SCX__ETHERNET_JGLEE

Header File 경로

"C:₩Module_Linker₩EQP_Link_Res₩Apps_Header₩EQP₩UTILITY₩IEQP_BODY__UTILITY_LINK.h"

Class SCI__ETHERNET_JGLEE

Example)

```
SCX__ETHERNET_JGLEE   x_net;

CString net_ip = "127.0.01";
int net_port = 10001;

x_net->INIT__PROPERTY( net_ip, net_port );
CString err_msg;
x_net->CONNECT( &err_msg );
```

INIT__PROPERTY 함수

EtherNetI 통신을 위한 net_ip, net_port 정보를 입력한다.

Syntax

```
int INIT__PROPERTY( const CString& net_ip, const int net_port );
```

Parameter

net_ip : ip address
Ex) "127.0.0.1"
net_port : Client Type인 경우 연결할 Port 값
Server Type인 경우 Service할 port 값

Return Value

> 0 : Initialize 성공 했을 때
< 0 : Initialize 실패 했을 때

SET__TERMINAL_InSTRING 함수

수신 Terminal Data를 설정한다.

Syntax

int SET__TERMINAL_InSTRING(const char* in_term, const int size);

Parameter

in_term : terminal character array
size : terminal character size

Return Value

> 0 : Initialize 성공 했을 때
< 0 : Initialize 실패 했을 때

SET__TERMINAL_OutSTRING 함수

송신 Terminal Data를 설정한다.

Syntax

int SET__TERMINAL_OutSTRING(const char* in_term, const int size);

Parameter

in_term : terminal character array
size : terminal character size

Return Value

> 0 : Initialize 성공 했을 때
< 0 : Initialize 실패 했을 때

SET__ETHERNET_TYPE__CLIENT 함수

Client Type의 EtherNet 통신을 지원한다. (Default)

Syntax

int SET__ETHERNET_TYPE__CLIENT( );

Parameter

없음

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

SET__ETHERNET_TYPE__SERVER 함수

Server Type의 EtherNet 통신을 지원한다.

Syntax

int SET__ETHERNET_TYPE__SERVER( );

Parameter

없음

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

START_SERVER 함수

Server Type의 EtherNet 통신을 시작한다.

Syntax

int START_SERVER( );

Parameter

없음

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CONNECT 함수

Client Type의 EtherNet 통신을 시작한다.

Syntax

```
int CONNECT( CString* msg );
```

Parameter

msg : Server 쪽에 Connection 실행 후 결과 Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

GET__CURRENT_ETHERNET_TYPE 함수

현재 실행되고 있는 EtherNet Type를 String으로 전달 받는다.

Syntax

CString GET__CURRENT_ETHERNET_TYPE( );

Parameter

없음

Return Value

현재 실행되고 있는 EtherNet Type를 String으로 전달 받는다.

DATA__SEND 함수

Data string을 전송한다.

Syntax

```
int  DATA__SEND( CString *data, const int sec );
```

Parameter

data : data string를 전송하고, 수신된 data를 받아온다.
sec : 전송 timeout

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

DATA__SEND 함수

Data Character Array을 전송한다.

Syntax

```
int  DATA__SEND( char* data, const int data_size, const int sec );
```

Parameter

data : 전송할 data character array
data_size : 전송할 data character size
sec : 전송 timeout

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CLEAR__BUFFER 함수

수신 버퍼에 저장된 모든 data 를 읽어오고, 수신 버퍼는 Clear 된다.

Syntax

```
int  CLEAR__BUFFER( CString* msg );
```

Parameter

msg : 수신 버퍼에 저장된 모든 data 를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

DISCONNECT 함수

연결된 통신 Connection를 Close 시킨다.

Syntax

```
int   DISCONNECT( );
```

Parameter

없음

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

IS_CONNECT 함수

EtherNet 통신의 연결 여부를 확인한다.

Syntax

```
int  IS_CONNECT( );
```

Parameter

없음

Return Value

> 0 : 통신 연결이 성공된 상태일 때
< 0 : 통신 연결이 실패된 상태일 때

DATA__SEND 함수

Data string을 전송한다.

Syntax

```
int  DATA__SEND( const CString& send_data, const int sec, CString* error );
```

Parameter

send_data : 전송할 data string
sec : 전송 timeout
error : 전송시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

DATA__RECV 함수

수신된 Data string을 받아온다.

Syntax

```
int  DATA__RECV( CString* recv_data, const int sec, CString* error );
```

Parameter

recv_data : 수신된 data string
sec : 전송 timeout
error : 수신시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

DATA__RECV_QUEUE 함수

수신된 임시 queue 버퍼에서 Data string을 받아온다.

