

UHF-RW-MP-232-4A

4 port UHF RFID reader/writer

DLL documents name: Mr915ApiV20.dll

1 Reader management functions

1.1 ConnectReader

Functions instructions	ConnectReader (HANDLE hPort, int nTransMode, int nNetPort, LPCTSTR strReaderIP, LPCTSTR strSerialNum);
Function	Initialize equipment COMM connect, baud rate and other parameters
Parameter	StrSerialNum : Com serial Number hPort : PC connect reader' effective control code, nTransMode: communication mode choose (0-TCP communication; 1-RS232 COMM communication), int nNetPort: TCP Commuication' internet interface No (Default: 100), strReaderIP: reader' hardware IP address (default: 192.168.1.200)
Return Value	See attachment on Page 10

1.2 Disconnect

Function instructions	Disconnect (HANDLE hPort)
Function	Close com port
Parameter	hPort - Have opened interface control code
Return Value	See attachment on Page 10

1.3 SetBaudRate

Function instructions	SetBaudRate(HANDLE hPort, USHORT usBaudRate, BYTE byReaderAddr)
Function	Set reader's baud rate
Parameter	usBaudRate: Baud rate value 0~4 (0:9600 ,1:19200 ,2:38400 ,3:57600 and 4:115200. byReaderAddr: reader address, Use for fixed reader RS485, Default 0XFF(handheld reader and module is invalid parameter)
Return Value	See attachment on Page 10

1.4 Reset

Function instructions	ResetReader (HANDLE hPort, BYTE byReaderAddr)
Function	Reader Reset
Parameter	byReaderAddr: reader address, Use for fixed reader RS485, Default 0XFF(handheld reader and module is invalid parameter)
Return Value	See attachment on Page 10

1.5 GetFirmwareVersion

Function instructions	GetFirmwareVersion (HANDLE hPort, BYTE &byMajor, BYTE &byMinor, BYTE byReaderAddr)
Function	Read the reader's firmware version number
Parameter	byMajor: major version information byMinor: minor version information
Return Value	See attachment on Page 10

1.6 SetRf

Functions instructions	SetRf(HANDLE hPort, BYTE byPower, BYTE byFreqType, BYTE byReaderAddr)
Function	Set reader's power and frequency parameter
Parameter	byPower: power value (0-30 ,corresponding to 0-30dBm) byFreqType: 0 : China standard (920MHz-925MHz) 1: America standard (902M-928M) others are special type (i.e.868MHz)
Return Value	See attachment on Page 10

1.7 GetRf

Function instructions	GetRf(HANDLE hPort, BYTE &byPower, BYTE &byFreqType, BYTE byReaderAddr)
Function	Read the reader's current RF parameter
Parameters	byPower : Power value byFreqType: Frequency type
Return Value	See attachment on Page 10

1.6 SetAntenna

Function instructions	SetAntenna(HANDLE hPort, BYTE byAntenna, BYTE byReaderAddr)
Function	Set reader's power and frequency parameters
Parameters	byAntenna: antenna # , 0-4 (max. 4 antenna)
Return value	See attachment on Page 10

1.7 GetAntenna

Functions instructions	GetAntenna(HANDLE hPort, BYTE &byAntenna, BYTE byReaderAddr)
Function	Set reader ' current RF parameter
Parameter	byAntenna: antenna # 0-4 (max. 4 antenna)
Return Value	See attachment on Page 10

1.8 GetInput

Function instructions	GetInput(HANDLE hPort,INPUT_LEVEL &inValue,BYTE byReaderAddr);
Function	Inquiry reader ' input interface level state
Parameters	INPUT_LEVEL structure: { BYTE byLevel[8]; // input COMM level state,0-7 each unit for a input COMM, 1 : high level,0 : low level (4-antennas version only 0,1 two input COMM are effective) }
Return Value	See attachment on Page 10

1.9 SetOutPort

Function instructions	SetOutPort(HANDLE hPort, BYTE byPortNum, BYTE byLevel, BYTE byReaderAddr);
Function	Set reader output COMM high-low level
Parameters	byPortNum is COMM # (0-3), byLevel is output level(0 : low level,1 : high level)
Return Value	See attachment on Page 10

