용할 수 있는지 확인합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

## Return values

*is\_uart\_rs485*는 *uart\_index*로 지정된 시리얼 포트가 RS-485를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

## Remarks

*is\_rs485*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.5 is\_uart\_rs422

```
int is_uart_rs422(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_rs422*는 *uart\_index*로 지정된 시리얼 포트가 [시리얼 종류]로 RS-422을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_rs422*는 *uart\_index*로 지정된 시리얼 포트가 RS-422를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_rs422*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.6 is\_uart\_ttl

```
int is_uart_ttl(unsigned char *mac_address);
```

### Description

*is\_uart\_ttl*은 제품(ezTCP) 시리얼 포트가 [TTL] 출력 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values



*is\_uart\_ttl*은 제품(ezTCP) 시리얼 포트가 [TTL] 출력 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

## Remarks

## 9.7 is\_uart\_8\_databit\_only

```
int is_uart_8_databit_only(unsigned char *mac_address);
```

## Description

*is\_uart\_8\_databit\_only*는 제품(ezTCP) 시리얼 포트가 [데이터 비트]로 8비트만 사용 가능한지 확인합니다.

## Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

## Return values

*is\_uart\_8\_databit\_only*는 제품(ezTCP)이 시리얼 포트의 데이터 비트로 8비트만 사용 가능할 경우 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

## Examples

## See also

## Remarks

## 9.8 is\_uart\_7\_and\_8\_databit\_only

```
int is_uart_7_and_8_databit_only(unsigned char *mac_address);
```

## Description



*is\_uart\_7\_and\_8\_databit\_only*는 제품(ezTCP)이 시리얼 포트의 [데이터 비트]로 7비트와 8비트만 사용 가능한지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_uart\_7\_and\_8\_databit\_only*는 제품(ezTCP)이 시리얼 포트의 [데이터 비트]로 7비트와 8비트만 사용 가능할 경우 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.9 is\_uart\_one5\_stopbit

```
int is_uart_one5_stopbit(unsigned char *mac_address);
```

### Description

*is\_uart\_one5\_stopbit*는 제품(ezTCP) 시리얼 포트가 [정지 비트]로 1.5비트를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_uart\_one5\_stopbit*는 제품(ezTCP)이 시리얼 포트의 [정지 비트]로 1.5비트를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

**Remarks****9.10 is\_uart\_dtrdsr**

```
int is_uart_dtrdsr(unsigned char *mac_address);
```

**Description**

*is\_uart\_dtrdsr*은 제품(ezTCP) 시리얼 포트가 **[DTR/DSR]** 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_dtrdsr*은 제품(ezTCP) 시리얼 포트가 **[DTR/DSR]** 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.11 is\_uart\_xonxoff**

```
int is_uart_xonxoff(unsigned char *mac_address);
```

**Description**

*is\_uart\_xonxoff*는 제품(ezTCP) 시리얼 포트가 **[흐름 제어]**로 XON/XOFF를 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_xonxoff*는 제품(ezTCP) 시리얼 포트가 [흐름 제어]로 XON/XOFF를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.12 is\_uart\_tx\_delay**

```
int is_uart_tx_delay(unsigned char *mac_address);
```

**Description**

*is\_uart\_tx\_delay*는 제품(ezTCP) 시리얼 포트가 [데이터 전송 간격] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_tx\_delay*는 제품(ezTCP) 시리얼 포트가 [데이터 전송 간격]을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.13 is\_uart\_data\_frame\_interval**

```
int is_uart_data_frame_interval(unsigned char *mac_address);
```

**Description**

*is\_uart\_data\_frame\_interval*은 제품(ezTCP) 시리얼 포트가 [데이터 프레임 간격(10ms)] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_data\_frame\_interval*은 제품(ezTCP) 시리얼 포트가 [데이터 프레임 간격(10ms)] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.14 is\_uart\_separator**

```
int is_uart_separator(unsigned char *mac_address);
```

**Description**

*is\_uart\_separator*는 제품(ezTCP) 시리얼 포트가 구분자 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_separator*는 제품(ezTCP) 시리얼 포트가 구분자 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also**

**Remarks****9.15 is\_uart\_tcp\_nodelay**