Syntax

```
int  DATA__RECV( CString* recv_data, const int sec, CString* error );
```

Parameter

recv_data : 수신된 data string
sec : 전송 timeout
error : 수신시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CHAR_SEND 함수

Data string을 전송한다.

Syntax

int CHAR_SEND(const CString& data, const int char_len, const int sec, CString* error);

Parameter

data : 전송할 data string
char_len : 전송할 string size
sec : 전송 timeout
error : 전송시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CHAR_RECV 함수

Data string을 전송 후 수신된 String을 받아온다.

Syntax

int CHAR_RECV(CString* data, const int char_len, const int sec, CString* error);

Parameter

data : data string을 전송 후 수신된 데이터를 받아온다.
char_len : 전송할 string size
sec : 전송 timeout
error : 전송시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CHAR__SEND 함수

Character array을 전송한다.

Syntax

int CHAR__SEND(const char* data, const int char_len, const int sec, CString* error);

Parameter

data : 전송할 character array
char_len : 전송할 character size
sec : 전송 timeout
error : 전송시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CHAR_RECV 함수

Character array을 전송 후 수신된 Character array 받아온다.

Syntax

int CHAR_RECV(char* data, const int char_len, const int sec, CString* error);

Parameter

data : character array을 전송 후 수신된 데이터를 받아온다.
char_len : 전송할 character size
sec : 전송 timeout
error : 전송시 실패할 경우 Error Message를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

INIT__MSG_QUEUE 함수

Message queue를 초기화 한다.

Syntax

```
int   INIT__MSG_QUEUE( );
```

Parameter

없음

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

SET__MSG_QUEUE__PROPERTY 함수

Message내에 ID 정보가 있을 때 몇 바이트 차지하는지? 설정한다.

Syntax

```
int   SET__MSG_QUEUE__PROPERTY( const int id_byte );
```

Parameter

id_byte : Message내 ID 정보의 바이트 수를 지정한다.
1 보다 작으면 ID가 없다고 판단한다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

RECV_QUEUE 함수

수신된 데이터를 내부 Queue 저장소로 옮겨 놓는다.

Syntax

```
int   RECV_QUEUE(   );
```

Parameter

없음.

* 주의사항 : 이 함수는 Monitoring 함수 내에서 호출되어야 한다.
송신과 수신 함수는 비동기 방식으로 실행된다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

GET__MSG_QUEUE 함수

Message queue 저장소에 수신 완료된 하나의 Packet를 받아온다.

Syntax

```
int  GET__MSG_QUEUE( CString* msg );
```

Parameter

msg : Queue에 저장된 하나의 Packet를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

SEARCH__MSG_QUEUE 함수

Message queue 저장소에서 ID와 일치하는 Packet를 받아온다.

Syntax

```
int  SEARCH__MSG_QUEUE( const CString& id_msg, CString* msg );
```

Parameter

id_msg : 검색할 ID
msg : 검색할 Message

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

CLEAR__MEM_BUFFER 함수

수신 버퍼에 수신된 데이터를 함수 parameter로 옮기고 수신 버퍼를 지운다.

Syntax

```
int  CLEAR__MEM_BUFFER( CString* msg );
```

Parameter

msg : 수신 버퍼에 수신된 데이터를 받아온다.

Return Value

> 0 : 성공 했을 때
< 0 : 실패 했을 때

READ_NETWORK_BUFFER 함수

Ethernet 수신 버퍼에 수신된 데이터를 읽어온다

Syntax

```
int  READ_NETWORK_BUFFER( char* recv_data, const int char_len );
```

Parameter

recv_data : Ethernet 수신 버퍼에 수신된 데이터를 받아온다.
char_len : 읽어올 데이터 수

Return Value

> 0 : 성공 했을 때, 읽어온 데이터 수
<= 0 : 실패 했을 때

IS__CONNECT_EX 함수

Ethernet 통신의 연결 상태를 확인한다 .

Syntax

```
int  IS__CONNECT_EX( );
```

Parameter

없음.
Ethernet 통상 상태가 정상적으로 연결되었는지? 알려준다.

Return Value

> 0 : 연결된 상태일 때
< 0 : 연결이 안된 상태

GET__TERMINAL_InSTRING 함수

수신시 설정된 Terminal String 정보를 받아온다.

Syntax

CString GET__TERMINAL_InSTRING( );

Parameter

없음.

Return Value

설정된 수신 End String을 전달한다.

GET__TERMINAL_OutSTRING 함수

송신시 설정된 Terminal String 정보를 받아온다.

Syntax

CString GET__TERMINAL_OutSTRING( );

Parameter

없음.

Return Value

설정된 송신 End String을 전달한다.