2. GetParameter

Function instructions	GetParameter(HANDLE hPort, BYTE byType, BYTE &byValue, BYTE byReaderAddr);
Function	Inquiry reader ' basic parameter' setting
Parameter	By type is parameters setting type that need to inquiry , including the following definitions: PARAM_INDICATE 0x01 // Identification Hint PARAM_READER_ADDR 0x02 //Reader address - RS485 communication use byValue is related return parameters values PARAM_INDICAT Type ,byValue ' the lowest binary digit : 1 : have "buzzer" / 0 : no 'buzzer" PARAM_READER_ADDR type. byValue is 0-240 (reader' 485 Communication address)
Return Value	See attachment on Page 10

2.1 SetParameter

Function Instructions	SetParameter(HANDLE hPort, BYTE byType, BYTE byValue, BYTE byReaderAddr);
Function	Setting reader' basical parameters
Parameters	<p>By type is parameters setting type that need to inquiry , including the following definitions:</p> <pre> PARAM_INDICATE 0x01 // Identification Hint PARAM_READER_ADDR 0x02 // Reader address - RS485 communication use byValue is related return parameters values PARAM_INDICAT Type ,byValue ' the lowest binary digit : 1: have "buzzer" , 0 : no 'buzzer" PARAM_READER_ADDR type. byValue is 0-240 (reader' 485 Communication address) </pre>
Return value	See attachment on Page 10

2.2 GetNetSetting

Function Instructions	GetNetSetting(HANDLE hPort, NET_SETTING &nsNetSetting, BYTE byReaderAddr);
Function	Inquiry reader' TCP internet communication COM ' IP address code ,subnet mask, Default Gateway, TCP com parameters
Parameters	<p>NET_SETTING structure:</p> <pre> { BYTE byIpValue[4]; //IP address BYTE byMarkValue[4]; //subnet mask BYTE byGateValue[4]; //default gateway int m_nPort; //Com No } </pre>
Return Value	See attachment on Page 10

2.3 SetNetSetting

Function Instructions	SetNetSetting(HANDLE hPort, NET_SETTING nsNetSetting, BYTE byReaderAddr);
Function	Inquiry reader' TCP internet communication COM ' IP address code ,subnet mask,Default Gateway, TCP com parameters
Parameters	NET_SETTING structure: <pre>{ BYTE byIpValue[4]; //IP address BYTE byMarkValue[4]; //subnet mask BYTE byGateValue[4]; //Default gateway int m_nPort; //COM No }</pre>
Return Value	See attachment on Page 10

2 ISO18000-6B tags operation Functions

2.1 IsoMultiTagIdentify

Function Instructions	IsoMultiTagIdentify(HANDLE hPort, short &nCount, TAGS_CONTROL *taTags, BYTE byReaderAddr)
Function	ISO18000-6B multi-tag identification contains repeat data filtration. Please use ClearIDBuffer functions to clearing reader's internal buffer before restart new operation of multi-tag identification.
Parameters	nCount: Reading tags' quantity this time ; taTags: Reading tags' data then store in the form of Tag ID TAGS_CONTROL Structure: <pre>{ BYTE byyLabelType; // Tag type: 1=ISO18000-6B tag, // 2=ISO18000-6C/EPC tag BYTE byyAntenna; // Antennas No(Related multi-antennas // reader) BYTE byyLabelID[8]; // ISO18000-6B tags ID number BYTE byyLabelEPC[12]; // ISO18000-6C/EPC tags' EPC code }</pre>
Return Values	See attachment on Page 10

2.4 IsoReadWithID

Function	IsoReadWithID(HANDLE hPort, ReadControl ReadCtrl, Instruction TAGDATA_CONTROL &tdData, BYTE ReaderAddr);
Function	Reading appointed ID number' tags data, each time reading appointed address ' beginning 8bytes data
Parameters	ReadControl structure { BYTE TagID[8] BYTE byAddress BYTE byAntenna} TAGDATA_CONTROLstructure: { BYTE byAntenna; //Antennas No(Related multi-antennas reader) BYTE byLabelData[8];// ISO18000-6B tags inside store data}
Return Value	See attachment on Page 10
Example	

