FC-AE 仿真测试卡

软件接口使用说明书

部门名称	研发部		
文档编号			
版 本 号	1.00		
拟　　制		日 期	
批　　准		日 期	
发放人员			

文档更新记录

日期	更新人	版本	备注

目　　录

1 结构变量说明

1.1 发送状态的设备参数（TbhSendCfg）

功能：发送状态的设备参数		
结构体定义		
typedef struct{ unsigned int nIsAsm; unsigned int nPktPri; unsigned int nChSel; }TBhSendCfg;		
参数名称	参数含义	备注
unsigned int nIsAsm	发送数据类型：是否为 ASM	
unsigned int nPktPri	发送报文的优先级	
unsigned int nChSel	选择发送端口 A/B	

1.2 端口状态（TBhFcPortStatus）

功能：端口状态		
结构体定义		
typedef struct{ unsigned int nLinkStatus; unsigned int nFcSpeed; unsigned int nFiberStatus; unsigned char nRes[8]; }TBhFcPortStatus;		
参数名称	参数含义	备注
unsigned int nLinkStatus	FC 端口连接状态	
unsigned int nFcSpeed	FC 端口的物理速度	
Unsigned int nFiberStatus	FC 核的缓存到缓存的信用值	
unsigned char nRes[8]	保留	

1.3 设备PCIE参数信息（TbhFcDevInfo）

功能：设备 PCIE 参数信息
结构体定义

```
typedef struct{
    unsigned int nVendorID;
    unsigned int nDeviceID;
    unsigned int nBusNum;
    unsigned int nSlotNum;
    unsigned int nFuncNum;
    unsigned int nHwVer;
    char sFcSerial[SERIAL_LEN];
}TbhFcDevInfo;
```

参数名称	参数含义	备注
unsigned int nVendorID	板卡制造商编号	
unsigned int nDeviceID	板卡设备编号	
unsigned int nBusNum	板卡上的总线编号	
unsigned int nSlotNum	板卡上的插槽编号	
unsigned int nFuncNum	板卡上的功能编号	
unsigned int nHwVer	板卡的硬件版本	
char sFcSerial[SERIAL_LEN]	板卡的生产编号	

1.4 接收端口MIB（TBhFcRXMib）

功能：接收端口 MIB

结构体定义

```
typedef struct{
    unsigned int nRXAllFrm;
    unsigned int nRXAllByte0;
    unsigned int nRXAllByte1;
    unsigned int nRX0To127;
    unsigned int nRX128To255;
    unsigned int nRX256To511;
    unsigned int nRX512To1023;
    unsigned int nRX1024To2112;
    unsigned int nRXShortErr;
    unsigned int nRXLongErr;
```

```
    unsigned int nRxCrcError;
    unsigned int nRXByteFlux;
    unsigned int nRXFrmFlux;
    unsigned int nRxPortIdError;
    unsigned int nRxPortEnableError;
    unsigned int nRxOffsetError;
    unsigned int nRxSeqIDError;
}TBhFcRXMib;
```

参数名称	参数含义	备注
unsigned int nRXAllFrm	Rx_frame_all 接收到所有数据包	
unsigned int nRXAllByte0	接收的所有数据字节的长度低 32 位	
unsigned int nRXAllByte1	接收的所有数据字节的长度高 32 位	
unsigned int nRX0To127	接收的数据区长度在 0 到 128 范围内的数量	
unsigned int nRX128To255	接收的数据区长度在 129 到 256 范围内的数量	
unsigned int nRX256To511	接收的数据区长度在 257 到 512 范围内的数量	
unsigned int nRX512To1023	接收的数据区长度在 513 到 1024 范围内的数量	
unsigned int nRX1024To2112	接收的数据区长度在 1024 到 2112 范围内的数量	
unsigned int nRXShortErr	包长过小错误包的数量	
unsigned int nRXLongErr	包长过大错误包的数量	
unsigned int nRxCrcError	CRC 错误帧数量	
unsigned int nRXByteFlux	接收端口 Byte 流量	
unsigned int nRXFrmFlux	接收端口数据帧流量	
unsigned int nRxPortIdError	接收端口 ID 错误	
unsigned int nRxPortEnableError	接收端口使能错误	
unsigned int nRxOffsetError	接收端口一个序列内按偏移重组使能错误	

unsigned int nRxSeqIDError	端口的序列 ID 错误	

1.5 发送端口MIB（TBhFcTxMib）

功能：发送端口 MIB

结构体定义

```
typedef struct{
    unsigned int nTXAllFrm;
    unsigned int nTXAllByte0;
    unsigned int nTXAllByte1;
    unsigned int nTx0To127;
    unsigned int nTx128To255;
    unsigned int nTx256To511;
    unsigned int nTx512To1023;
    unsigned int nTx1024To2112;
    unsigned int nTxByteFlux;
    unsigned int nTxFrmFlux;
}TBhFcTxMib;
```