```
int is_uart_tcp_nodelay(unsigned char *mac_address, int uart_index);
```

**Description**

*is\_uart\_tcp\_nodelay*는 제품(ezTCP) 시리얼 포트가 [전송지연 기능 사용안함] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_uart\_tcp\_nodelay*는 제품(ezTCP) 시리얼 포트가 [전송지연 기능 사용안함] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks**

[전송지연 기능 사용안함] 기능의 사용 가능 여부는 다른 설정 값에 따라 달라질 수 있습니다.

**9.16 is\_uart\_rfc2217**

```
int is_uart_rfc2217(unsigned char *mac_address, int uart_index);
```

**Description**

*is\_uart\_rfc2217*은 제품(ezTCP) 시리얼 포트가 [시리얼 포트 설정/상태 전송(RFC2217)] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_uart\_rfc2217*은 제품(ezTCP) 시리얼 포트가 [시리얼 포트 설정/상태 전송(RFC2217)] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

[시리얼 포트 설정/상태 전송(RFC2217)] 기능의 사용 가능 여부는 다른 설정 값에 따라 달라질 수 있습니다.

## 9.17 is\_uart\_protocol

```
int is_uart_protocol(unsigned char *mac_address);
```

### Description

*is\_uart\_protocol*은 제품(ezTCP) 시리얼 포트가 [프로토콜] 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_uart\_protocol*은 제품(ezTCP) 시리얼 포트가 [프로토콜] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

현재 CSE-T16, CSE-T32, CSE-T48만 사용 가능하며 추후 지원가능 모델은 추가 될 수 있습니다. 보다 자세한 내용은 해당 제품 사용설명서를 참조하시기 바랍니다.

## 9.18 is\_uart\_tcp\_server

```
int is_uart_tcp_server(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_tcp\_server*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 T2S - TCP 서버를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_tcp\_server*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 T2S - TCP 서버를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_uart\_tcp\_server*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.19 is\_uart\_at\_command

```
int is_uart_at_command(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_at\_command*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 ATC - AT 명령을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*



제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_at\_command*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 ATC – AT 명령을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_uart\_at\_command*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.20 is\_uart\_tcp\_client

```
int is_uart_tcp_client(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_tcp\_client*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 COD – TCP 클라이언트를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_tcp\_client*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 COD – TCP 클라이언트를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_uart\_tcp\_client*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.21 is\_uart\_udp

```
int is_uart_udp(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_udp*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 U2S – UDP를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_udp*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 U2S – UDP를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_uart\_udp*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.22 is\_uart\_serial\_modbus

```
int is_uart_serial_modbus(unsigned char *mac_address, int uart_index);
```

### Description

*is\_uart\_serial\_modbus*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 시리얼 Modbus/TCP 를 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

*int uart\_index*

제품(ezTCP)의 시리얼 포트 순번으로 0부터 시작합니다.

### Return values

*is\_uart\_serial\_modbus*는 *uart\_index*로 지정된 시리얼 포트가 [통신모드]로 시리얼 Modbus/TCP를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_uart\_serial\_modbus*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.23 is\_csc\_hr2

```
int is_csc_hr2(unsigned char *mac_address);
```

### Description

*is\_csc\_hr2*는 *mac\_address*가 가리키는 제품(ezTCP)이 CSC-HR2인지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_csc\_hr2*는 *mac\_address*가 가리키는 제품(ezTCP)이 CSC-HR2이면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.24 is\_io

```
int is_io(unsigned char *mac_address);
```

### Description

*is\_io*는 *mac\_address*가 가리키는 제품(ezTCP)이 디지털 입력과 출력 포트를 지원하는 제품인지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_io*는 *mac\_address*가 가리키는 제품(ezTCP)이 디지털 입력과 출력 포트를 지원하면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.25 is\_io\_output\_automatic\_initialize

```
int is_io_output_automatic_initialize(unsigned char *mac_address);
```

### Description

*is\_io\_output\_automatic\_initialize*는 I/O 제품이 [출력포트 상태 초기화(Modbus/TCP 접속 종료 시 출력포트[초기상태] 값으로 상태 변경) 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_io\_output\_automatic\_initialize*는 제품(ezTCP)이 [출력포트 상태 초기화(Modbus/TCP 접속 종료 시 출력포트[초기상태] 값으로 상태 변경) 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.26 is\_event\_notification

```
int is_event_notification(unsigned char *mac_address);
```

### Description

*is\_event\_notification*은 I/O 제품이 [입력 또는 출력포트 변경 알림(전자메일)] 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_event\_notification*은 제품(ezTCP)이 [입력 또는 출력포트 변경 알림(전자메일)] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.27 is\_wlan