2.5 IsoWriteWithID

Function	IsoWriteWithID(HANDLE hPort, WriteControl WriteCtrl, BYTE Instructions ReaderAddr)
Function	Write data into appointed tags
Parameter	TagID, 8 bytesUID ; WriteControl structure { BYTE TagID[8] // Tags' ID BYTE byAddress // write into address BYTE byValue[4] // Write into data .that only need to add the first byte
Return Value	See attachment on Page 10

2.8 IsoBlockWrite

Function	IsoBlockWrite(HANDLE hPort, WriteControl WriteCtrl, BYTE Instructions ReaderAddr)
Function	Successive write 4 bytes contexts into appointed address
Parameter	WriteControl structure { BYTE TagID[8] // Tags' ID BYTE byAddress // write into address, must be 4bytes multiple BYTE byValue[4] // write into data,4 bytes }
Return Value	See attachment on Page 10

2.6 IsoLockTag

Function	IsoLockTag(HANDLE hPort ,BYTE byRomAddr, BYTE ReaderAddr)
Instruction	
Function	Lock and write for appointed tags' address, once locked ,this address cannot unlock again
Parameters	iRomAddr, Need to lock writing' tags address
Return value	See attachment on Page 10

2.7 IsoQueryLock

Function	IsoQueryLock(HANDLE hPort, BYTE byRomAddr,BYTE &byIsLocked, instruction BYTE ReaderAddr)
Function	Query appointed address if is locked or not
Parameter	byRomAddr, Query address ;byIsLocked, Return status, 0 : not locked ,1 : is locked
Return Value	See attachment on Page 10

3. EPC GEN2 Tag Operation Function

3.1 Gen2MultiTagIdentify

Function	Gen2MultiTagIdentify(HANDLE hPort, short &nCount, Instruction TAGS_CONTROL *taTags, BYTE byReaderAddr)
Function	EPC GEN2 multi-tag identification contains repeat data filtration
Parameters	nCount: Reading tags' quantity this time ; taTags: Reading tags' data,then store in the form of Tag ID TAGS_CONTROL structure: { BYTE byyLabelType; // Tag type: 1=ISO18000-6B tag, 2=ISO18000-6C/EPC tag BYTE byyAntenna; // Antennas No(Related multi-antennas reader) BYTE byyLabelID[8]; // ISO18000-6B tags' ID number BYTE byyLabelEPC[12]; // ISO18000-6C/EPC tags' EPC code }
Return Value	See attachment on Page 10

3.2 Gen2WriteEPC

Function	Gen2WriteEPC(HANDLE hPort , EPCWriteControl byWriteCtrl, BYTE ReaderAddr)
Function	EPC GEN2 tag: EPC write data, write 1 word(2 bytes) data on each time
Input Parameter	EPCWriteControl structure <pre>{ BYTE byWordPtr // write into address, Word as a unit BYTE byValue[2] // data written, 2 bytes }</pre>
Return Value	See attachment on Page 10

3.3 Gen2LockTag

Function	Gen2LockTag(HANDLE hPort, BYTE byMemBank = 1, Byte ReaderAddr)
Function	Write and lock operation to EPC tag, write and lock one area each time
Input parameter	byMemBank lock area, 0 is reserved area, 1 is EPC area, 2 is TID area, 3 is user area
Return Value	See attachment on Page 10

3.4 Gen2KillTag

Function	Gen2KillTag(HANDLE hPort , unsigned int PassWord, BYTE ReaderAddr)
Function	EPC GEN2 kill tags
Input parameter	PassWord, Totally have 32bytes kill password, Tags cannot killed if its all password are all 0
Return	See attachment on Page 10

3.5 Gen2InitEPCSize

Function	Gen2InitEPCSize(HANDLE hPort, BYTE byWordCount = 6, BYTE ReaderAddr)
Function	All 0 : Initialize EPC code length ,default initialization as 96bits(6 bytes), and all values are 0
Input parameter	byWordCount, initialized number(1 word=2bytes)
Return Value	See attachment on Page 10