参数名称	参数含义	备注
unsigned int nTXAllFrm	Tx_frame_all 发送的所有数据包	
unsigned int nTXAllByte0	发送的所有数据字节的长度低 32 位	
unsigned int nTXAllByte1	发送的所有数据字节的长度高 32 位	
unsigned int nTx0To127	发送的数据区长度在 0 到 128 范围内的数量	
unsigned int nTx128To255	发送的数据区长度在 129 到 256 范围内的数量	
unsigned int nTx256To511	发送的数据区长度在 257 到 512 范围内的数量	
unsigned int nTx512To1023	发送的数据区长度在 513 到 1024 范围内的数量	
unsigned int nTx1024To2112	发送的数据区长度在 1024 到 2112 范围内的数量	
unsigned int nTxByteFlux	发送端口 Byte 流量	
unsigned int nTxFrmFlux	发送端口数据帧流量	

1.6 其他统计信息MIB（TBhFcSysMib）

功能：其他统计信息 MIB		
结构体定义		
typedef struct{ unsigned int nRXMinErr; unsigned int nRXMaxErr; unsigned int nRXCrcErr; unsigned int nRxerr; unsigned int nRxDrop; }TBhFcSysMib;		
参数名称	参数含义	备注
unsigned int nRXMinErr	接收包长过小错误包的数量	
unsigned int nRXMaxErr	接收包长过大错误包的数量	
Unsigned int nRXCrcErr	接收 CRC 错误帧数量	
unsigned int nRxerr	接收错误	
unsigned int nRxDrop	保留	

1.7 MIB类型（TBhFcMib）

功能：MIB 类型		
结构体定义		
typedef enum{ eMibRx, eMibTx, eMibSys, eMibNr }TBhFcMib;		
参数名称	参数含义	备注
eMibRx	接收端口 MIB	
eMibTx	发送端口 MIB	
eMibSys	其他信息统计 MIB	
eMibNr	未知类型/MIB 类型个数	

1.8 端口数据流向（TBhFcPortDir）

功能：端口数据流向		
结构体定义		
typedef enum{ eDirSend, eDirRecv, ePortDirNr, }TBhFcPortDir;		
参数名称	参数含义	备注
eDirSend	接收端口	
eDirRecv	发送端口	
ePortDirNr	未知端口	

1.9 端口类型（TBhFcPortType）

功能：端口类型		
结构体定义		
typedef enum{ eTypeNormal, eTypeBlock, eTypeSpecial, ePortTypeNr, }TBhFcPortType;		
参数名称	参数含义	备注
eTypeNormal	正常传输类型	
eTypeBlock	数据块传输类型	
eTypeSpecial	特殊传输类型	
ePortTypeNr	未知类型	

1.10 端口选择（TBhFcPortSel）

功能：端口选择
结构体定义

```
typedef enum{
    ePortA = 1,
    ePortB,
    ePortAB,
}TBhFcPortSel;
```

参数名称	参数含义	备注
ePortA	选择 A 端口	
ePortB	选择 B 端口	
ePortAB	选择 A 和 B 端口	

1.11 端口的消息类型（TBhFcMsgType）

功能：端口的消息类型

结构体定义

```
typedef enum{
    eTypeAsm,
}TBhFcMsgType;
```

参数名称	参数含义	备注
eTypeAsm	ASM 消息（匿名订户消息）	

1.12 时间描述类型（TBhFcTimeDir）

功能：时间描述类型

结构体定义

```
typedef enum{
    eSys2Dev,
    eDev2Fc,
}TBhFcTimeDir
```

参数名称	参数含义	备注
eSys2Dev	系统时间信息转换为板卡时间	
eDev2Fc	板卡时间信息转换为系统时间	

1.13 板卡时间戳信息形式（TBhFcDevTime）

功能：板卡时间戳信息形式

结构体定义

```
typedef struct {
    unsigned int nHighTime;
    unsigned int nLowTime;
}TBhFcDevTime;
```

