分析思路

FF协议栈-功能块参数

# 触发入口

目前西安没有设备，缺乏下装环境，现在的触发入口有工程编译、工程保存、工程打开等会触发创建协议数据，进而写入功能块参数信息，该功能需要将参数实际值和默认值不一致的写入

参照栈信息

CFFProtocolAdapter::CreateProtocolDownloadData(CDevice \* 0x111b45c8 {CGeneralSlave}, int 0, unsigned char \* & 0x00000000, unsigned int & 0) line 1630

CGeneralSlave::GetSpecialInfo(unsigned int & 0, unsigned char \* & 0x00000000) line 5963 + 29 bytes

CGeneralSlave::CreateCfgModule(CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &> & {CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &>}) line 4931

CGeneralLink::CreateCfgModule(CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &> & {CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &>}) line 5186 + 26 bytes

CGeneralGateway::CreateCfgModule(CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &> & {CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &>}) line 1186 + 12 bytes

CGeneralLink::CreateCfgModule(CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &> & {CMap<unsigned int,unsigned int &,CDownloadInfoBase \*,CDownloadInfoBase \* &>}) line 5186 + 26 bytes

CGeneralLink::CompareLastDownInfoAndUpdateData(CList<CDownloadInfoBase \*,CDownloadInfoBase \* &> \* 0x12ff9e20 {CList<CDownloadInfoBase \*,CDownloadInfoBase \* &>}, CMemFile & {pos=2372 size=2372}, CMemFile & {pos=2372 size=2372}) line 1757 + 12 bytes

CControl::HWCreateDownloadData(CObject \* 0x0ba054c8 {CDownloadInfoBase}, int & 0, char \* & 0x00000000, int & 0, char \* & 0x00000000) line 501 + 43 bytes

HWCreateDownloadDataImp(unsigned int 0, CObject \* 0x0ba054c8 {CDownloadInfoBase}, int & 0, char \* & 0x00000000, int & 0, char \* & 0x00000000) line 876 + 36 bytes

HWDATA! 0133926e()

HWDATA! 01324180()

APPDATA! 00d06091() line 205 + 47 bytes

COMPILE! 02a77907() line 1481 + 24 bytes

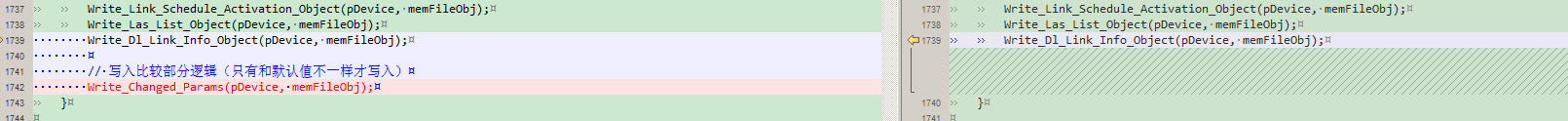
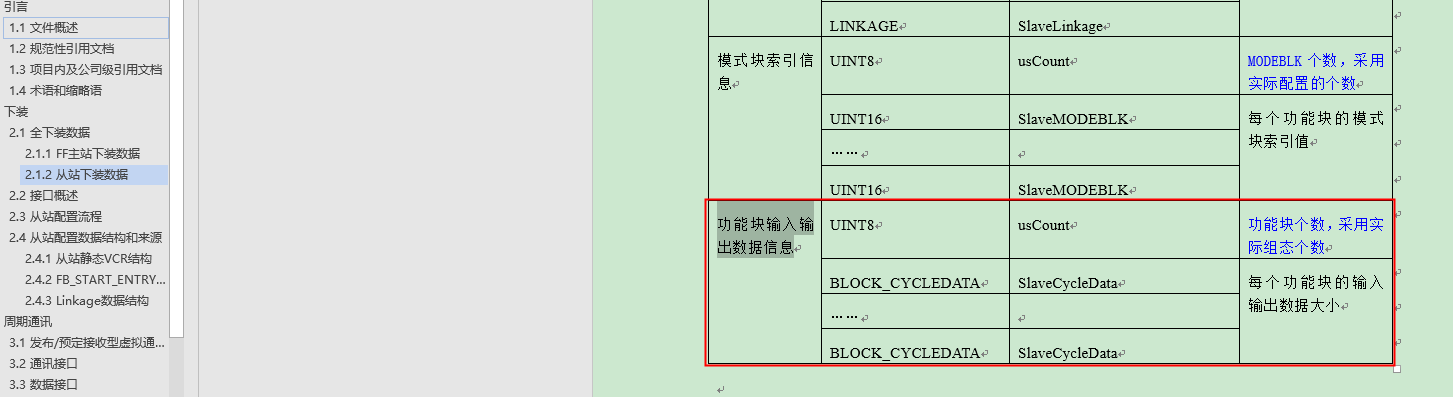
COMPILE! 02a57755() line 15134 + 26 bytes

COMPILE! 02a1d863() line 2169

AUTOTHINK! 00420019() line 2916 + 66 bytes

# 涉及范围

跟赵工和郝工沟通，本次的参数修改写入，暂不涉及主控，只写入从站中，且数据写在功能块输入输出数据信息之后



# 数据格式

|  |  |  |  |
| --- | --- | --- | --- |
| 功能块参数信息 | UINT16 | usCount | 功能块修改参数个数 |
| 参数遍历 | | |
| UINT16 | index | 参数索引（从1开始）+块ID |
| UINT16 | subindex | 子参数索引（从1开始） |
| UINT8 | usLength | 参数数据长度（字节数） |
| UINT8 | data[usLength] | 参数数据 |

注：

该数据格式已分别和赵工、郝工达成一致

# 实现

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*功能 : 写入比较部分逻辑（只有和默认值不一样才写入）