```
int is_wlan(unsigned char *mac_address);
```

### Description

*is\_wlan*은 *mac\_address*가 가리키는 제품(ezTCP)이 무선랜을 지원하는 제품인지 확인합니다.

**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.**Return values***is\_wlna*은 *mac\_address*가 가리키는 제품(ezTCP)이 무선랜을 지원하면 1을 반환하고 그렇지 않으면 0을 반환합니다.*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.**Examples****See also****Remarks****9.28 is\_wlan\_soft\_ap**

```
int is_wlan_soft_ap(unsigned char *mac_address);
```

**Description***is\_wlan\_soft\_ap*는 무선랜 제품이 [무선랜 종류]로 **Soft AP**를 사용할 수 있는지 확인합니다.**Parameters***unsigned char \*mac\_address**EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.**Return values***is\_wlan\_soft\_ap*는 무선랜 제품이 [무선랜 종류]로 **Soft AP**를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.**Examples****See also****Remarks**

## 9.29 is\_wlan\_antenna

```
int is_wlan_antenna(unsigned char *mac_address);
```

### Description

*is\_wlan\_antenna*는 무선랜 제품이 [안테나] 설정 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_wlan\_antenna*는 무선랜 제품이 [안테나] 설정 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.30 is\_wlan\_phy\_mode

```
int is_wlan_phy_mode(unsigned char *mac_address);
```

### Description

*is\_wlan\_phy\_mode*는 무선랜 제품이 [무선 고급설정]을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_wlan\_phy\_mode*는 무선랜 제품이 [무선 고급설정]을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.31 is\_wlan\_background\_scan**

```
int is_wlan_background_scan(unsigned char *mac_address);
```

**Description**

*is\_wlan\_background\_scan*은 무선랜 제품이 [무선 고급설정]들 중에서 [Background Scan]을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_wlan\_background\_scan*은 무선랜 제품이 [무선 고급설정]들 중에서 [Background Scan]을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.32 is\_wlan\_rsn**

```
int is_wlan_rsn(unsigned char *mac_address);
```

**Description**

*is\_wlan\_rsn*은 무선랜 제품이 **RSN(Robust Security Network)**을 사용할 수 있는지 확인합니다.

**Parameters**



*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_wlan\_rsn*은 무선랜 제품이 **RSN(Robust Security Network)**을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

RSN을 지원하는 무선랜 제품은 무선랜에 접속할 때 암호화 방식과 인증방식을 자동으로 선택합니다. 그러므로 RSN을 지원하는 제품과 그렇지 않은 제품은 암호화 방식과 인증방식에 관련된 설정 값을 읽고 변경하는 함수들이 따로 제공됩니다.

## 9.33 is\_wlan\_authentication\_type

```
int is_wlan_authentication_type(unsigned char *mac_address);
```

### Description

*is\_wlan\_authentication\_type*은 **RSN을 지원하지 않는** 무선랜 제품이 사용 가능한 **WPA [인증 방식/암호화 방법]**을 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_wlan\_authentication\_type*이 0을 반환하는 무선랜 제품은 **WPA [인증 방식/암호화 방법]**으로 WPA PSK – TKIP 와 WPA2 PSK – AES만 사용할 수 있습니다.

*is\_wlan\_authentication\_type*이 1을 반환하는 무선랜 제품은 **WPA [인증 방식/암호화 방법]**으로 WPA PSK – TKIP, WPA PSK – AES, WPA PSK – TKIP/AES, WPA2 PSK – TKIP, WPA2 PSK – AES, WPA2 PSK – TKIP/AES를 사용할 수 있습니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples



**See also****Remarks****9.34 is\_wlan\_wpa\_enterprise**

```
int is_wlan_wpa_enterprise(unsigned char *mac_address);
```

**Description**

*is\_wlan\_wpa\_enterprise*는 무선랜 제품이 보안 설정 중에서 **[802.1X]**를 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_wlan\_wpa\_enterprise*는 무선랜 제품이 보안 설정 중에서 **[802.1X]**를 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.35 is\_send\_mac\_address**

```
int is_send_mac_address(unsigned char *mac_address);
```

**Description**

*is\_send\_mac\_address*는 제품(ezTCP)이 **[MAC 주소 전송]** 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_send\_mac\_address*는 제품(ezTCP)이 [MAC 주소 전송] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_send\_mac\_address*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.36 is\_ssl