参数名称	参数含义	备注
unsigned int nHighTime	主机时间同步（高位）	day/hour
unsigned int nLowTime	主机时间同步（低位）	min/s/ms/us

1.14 板卡工作模式（TBhFcDevMode）

功能：板卡工作模式

结构体定义

```
typedef enum{
    eModeNormal,
    eModeCapture,
    eDevModeNr
}TBhFcDevMode;
```

参数名称	参数含义	备注
eModeNormal	正常模式	
eModeCapture	采集模式	
eDevModeNr	未知模式	

1.15 板卡工作速率（TBhFcDevSpeed）

功能：板卡工作速率

结构体定义

```
typedef enum{
    eSpeed1G,
    eSpeed2G,
    eSpeedNr
}TBhFcDevSpeed;
```

参数名称	参数含义	备注
eSpeed1G	工作速率 1G	
eSpeed2G	工作速率 2G	

eDevModeNr	工作速率未知（错误）	

1.16 IRIG-B模式（TBhFcIrigMode）

功能：IRIG-B 模式		
结构体定义		
typedef enum{ eModeMaster, eModeSlave, eIrigModeNr }TBhFcIrigMode;		
参数名称	参数含义	备注
eModeMaster	主模式	
eModeSlave	从模式	
eIrigModeNr	未知模式	

1.17 板卡时间戳信息转换形式（TBhFcTime）

功能：板卡时间戳信息转换形式		
结构体定义		
typedef struct{ int nDays; int nHours; int nMins; int nSecs; int nMsecs; int nUsecs; }TBhFcTime;		
参数名称	参数含义	备注
int nDays;	日	
int nHours	小时	
int nMins;	分钟	
int nSecs;	秒	
int nMsecs;	毫秒	
int nUsecs;	微秒	

1.18 板卡配置信息（TBhFcCardCfg）

功能：板卡配置信息		
结构体定义		
typedef struct{ TBhFcDevMode eWorkMode; TBhFcDevSpeed eSpeed; TBhFcIrigMode eIrigbMod; }TBhFcCardCfg;		
参数名称	参数含义	备注
TBhFcDevMode eWorkMode	板卡工作模式	typedef enum{ eModeNormal, eModeCapture, eDevModeNr }TBhFcDevMode;
TBhFcDevSpeed eSpeed	板卡工作速率	typedef enum{ eSpeed1G, eSpeed2G, eSpeedNr }TBhFcDevSpeed;
TBhFcIrigMode eIrigbMod	IRIG-B 模式	typedef enum{ eModeMaster, eModeSlave, eIrigModeNr }TBhFcIrigMode;

1.19 端口配置信息（TBhFcPortCfg）

功能：端口配置信息
结构体定义
typedef struct { unsigned int nPortId; TBhFcPortType ePortType; TBhFcPortDir ePortDir; unsigned int nSid; unsigned int nDid; unsigned int nOxId; TBhFcPortSel ePortPortSel; unsigned int nPortMsgType; unsigned int nMsgId;

```
    unsigned int nSeqId;
    unsigned int nFragSeq;
    unsigned int nFragOffset;
    unsigned int nPri;
    unsigned int nMaxLen;
}TBhFcPortCfg;
```

参数名称	参数含义	备注
unsigned int nPortId	端口 ID	
TBhFcPortType ePortType	端口类型	typedef enum{ eTypeNormal, eTypeBlock, eTypeSpecial, ePortTypeNr, }TBhFcPortType;
TBhFcPortDir ePortDir	端口数据流向	typedef enum{ eDirSend, eDirRecv, ePortDirNr, }TBhFcPortDir;
unsigned int nSid	端口的源 ID	
unsigned int nDid	端口的目的 ID	
unsigned int nOxId	端口的交换 ID	
TBhFcPortSel ePortPortSel	端口选择	typedef enum{ ePortA = 1, ePortB, ePortAB, }TbhFcPortSel;
unsigned int nPortMsgType	端口的消息类型	
unsigned int nMsgId	端口消息 ID	
unsigned int nSeqId;	端口的序列 ID	
unsigned int nFragSeq	端口一个序列内按序列号重组使能	
unsigned int nFragOffset	端口一个序列内按偏移重组使能	
unsigned int nPri	端口数据优先级	