3.7 Gen2ReadBlock

Function	Gen2ReadBlock(HANDLE hPort, EPC_READ_CONTROL
Instruction	epcReadCtrl,EPC_LABEL_VALUE &epcValue, BYTE ReaderAddr)
Function	Reading EPC tags' TID code,not all tags have TID code
Input	<p>epcValue, Reading' data, :</p> <p>Refer to below structure definition:</p> <pre> EPC_READ_CONTROL { BYTE byMembank; // Membank, reading area(kill password and query password area is 0, EPC coding area is 1, TID area is 2, USER area is 3) BYTE byWordPtr; // reading data address, word is a unit (Word=2 Byte) BYTE byWordCnt; // (reading length(only are effective when operating in user area , when reading others area, no need to length } EPC_LABEL_VALUE {BYTE byKillPWD[4]; // Tag ' kill password, 4 bytes, 2 word BYTE byAccessPWD[4];// Tag' query password, 4 bytes, 2 word BYTE byLabelEPC[12]; // Tag' EPC code , 6 word, 12 bytes BYTE byLabelTID[8]; // Tag'TID code, 8bytes BYTE byMemoData[16];// Tag' user data area(Can support 16 bytes,8 word at the most } </pre>
Return Value	See attachment on Page 10

3.8 Gen2BlockWrite

Function	Gen2BlockWrite(HANDLE hPort,EPC_WRITE_CONTROL
Instruction	epcWriteCtrl , BYTE ReaderAddr)
Function	EPC GEN2 tag block write
Parameters	<p>epcValue data and parameter that need to wrte,</p> <p>refer to below structure definition:</p> <pre> EPC_WRITE_CONTROL { BYTE byMembank; // writing area, kill password and query password area is 0,EPC code is 1,user area is 3 BYTE byWordPtr; // Writing into word address 0-7,only need to appointed write into user area BYTE byWordLen; //Writing into data length,only need to appointed write into user area(the maximum related to tags' supporting length, normal tags only support the largest 2 bytes BYTE byKillPWD[4]; // Tag ' kill password, 4 bytes, 2 word BYTE byAccessPWD[4];// Tag' query password, 4 bytes, 2 word BYTE byLabelEPC[12]; //Tag' EPC code , 6 word, 12 bytes BYTE byMemoData[16];// Tag' user data area(Can support 16 bytes,8 word at the most } </pre>
Return Value	See attachment on Page 10

4. Tags' data processing function

4.2 ClearIDBuffer

Function	ClearIDBuffer(HANDLE hPort, BYTE byReaderAddr);
Instruction	
Function	Clear reader tags data buffer area. It can be used before multi-tags identification each time
Parameters	
Return Value	See attachment on Page 10

4.3 QueryIDCount

Function	QueryIDCount(HANDLE hPort, BYTE &byCount, BYTE ReaderAddr);
Instruction	
Function	Query tags' quantity when buffering
Parameters	Count, Return tags' quantity pointer
Return Value	See attachment on Page 10

5. API - Returning parameters & definition

FUCCESS_RETURN	2001	Operation successfully return, operation finish
ERR_NOTAG_RETURN	2002	No tag
ERR_HANDLE_VALUE	2003	COM Control Code error
ERR_UDATA_LEN	2004	data length error
ERR_UDATA_ADDRESS	2006	data address error
ERR_RDATA_LEN	2007	receiving data length is not matched
ERR_SDATA_FAIL	2008	Incorrect data format
ERR_SCMND_FAIL	2009	send command fail
ERR_READ_FAIL	2010	read tags fail
ERR_WRITE_FAIL	2011	write tags fail
ERR_LOCK_FAIL	2012	lock tags fail
ERR_ERASE_FAIL	2013	erase tags fail
ERR_PORT_OPENED	2014	open registering form fail
ERR_OPEN_REGSTER	2015	Interface have opened ,control code is not INVALID_HANDLE_VALUE
ERR_LOAD_INI_LOST	2016	Equipment messages cannot be found or reading fail, Key parameters cannot be initialized
ERR_DEV_INIT_FAIL	2017	To initialize equipment fail
ERR_PORT_OPEN_FAIL	2018	Open com port fail
ERR_PORT_CLOSE_FAIL	2019	Close com port fail
ERR_OUT_PARA_LEN	2020	Parameter' data length is out of range
ERR_GET_PARA_FAIL	2021	Query equipment parameters fail
ERR_SET_PARA_FAIL	2022	Set equipment parameters fail
ERR_OTHER_FAIL	2000	Others unknown error