\*参数 : pDeviceObj: 设备对象

\* memFileObj: 内存文件对象

\*返回值 : 无

\*创建者 : Wang Bin

\*创建日期 : 2020.02.25

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void CFFProtocolAdapter::Write\_Changed\_Params(CDevice \* pDeviceObj, CMemFile & memFileObj)

{

AT\_ASSERT(pDeviceObj != NULL && memFileObj.m\_hFile != NULL);

UINT32 uiLength = memFileObj.GetLength();

UINT16 uiParamCount = 0x00000000;

memFileObj.Write(&uiParamCount, sizeof(uiParamCount));

CList<CGeneralModule \*, CGeneralModule \*&> moduleList;

pDeviceObj->GetModuleList(moduleList);

for(POSITION pos = moduleList.GetHeadPosition(); pos != NULL;)

{

CGeneralModule \* pModule = moduleList.GetNext(pos);

HW\_MODULE \* pHWModule = pModule->GetSourceData();

for (UINT uiParamIdx = 0; uiParamIdx < pHWModule->uiParamCount; uiParamIdx++)

{

HW\_PARAM \* pParam = pHWModule->pParam + uiParamIdx;

if (HW\_PARAM\_COMBOX == pParam->emControlType)

{

UINT uiDefaultIdx = pParam->Value.tagOption.uiDefaultIdx;

int iDefaultValue = pParam->Value.tagOption.pOption[uiDefaultIdx].iValue;

CString sActualValue = pModule->GetParamValue(pParam->tagKey);

if (iDefaultValue != atoi(sActualValue))

{

UINT16 index = (UINT16)strtoul(pParam->tagKey.pItem->cValue, NULL, 10);

UINT16 subindex = 0;

memFileObj.Write(&index, sizeof(index));

memFileObj.Write(&subindex, sizeof(subindex));

UINT8 length = 4;

memFileObj.Write(&length, sizeof(length));

UINT32 newVaule = (UINT32)atoi(sActualValue);

memFileObj.Write(&newVaule, sizeof(newVaule));

uiParamCount ++;

}

}

else

{

CString sDefaultValue = pParam->Value.tagNoraml.cDefaultValue;

CString sActualValue = pModule->GetParamValue(pParam->tagKey);

if (sDefaultValue != sActualValue)

{

UINT16 index = (UINT16)strtoul(pParam->tagKey.pItem->cValue, NULL, 10);

UINT16 subindex = 0;

memFileObj.Write(&index, sizeof(index));

memFileObj.Write(&subindex, sizeof(subindex));

if (CString(pParam->cName) == "TAG\_DESC")

{

char cValue[32];

memset(&cValue, '\0', sizeof(cValue));

strcpy(cValue, sActualValue);

UINT8 length = 32;

memFileObj.Write(&length, sizeof(length));

memFileObj.Write(cValue, sizeof(cValue));

}

else

{

UINT8 length = 4;

memFileObj.Write(&length, sizeof(length));

UINT32 newVaule = (UINT32)atoi(sActualValue);

memFileObj.Write(&newVaule, sizeof(newVaule));

}

uiParamCount ++;

}

}

for (UINT uiChildIdx = 0; uiChildIdx < pParam->uiChildCount; uiChildIdx++)

{

HW\_PARAM \* pChild = pParam->pChild + uiChildIdx;

if (HW\_PARAM\_COMBOX == pChild->emControlType)

{

UINT uiDefaultIdx = pChild->Value.tagOption.uiDefaultIdx;

int iDefaultValue = pChild->Value.tagOption.pOption[uiDefaultIdx].iValue;

CString sActualValue = pModule->GetParamValue(pParam->tagKey, atoi(pChild->tagKey.pItem->cValue));

if (iDefaultValue != atoi(sActualValue))

{

UINT16 index = (UINT16)strtoul(pParam->tagKey.pItem->cValue, NULL, 10);

UINT16 subindex = (UINT16)strtoul(pChild->tagKey.pItem->cValue, NULL, 10);

memFileObj.Write(&index, sizeof(index));

memFileObj.Write(&subindex, sizeof(subindex));

UINT8 length = 4;

memFileObj.Write(&length, sizeof(length));

UINT32 newVaule = (UINT32)atoi(sActualValue);

memFileObj.Write(&newVaule, sizeof(newVaule));

uiParamCount ++;

}

}

else

{

CString sDefaultValue = pChild->Value.tagNoraml.cDefaultValue;

CString sActualValue = pModule->GetParamValue(pParam->tagKey, atoi(pChild->tagKey.pItem->cValue));

if (sDefaultValue != sActualValue)

{

UINT16 index = (UINT16)strtoul(pParam->tagKey.pItem->cValue, NULL, 10);

UINT16 subindex = (UINT16)strtoul(pChild->tagKey.pItem->cValue, NULL, 10);

memFileObj.Write(&index, sizeof(index));

memFileObj.Write(&subindex, sizeof(subindex));

UINT8 length = 4;

memFileObj.Write(&length, sizeof(length));

UINT32 newVaule = (UINT32)atoi(sActualValue);

memFileObj.Write(&newVaule, sizeof(newVaule));

uiParamCount ++;

}

}

}

}

}

// 回填参数数量

if (this->IsBigEnd())

{

uiParamCount = BaseFunc::TransEndian32(uiParamCount);

}

memFileObj.Seek(uiLength, CFile::begin);

memFileObj.Write(&uiParamCount, sizeof(uiParamCount));

memFileObj.SeekToEnd();

}

# 遗留问题

和赵工沟通，参数需要的索引值目前没实现，需要参考通道的索引值添加下

（2.26完成）