unsigned int nMaxLen	端口允许的最大包长	

1.20 接收端口信息（TBhFcRecvInfo）

功能：接收端口信息		
结构体定义		
typedef struct{ TBhFcPortSel ePort; TBhFcDevTime nRecvTime; TBhFcMsgType eMsgType; }TBhFcRecvInfo;		
参数名称	参数含义	备注
TBhFcPortSel ePort	板卡工作模式	typedef enum{ ePortA = 1, ePortB, ePortAB, }TbhFcPortSel;
TBhFcDevTime nRecvTime	接收时间戳	typedef struct { unsigned int nHighTime; unsigned int nLowTime; }TBhFcDevTime;
TBhFcMsgType eMsgType	端口的消息类型	typedef enum{ eTypeAsm, }TBhFcMsgType;

1.21 系统时间戳信息形式（TBhFcSysTime）

功能：系统时间戳信息形式		
结构体定义		
typedef struct{ int wYear; int wMonth; int wDayOfWeek; int wDay; int wHour; int wMinute; int wSecond; int wMilliseconds; } TBhFcSysTime;		
参数名称	参数含义	备注
int wYear	年	
int wMonth	月	
int wDayOfWeek	星期	
int wDays;	日	
int wHour	小时	
int wMinute	分钟	
int wSecond	秒	
int wMilliseconds	毫秒	

1.22 监听情况下的工作模式（TBhFcMonMode）

功能：监视情况下的工作模式		
结构体定义		
typedef enum{ eMonSelect, eMonFilter, eMonModeNr, }TBhFcMonMode;		
参数名称	参数含义	备注
eMonSelect	Select 模式	
eMonFilter	Filter 模式	
eMonModeNr	保留	

1.23 监听情况下的操作（TBhFcMonOperation）

功能：监视模式操作		
结构体定义		
typedef enum{ eStopRecv, eStartRecv, ePauseRecv, eMonOpNr, }TBhFcMonOperation;		
参数名称	参数含义	备注
eStopRecv	停止接收	
eStartRecv	开始接收	
ePauseRecv	暂停接收	
eMonOpNr	保留	

1.24 监听情况下的配置信息（TBhFcMonCfg）

功能：监视情况下的配置信息		
结构体定义		
typedef struct{ unsigned int nMonThresdHold; TBhFcMonMode nWorkMode; unsigned int nMonTimedOut; }TBhFcMonCfg;		
参数名称	参数含义	备注
Unsigned int nMonThresdHold	接收中断门限寄存器大小	
TBhFcMonMode nWorkMode	工作模式	
unsigned int nMonTimedOut	超时时间	

1.25 监听情况下的端口配置（TBhFcMonFilter）

功能：监视情况下的端口配置
结构体定义
typedef struct{ unsigned int nSid; unsigned int nDid; unsigned int nType;

```
        unsigned int nMsgId;
        unsigned int nOxId;
        unsigned int nSeqId;
        unsigned int nPortSel;
        unsigned char nFilterFiledSel[BH_FC_FILTER_FILED_CNT];
}TBhFcMonFilter;
```

参数名称	参数含义	备注
unsigned int nSid;	端口的源 ID	
unsigned int nDid;	端口的目的 ID	
unsigned int nType;	端口的消息类型	
unsigned int nMsgId;	端口消息 ID	
unsigned int nOxId;	端口的交换 ID	
unsigned int nSeqId;	端口的序列 ID	
unsigned int nPortSel;	端口选择	
unsigned char nFilterFiledSel[BH_FC_FILTER_FILED_CNT];	端口数量\选择	

2 2.接口函数说明

2.1 BhFcInit(void)

函数原型：int BhFcInit(void)		
函数功能		
驱动初始化，完成接口卡扫描并将接口卡信息存入对应的数据结构。		
参数名称	参数含义	备注
无		
返回值：成功，返回板卡数量； 失败，返回失败错误码（BH_FC_ERR_INIT）。		

2.2 BhFcDeInit(void)

函数原型：int BhFcDeInit(void)		
函数功能		
驱动注销，释放板卡资源。		
参数名称	参数含义	备注
无		
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码（BH_FC_ERR_UNINIT）。		

2.3 BhFcOpenCard(int nCardNum)

函数原型：int BhFcOpenCard(int nCardNum)		
函数功能		
打开板卡		
参数名称	参数含义	备注
int nCardNum	板卡编号	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.4 BhFcCloseCard

