

CTCCOMM.DLL Function and Data Type Declarations:

```
//-----
//      "C" Function Prototypes
//-----
#define NO_CHANGE -1
/* Define success and failure which will be used in various contexts,
   so definitions are needed which actually don't specify data type */
#define FAILURE 0
#define SUCCESS 1

typedef unsigned short int  UINT;
typedef UINT              STATUS;
typedef unsigned char      CHAR;
typedef CHAR              BYTE;
typedef CHAR              FLAG;
typedef short int         INT;
typedef unsigned long int  ULONG;
typedef ULONG            DWORD;
typedef ULONG            REGS_16[16];
typedef ULONG            REGS_50[50];
enum comm_type_enum { NO_PORT, SERIAL, ETHER};
typedef enum comm_type_enum COMM_TYPE;
enum comm_port_enum { COMM1, COMM2, COMM3, COMM4, COMM5, COMM6, COMM7, COMM8};
typedef enum comm_port_enum COMM_PORT;

typedef enum
{
    CTCNONE=-1,
    CTC2200=0,
    CTC2200XM,
    CTC2800iE,
    CTC2800iEA,
    CTC2800iEAXM,
    CTCbar,
    CTC2600XM,
    CTC2700iEA,
    CTC2601iEA,
    CTC2601FiEA,
    CTC2602iEA,
    /* Insert new models before here to keep count correct.*/
    CT_MODELS
} ct_model;

// Private Internal Packet Structure
typedef struct { /* 40 bytes + data */
    /* First 8 Bytes are for Primary Routing Information */
    UINT16 destination; /* destination ct-net node address */
    UINT16 source; /* source ct-net node address */
    BYTE version_major;
    BYTE version_minor;
    BYTE type; /* pkt type */
    BYTE spare1;
    /* Next 8 Bytes reserved for Secondary Routing Information */
    BYTE AltRoute[8];
    /* Next 12 Bytes are for Closure information from the orginal sender
       These bytes must be preserved and included in a reply/response
       packet. */
    UINT32 time; /* Creation/Transmit Time of Packet */
    UINT16 exttime; /* reserved expand time to 48bits */
    /* if we go to 64bit time, we will

```

```

                                give up the transaction number
                                or 16bits of the pkt_link
                                does not matter, since this is
                                in the closure section. */
UINT16  transaction;           /* transaction number */
UINT32  pkt_link;             /* link to original packet */
/* Next 12 Bytes may be used by local cpu as scratchpad for
   processing. */
UINT32  timeout;             /* internal timeout */
UINT16  exttimeout;
UINT16  exttimeout2;         /* in-case we go to 64-bit time */
UINT8   src_device;          /* source device of packet */
UINT8   dst_device;          /* destination device of packet */
UINT8   retries;             /* number of times to retry */
BYTE    scratch;
/* The following contains the CTC Message or other data */
BYTE    data[MaxMsgSize];
} Packet;

// Ethernet Address Structure
typedef char EADDR[6];

// Ethernet Packet Structure
typedef struct {
    EADDR destination;
    EADDR source;
    UINT type;
    Packet packet;
} EPacket;

// STATUS = SUCCESS, FAILURE, or an Error Code
STATUS FAR PASCAL _export OpenNetwork( COMM_TYPE port,
    COMM_PORT port_name,
    UINT nid);
STATUS FAR PASCAL _export InitPort( COMM_TYPE port,
    COMM_PORT comm_port,
    UINT nid,
    long baud_rate=NO_CHANGE,
    int parity=NO_CHANGE,
    int word_length=NO_CHANGE,
    int stop_bits=NO_CHANGE);
STATUS FAR PASCAL _export CloseNetwork( void);
STATUS FAR PASCAL _export GetRegister( UINT nid,
    UINT index,
    LONG FAR* value);
STATUS FAR PASCAL _export GetRegister16( UINT nid,
    UINT index,
    REGS_16 FAR* values);
STATUS FAR PASCAL _export GetRegister50( UINT nid,
    UINT bank,
    REGS_50 FAR* values);
STATUS FAR PASCAL _export PutRegister( UINT nid,
    UINT index,
    LONG value);
STATUS FAR PASCAL _export SendData( UINT nid,
    UINT type,
    unsigned char FAR* send_data,
    int send_data_len,
    Packet FAR* rsp_packet);
STATUS FAR PASCAL _export GetModel( UINT nid,
    ct_model FAR *pModel,