```
int is_ssl(unsigned char *mac_address);
```

### Description

*is\_ssl*은 제품(ezTCP)이 [SSL 보안통신]을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_ssl*은 제품(ezTCP)이 [SSL 보안통신]을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_ssl*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.37 is\_ssh

```
int is_ssh(unsigned char *mac_address);
```

### Description

*is\_ssh*는 제품(ezTCP)이 [SSH 보안통신]을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_ssh*는 제품(ezTCP)이 [SSH 보안통신]을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

*is\_ssh*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

## 9.38 is\_remote\_debug

```
int is_remote_debug(unsigned char *mac_address);
```

### Description

*is\_remote\_debug*는 제품(ezTCP)이 [디버깅 로그 보기] 기능을 사용할 수 있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_remote\_debug*는 제품(ezTCP)이 [디버깅 로그 보기] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks****9.39 is\_tcp\_multi\_connection**

```
int is_tcp_multi_connection(unsigned char *mac_address);
```

**Description**

*is\_tcp\_multi\_connection*은 제품(ezTCP)이 [다중 접속] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

**Return values**

*is\_tcp\_multi\_connection*은 제품(ezTCP)이 [다중 접속] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

**Examples****See also****Remarks**

*is\_tcp\_multi\_connection*의 반환값은 다른 설정 값에 따라 달라질 수 있습니다.

**9.40 is\_power\_management**

```
int is_power_management(unsigned char *mac_address);
```

**Description**

*is\_power\_management*는 제품(ezTCP)이 [전원 관리] 기능을 사용할 수 있는지 확인합니다.

**Parameters**

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_power\_management*는 제품(ezTCP)이 [전원 관리] 기능을 사용할 수 있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 9.41 is\_password

```
int is_password(unsigned char *mac_address);
```

### Description

*is\_password*는 비밀번호가 제품(ezTCP)에 설정되어있는지 확인합니다.

### Parameters

*unsigned char \*mac\_address*

*EzTCP\_Search* 또는 *EzTCP\_Read*를 사용하여 검색된 제품(ezTCP)의 MAC주소 입니다.

### Return values

*is\_password*는 제품(ezTCP)에 비밀번호가 설정되어있으면 1을 반환하고 그렇지 않으면 0을 반환합니다.

*mac\_address*을 가진 제품(ezTCP)을 찾을 수 없는 경우에는 -1을 반환합니다.

### Examples

### See also

### Remarks

## 10 면책 고지 사항

### 10.1 면책 고지 사항

솔내시스템(주)과 그 대리점은 CIE-H14의 사용 또는 사용불능에 따른 손해 및 손실, 영업중지로 인한 비용, 정보 손실을 포함한 기타 고지 받은 어떠한 재정적 손해에 대해서도 책임지지 않습니다.

CIE-H14는 허락되지 않는 응용분야에서의 사용을 금지합니다. 허락되지 않은 응용분야라 함은 군사, 핵, 항공, 폭발물, 의학, 방범설비, 화재경보기, 엘리베이터를 수반한 용도 혹은 차량, 항공기, 트럭, 보트, 헬리콥터 및 이에 국한되지 않는 모든 교통수단을 포함합니다.

또한, 고장 및 실패로 인한 재정적 손실 및 기물파손, 신체 상해 혹은 사람이나 동물의 사상을 초래하는 실험, 개발 및 각종 응용분야에 사용할 수 없습니다. 구매자(혹은 업체)가 자발적 혹은 비자발적으로 이러한 허락되지 않는 응용분야에 사용할 시 솔내시스템(주)과 그 대리점에 손해배상을 포함한 어떠한 책임도 묻지 않을 것에 동의한 것으로 간주합니다.

구매한 제품의 환불 및 수리, 교환에 대한 배상 책임과 구매자(혹은 업체)의 단독 구제책은 솔내시스템(주)과 그 대리점의 선택사항입니다.

솔내시스템(주)과 그 대리점은 동반된 기술자료, 하드웨어, 펌웨어를 포함한 CIE-H14의 상업성이나 특정목적에 따른 적합성에 대한 모든 명시적 혹은 묵시적 보증 및 기타 이에 국한되지 않는 여타의 보증을 하지 않습니다.

## 11문서 변경 이력

날짜	버전	변경내용	작성자
2013.10.30	1.0	○ 최초 작성	김형준