函数原型：int BhFcCloseCard(int nCardNum)
函数功能

关闭板卡		
参数名称	参数含义	备注
int nCardNum	板卡编号	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.5 BhFcLoadCfg

函数原型：int BhFcLoadCfg(int nCardNum,char *pFileName)		
函数功能		
加载 FC 配置文件		
参数名称	参数含义	备注
int nCardNum	板卡编号	
char *pFileName	配置文件路径	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.6 BhFcSetCardCfg

函数原型：int BhFcSetCardCfg(int nCardNum,TBhFcCardCfg nCfg)		
函数功能		
配置板卡工作模式、工作速率、IRIG-B 模式		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcCardCfg nCfg	板卡配置信息	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.7 BhFcAddPort

函数原型： int BhFcAddPort(int nCardNum,TBhFcPortCfg * pPortList,int nPortCnt)
函数功能
添加端口

参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcPortCfg* pPortList	端口配置信息	
int nPortCnt	添加端口数量	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.8 BhFcDelPort

函数原型： int BhFcDelPort(int nCardNum,unsigned int nPortId, TBhFcPortDir ePortDir)		
函数功能		
删除端口		
参数名称	参数含义	备注
int nCardNum	板卡编号	
unsigned int nPortId	端口 ID	
TBhFcPortDir ePortDir	端口数据流向	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.9 BhFcOpenPort

函数原型： int BhFcOpenPort(int nCardNum,unsigned int nPortId, TBhFcPortHandle *pPortHandle,TBhFcPortDir eDir)		
函数功能		
打开端口		
参数名称	参数含义	备注
int nCardNum	板卡编号	
unsigned int nPortId	端口 ID	
TBhFcPortHandle *pPortHandle	创建的端口的操作句柄指针	
TBhFcPortDir ePortDir	端口数据流向	

返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。

2.10 BhFcClosePort

函数原型： int BhFcClosePort(int nCardNum,TbhFcPortHandle nPortHandle, TBhFcPortDir eDir)		
函数功能		
关闭端口		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcPortHandle nPortHandle	创建的端口的操作句柄	
TBhFcPortDir ePortDir	端口数据流向	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.11 BhFcSendData

函数原型： int BhFcSendData(int nCardNum,TBhFcPortHandle nHandle, char *pBuf,unsigned int nBufSize,TBhSendCfg nCfg)		
函数功能		
发送数据		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcPortHandle nPortHandle	创建的端口的操作句柄	
char *pBuf	发送数据的数据缓冲区	
unsigned int nBufSize	发送数据的长度	
TBhSendCfg nCfg	需要发送的数据帧的配置	
返回值：成功，返回需要发送的数据的长度； 失败，返回失败错误码。		

2.12 BhFcRecvData

函数原型： int BhFcRecvData(int nCardNum,TBhFcPortHandle nPortHandle, char *pBuf,TBhFcRecvInfo *pTime, int nBufSize,int nWaitTime)		
函数功能		
接收数据		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcPortHandle nPortHandle	创建的端口的操作句柄	
char *pBuf	发送数据的数据缓冲区	
TBhFcRecvInfo *pTime	用来存储数据帧的信息	
unsigned int nBufSize	发送数据的长度	
int nWaitTime	等待接收数据的时间	
返回值：成功，返回实际接收到的数据长度； 失败，返回失败错误码。		

2.13 BhFcGetCardNum

函数原型：int BhFcGetCardNum(void)		
函数功能		
得到板卡数量		
参数名称	参数含义	备注
无		
返回值：成功，返回板卡数量； 失败，返回失败错误码。		

2.14 BhFcGetCardInfo

函数原型：int BhFcGetCardInfo(int nCardNum,TbhFcDevInfo *pDevInfo)		
函数功能		
得到所需板卡信息		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcDevInfo *pDevInfo	板卡参数信息指针	

返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。

2.15 BhFcGetPortStatus

函数原型： int BhFcGetPortStatus(int nCardNum,TBhFcPortStatus * pPortStatus)		
函数功能		
得到所需板卡信息		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcPortStatus *pPortStatus	端口状态	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.16 BhFcDumpCardInfo

函数原型： void BhFcDumpCardInfo(int nCardNum,TbhFcDevInfo nDevInfo)		
函数功能		
打印版卡信息		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TbhFcDevInfo nDevInfo	板卡参数信息	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.17 BhFcReadReg