```

```
    BOOL FAR *pIsEA);
STATUS FAR PASCAL _export IsExpandedArchitecture( UINT nid,
    BOOL FAR *pIsEA);
void FAR PASCAL _export SetTimeout( ULONG milli);
void FAR PASCAL _export SetRetries( UINT retries);
ULONG FAR PASCAL _export GetTime(UINT type);
void FAR PASCAL _export ClearCounts(void);
void FAR PASCAL _export ClearTimes(void);
UINT FAR PASCAL _export GetBadCounts( UINT type);
STATUS FAR PASCAL _export EnterProgramMode( UINT nid);
STATUS FAR PASCAL _export LeaveProgramMode( UINT nid);
STATUS FAR PASCAL _export LoadProgramPacket( UINT nid,
    byte type,
    UINT VAddress,
    UINT packet_length,
    unsigned char FAR *pProgPkt);
STATUS FAR PASCAL _export LoadNonEAPProgramPacket( UINT nid,
    UINT VAddress,
    UINT packet_length,
    unsigned char FAR *pProgPkt);
STATUS FAR PASCAL _export SendAscii( UINT nid,
    UINT packet_length,
    unsigned char FAR *pASCII,
    Packet FAR* rsp_packet);
EPacket* FAR PASCAL _export GetEPacket(void);
int FAR PASCAL _export ct_type( int len,
    LPSTR driver_buf);
STATUS FAR PASCAL _export ReadAsciiRspSerial( BOOL usingRing,
    Packet FAR* rsp_packet);
```

CTCCOMM.DLL Function and Data Type Declarations:

```
//-----  
//      Visual Basic Function Prototypes  
//-----  
'Enum Constants for port types  
Global Const NO_PORT% = 0      ' No comm port  
Global Const SERIAL% = 1      ' Serial port  
Global Const ETHER% = 2       ' Ethernet  
  
'Enum Constants for Com ports  
Global Const Com1% = 0  
Global Const Com2% = 1  
Global Const Com3% = 2  
Global Const Com4% = 3  
Global Const Com5% = 4  
Global Const Com6% = 5  
Global Const Com7% = 6  
Global Const Com8% = 7  
  
Declare Function OPENNETWORK Lib "ctccomm.dll" (ByVal CommType as Integer,  
    ByVal CommPort as Integer, ByVal NodeId as Integer) as Integer  
Declare Function INITPORT Lib "ctccomm.dll" (ByVal CommType as Integer,  
    ByVal CommPort as Integer, ByVal NodeId As Integer, ByVal CommBaud as  
    Long, ByVal CommParity as Integer, ByVal CommData as Integer, ByVal  
    CommStop as Integer) as Integer  
Declare Function CLOSENETWORK Lib "ctccomm.dll" () as Integer  
Declare Function GETREGISTER Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal index as Integer, ByRef value as Long) as Integer  
Declare Function GETREGISTER16 Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal index as Integer, ByRef values_array as Long) as Integer  
Declare Function GETREGISTER50 Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal index as Integer, ByRef values_array as Long) as Integer  
Declare Function PUTREGISTER Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal index as Integer, ByVal value as Long) as Integer  
Declare Function SENDDATA Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal pkttype as Integer, ByVal pkt As String, ByVal pktlen as Integer,  
    ByVal returnpkt As String) as Integer  
Declare Function GETMODEL Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByRef pModel as Integer, ByRef pIsEA as Boolean) as Integer  
Declare Function ISEXPANDEDARCHITECTURE Lib "ctccomm.dll" (ByVal NodeId as  
    Integer, ByRef pIsEA as Boolean) as Integer  
Declare Sub SETTIMEOUT Lib "ctccomm.dll" (ByVal timeout as Long)  
Declare Sub SETRETRIES Lib "ctccomm.dll" (ByVal retries as Integer)  
Declare Function GETTIME Lib "ctccomm.dll" (ByVal timeType as Integer) as Long  
Declare Sub CLEARCOUNTS Lib "ctccomm.dll" ()  
Declare Sub CLEARTIMES Lib "ctccomm.dll" ()  
Declare Function GETBADCOUNTS Lib "ctccomm.dll" (ByVal countType as Integer)  
    as Integer  
Declare Function ENTERPROGRAMMODE Lib "ctccomm.dll" (ByVal NodeId as Integer)  
    as Integer  
Declare Function LEAVEPROGRAMMODE Lib "ctccomm.dll" (ByVal NodeId as Integer)  
    as Integer  
Declare Function LOADPROGRAMPACKET Lib "ctccomm.dll" (ByVal NodeId as Integer,  
    ByVal type as Integer, ByVal Vaddress as Integer, ByVal packet_length as  
    Integer, ByRef pProgPkt as String) as Integer  
Declare Function LOADNONEAPROGRAMPACKET Lib "ctccomm.dll" (ByVal NodeId as  
    Integer, ByVal Vaddress as Integer, ByVal packet_length as Integer,  
    ByRef pProgPkt as String) as Integer  
Declare Function SENDASCII Lib "ctccomm.dll" (ByVal NodeId as Integer,
```

```
    ByVal pktlen as Integer, ByVal pkt As String, ByVal returnpkt As String)
    as Integer
Declare Function GETEPACKET Lib "ctccomm.dll" () as String
Declare Function CT_TYPE Lib "ctccomm.dll" (ByVal len as Integer,
    ByVal driver_buf as String) as integer
Declare Function GETEPACKET Lib "ctccomm.dll" (ByVal usingRing as Boolean,
    ByVal rsp_packet as String) as Integer
```