函数原型：int BhFcReadReg(int nCardNum,int nAddr)		
函数功能		
从读寄存器值（存储区）		
参数名称	参数含义	备注
int nCardNum	板卡编号	
int nAddr	寄存器地址	
返回值：成功，返回相应寄存器中的值； 失败，返回失败错误码。		

2.18 BhFcWriteReg

函数原型： int BhFcWriteReg(int nCardNum,int nAddr,unsigned int nValue)		
函数功能		
向寄存器写值（存储区）		
参数名称	参数含义	备注
int nCardNum	板卡编号	
int nAddr	寄存器地址	
unsigned int nValue	需要写入的值	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.19 BhFcReadCfgReg

函数原型：int BhFcReadCfgReg(int nCardNum,int nAddr)		
函数功能		
读寄存器值（PCI 配置区）		
参数名称	参数含义	备注
int nCardNum	板卡编号	
int nAddr	寄存器地址	
返回值：成功，返回相应寄存器中的值； 失败，返回失败错误码。		

2.20 BhFcWriteCfgReg

函数原型： int BhFcWriteCfgReg(int nCardNum,int nAddr,unsigned int nValue)		
函数功能		
向寄存器写值（PCI 配置区）		
参数名称	参数含义	备注
int nCardNum	板卡编号	
int nAddr	寄存器地址	
unsigned int nValue	需要写入的值	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.21 BhFcGetLastErr

函数原型：const char *BhFcGetLastErr(void)		
函数功能		
获得最新的错误信息		
参数名称	参数含义	备注
无		
返回值：错误信息的地址。		

2.22 BhFcClearMibs

函数原型：int BhFcClearMibs(int nCardNum)		
函数功能		
清除端口的统计信息		
参数名称	参数含义	备注
int nCardNum	板卡编号	
返回值：返回 BH_FC_OP_OK;		

2.23 BhFcGetTxMibs

函数原型： int BhFcGetTxMibs(int nCardNum, TBhFcTxMib *pMibs, TBhFcPortSel ePort)		
函数功能		
得到发送端口的信息统计		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcTxMib *pMibs	发送端口 MIB 的指针	
TBhFcPortSel ePort	端口的选择	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.24 BhFcGetRxMibs

函数原型： int BhFcGetRxMibs(int nCardNum, TBhFcRXMib *pMibs, TBhFcPortSel ePort)
函数功能
得到接收端口的信息统计

参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcRXMib *pMibs	接收端口 MIB 的指针	
TBhFcPortSel ePort	端口的选择	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.25 BhFcGetSysMibs

函数原型： int BhFcGetSysMibs(int nCardNum,TBhFcSysMib *pMibs)		
函数功能		
得到其他信息统计		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcSysMib *pMibs	信息统计的指针	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.26 BhFcGetPortOverFlow

函数原型： int BhFcGetPortOverFlow(int nCardNum,unsigned int nPortIndex, TBhFcPortDir eDir)		
函数功能		
得到其他信息的 MIB 统计		
参数名称	参数含义	备注
int nCardNum	板卡编号	
unsigned int nPortIndex	端口索引号	
TBhFcPortDir eDir	端口数据流向	
返回值：成功，返回端口溢出帧数; 失败，返回失败错误码。		

2.27 BhFcDriverChangeTime

函数原型： int BhFcDriverChangeTime (TBhFcSysTime *pSysTime, TBhFcSysTime *pSysTime,TBhFcTime *pFcTime, TBhFcTimeDir eDir)		
函数功能		
将系统时间信息与板卡时间信息进行相互转换		
参数名称	参数含义	备注
TBhFcSysTime *pSysTime	描述系统时间的指针	设置板卡时间使用
TBhFcSysTime *pSysTime	描述系统时间的指针	获取板卡时间使用
TBhFcTime *pFcTime	描述设备板卡时间的指针	
TBhFcTimeDir eDir	时间描述类型	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.28 BhFcSetMonOperation

函数原型： int BhFcSetMonOperation(int nCardNum, TBhFcMonOperation eOperation);		
函数功能		
在监视模式下选择操作，并执行		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcMonOperation eOperation	监听模式操作	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.29 BhFcMonRecvData

函数原型： int BhFcMonRecvData(int nCardNum,char *pBuf, int nBufSize, int nWaitTime,int *pDataType);		
函数功能		
监听模式下进行数据接收		
参数名称	参数含义	备注
int nCardNum	板卡编号	
char *pBuf	数据指针	
int nBufSize	数据大小	
int nWaitTime	等待接收数据时间	
int *pDataType	数据类型 （真实\非真实）	#define MON_DATA_REA 0x1 #define MON_DATA_NOT_REAL 0x0
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.30 BhFcSetMonCfg

函数原型： int BhFcSetMonCfg(int nCardNum,TBhFcMonCfg nCfg);		
函数功能		
监听模式下进行信息配置		
参数名称	参数含义	备注
int nCardNum	板卡编号	
TBhFcMonCfg nCfg	配置信息	
返回值：成功，返回 BH_FC_OP_OK; 失败，返回失败错误码。		

2.31 BhFcAddMonFilter

函数原型： int BhFcAddMonFilter(int nCardNum,int nFilterCnt, TBhFcMonFilter *pMonFilter);		
函数功能		
监听模式下添加过滤配置		
参数名称	参数含义	备注
int nCardNum	板卡编号	
int nFilterCnt	寄存器个数	
TBhFcMonFilter *pMonFilter	过滤配置	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

2.32 BhFcDelMonFilter

函数原型： int BhFcDelMonFilter(int nCardNum,int nIndex);		
函数功能		
监听模式下删除过滤配置		
参数名称	参数含义	备注
int nCardNum	板卡编号	
nIndex	索引号	
返回值：成功，返回 BH_FC_OP_OK； 失败，返回失败错误码。		

3 返回码说明

宏定义	值	描述
BH_FC_OP_OK	0x0	操作成功
BH_FC_ERR_PARA	-1	参数错误
BH_FC_ERR_INIT	-2	初始化错误
BH_FC_ERR_UNINIT	-3	释放系统资源失败
BH_FC_ERR_OPEN	-4	打开发生错误
BH_FC_ERR_CLOSE	-5	关闭发生错误
BH_FC_ERR_SEND	-6	发送数据错误
BH_FC_ERR_RECV	-7	接收数据错误
BH_FC_ERR_SEND_OVERFLOW	-8	发送数据溢出错误
BH_FC_ERR_XML	-9	XLM 配置错误
BH_FC_ERR_CFG	-10	端口配置信息错误
BH_FC_ERR_MSG	-11	句柄错误
BH_FC_ERR_FIFO	-12	配置信息错误
BH_FC_ERR_PORT	-13	打开/关闭端口失败

4 常量定义

宏定义	值	描述
SERIAL_LEN	16	长度
BH_FC_MAX_NOR_PORT	256	正常端口个数
BH_FC_MAX_BLK_PORT	16	数据块传输端口最大个数
FC_MAX_SPE_PORT	1	特殊类型端口个数
BH_FC_MAX_PORT_CNT	BH_FC_MAX_PORT_CNT (BH_FC_MAX_NOR_PORT + BH_FC_MAX_BLK_PORT +BH_FC_MAX_SPE_PORT)	最大端口个数
BH_FC_PORT_MASK_CNT	9	端口个数
BH_FC_MAX_DEV_CNT	8	设备最大数量
BH_FC_ASM_MAX_PAYLOAD	2096	ASM 有效负载
BH_FC_NASM_MAX_PAYLOAD	2148	NOT ASM 有效负载
BH_FC_BLK_MAX_PAYLOAD	16*1024*1024	数据块最大负载
BH_FC_MAX_FILTERCNT	256	最大端口个数
BH_FC_FILTER_MASKCNT	8	端口个数
BH_FC_FILTER_FILED_CNT	6	端口申请个数

5 代码实例

本例主要实现了对 FC 网卡的初始化、开卡、添加端口、打开端口、发送数据、关闭端口、删除端口以及注销的驱动流程测试。

```
#include "stdafx.h"
#include "BhFcDriver.h"
#include <vxWorks.h>
#include <taskLib.h>
#include "BhFcStruct.h"

#ifdef DEBUG
   #define DebugMsg printf
#endif
#define SOCKDATA_LEN 2096
#define SendLength 500
char buf_data[SOCKDATA_LEN] = {0};

int GetContinue()
{
      char buf[10] = {0};
      printf("Press Enter to continue,Q to quit....\n");
      printf("Enter:");
      gets(buf);
      if((buf[0] == 'Q') || (buf[0] == 'q'))
      {
            return 0;
      }
      return 1;
}

int BhFcTest(void)
{
      int nCardNum = 0;
      int i = 0;
      int nSendLen = 0;
      TBhFcCardCfg nCfg = {0};
      TBhFcPortStatus nPortStatus = {0};
      TBhSendCfg nCfg_send = {0};
            nCfg_send.nChSel =ePortA;
            nCfg_send.nIsAsm = IS_ASM;
             nCfg_send.nPktPri = 0;
       TBhFcPortCfg nPortTypePara = {1,    eTypeNormal,eDirSend,   1,    1, 1,
      ePortA,
                                                  1,     1,      1,    0,       0,       0, 2096};
       TBhFcPortHandle nPortHandle = -1;
```

```
    if((nCardNum = BhFcInit()) < 0)
    {
        DebugMsg("%s\n",BhFcGetLastErr());
        getchar();
        return 0;
    }

    for(i = 0; i < nCardNum; i++)
    {
        TbhFcDevInfo nDevInfo;
        if(BhFcGetCardInfo(i,&nDevInfo) != BH_FC_OP_OK)
        {
            DebugMsg("%s\n",BhFcGetLastErr());
            return 0;
        }
        BhFcDumpCardInfo(i,nDevInfo);
    }
    if(nCardNum > 0)
    {
        char buf[10];
        printf("Select Card:");
        gets(buf);
        nCardNum = strtol(buf,NULL,0);
        printf("Selected Card %d\n",nCardNum);
    }
    if(BhFcOpenCard(nCardNum) < 0)
    {
        printf("[Err]%s\n",BhFcGetLastErr());
        getchar();
        goto ErrOpen;
        return 0;
    }
    DebugMsg("Open Card success \n");



    if(BhFcAddPort(nCardNum,&nPortTypePara,1) != BH_FC_OP_OK)
    {
        printf("%s\n",BhFcGetLastErr());
        getchar();
        goto ErrNeedClose;
    }


/*get speed cfg*/
    {
```

```
        char bufs[10];
        printf("Select Card Speed:\n");
        printf("1         : Select 1G\n");
        printf("other : Select 2G\n");
        printf("Enter your choice:");
        gets(bufs);
        if(bufs[0] == '1')
        {
            nCfg.eSpeed = eSpeed1G;
        }else{
            nCfg.eSpeed = eSpeed2G;
        }
    }
    nCfg.eWorkMode = eModeNormal;
    if(BhFcSetCardCfg(nCardNum,nCfg) != BH_FC_OP_OK)
    {
        printf("%s\n",BhFcGetLastErr());
        getchar();
        goto ErrNeedClose;
    }

    if(BhFcGetPortStatus(nCardNum,&nPortStatus) != BH_FC_OP_OK)
    {
        printf("%s\n",BhFcGetLastErr());
        getchar();
        goto ErrNeedClose;
    }
    else
    {
        DebugMsg("PortLink Status:\nPortA : %s \nPortB : %s\nSpeed : %s\n",
            (nPortStatus.nLinkStatus & 0x1 ) ? "Link Up": "Link Down" ,
            (nPortStatus.nLinkStatus & 0x2 ) ? "Link Up": "Link Down",
            nPortStatus.nFcSpeed == 0 ? "1G\n" : "2G\n");
        if(nPortStatus.nLinkStatus != 0x3)
        {
            if(!GetContinue())
            {
                goto ErrNeedClose;
            }
        }
    }

if(BhFcOpenPort(nCardNum,nPortTypePara.nPortId,&nPortHandle,eDirSend)
                                                    != BH_FC_OP_OK )
    {
        DebugMsg("%s\n",BhFcGetLastErr());
        goto ErrOpenPort;
```

```
    }
    else
        DebugMsg("Open port success!\n");

   for(i = 0; i < SOCKDATA_LEN; i++)
       {
           buf_data[i] = i + 1;
       }


for(i = 0;i < SendLength; i++)
{
  if((nSendLen=BhFcSendData(nCardNum,nPortHandle,buf_data,sizeof(buf_data),
                                                          nCfg_send))<=0)
  {
                DebugMsg("%s\n",BhFcGetLastErr());
                 goto ErrOpenPort;
  }
}

ErrOpenPort:

    BhFcClosePort(nCardNum,nPortHandle,eDirSend);
    BhFcDelPort(nCardNum,nPortTypePara.nPortId,eDirSend);

ErrNeedClose:
    BhFcCloseCard(nCardNum);
    BhFcDeInit();
ErrOpen:
    return 0;
}
```