文件名称：基础技术.SDD.01.CPL SDB文件说明文档

文件编号：A12-B13-000090

项目名称：MACS V6.5.1-基础技术开发

项目编号：B11059A-5-A

物料编码：

版 本 号：A

文件密级：机密

文件状态：CFC

受控标识：受控

|  |  |
| --- | --- |
| 拟制： | 师嘉 2012 年 03 月 25 日 |
| 审核： | 2012 年 03 月 26日 |
| 会签： |  |
|  |  |
| 批准： | 李宗杰 2012 年 03 月 日 |

**文件发放范围**：

**修订页**

| 序号 | 版本号 | 修订内容简述 | 拟制/日期 | 审核 | 批准 |
| --- | --- | --- | --- | --- | --- |
|  | A | 创建 | 师嘉/2012.03.25 | 师嘉 | 李宗杰 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**目 录**

[1. 概述 1](#_Toc323113215)

[1.1. 项目背景 1](#_Toc323113216)

[1.2. 编写目的 1](#_Toc323113217)

[2. SDB文件的内容 1](#_Toc323113218)

[3. SDB文件的格式 1](#_Toc323113219)

[3.1. SDB文件头部分 2](#_Toc323113220)

[3.2. 类型定义部分 2](#_Toc323113221)

[3.2.1. 类型定义头 (Type Header) 3](#_Toc323113222)

[3.2.2. 类型定义体 (Type Element) 3](#_Toc323113223)

[3.2.3. 自定义类型 4](#_Toc323113224)

[3.2.4. 数组类型 4](#_Toc323113225)

[3.3. 变量定义部分 5](#_Toc323113226)

[3.3.1. 变量定义头（Var Header） 5](#_Toc323113227)

[3.3.2. 变量定义体（Var Element） 5](#_Toc323113228)

[3.4. POU索引信息部分 6](#_Toc323113229)

[3.4.1. POU索引信息头（POU Index Header） 6](#_Toc323113230)

[3.4.2. POU索引定义体（POU Info Element） 7](#_Toc323113231)

# 概述

## 项目背景

## 编写目的

本文档主要说明符号表SDB文件的详细内容与格式，适用于MACS 6.5.1项目的开发与维护人员。

# SDB文件的内容

SDB文件主要包括了：

* 组态工程中涉及到的变量类型的相关信息；
* 组态工程中用户定义和系统自动生成的变量的相关信息；
* 组态工程中涉及到得POU的相关信息。

# SDB文件的格式

SDB文件结构如下图所示：



图3-1 SDB表文件结构示意图

下面针对每部分内容进行详细介绍：

## SDB文件头部分

SDB文件头部分记录了组态工程的标识号，同时记录了SDB文件的CRC32校验和，用于上位机对SDB文件进行校验。

该部分内容结构定义如下：

struct SDBHeader

{

ULONG ulTag; //SDBT\_TYPE 0x0001

ULONG ulHeaderSize; //SDB表文件头大小

ULONG ulSdbVersion; //目前为0x0003

ULONG ulSdbProjectId; //组态工程标识号

ULONG ulSdbChecksum; //SDB校验和，采用CRC32校验算法

ULONG ulSdbSize; //SDB文件的总长度（以字节为单位）

unsigned long ulMAreaLen;

unsigned long ulIAreaLen;

unsigned long ulOAreaLen;

unsigned long ulGAreaLen;

unsigned long ulRAreaLen;

unsigned long ulLAreaLen;

unsigned long ulBAreaLen;

unsigned long ulReserved0;

unsigned long ulReserved1;

unsigned long ulReserved2;

};

## 类型定义部分

变量的类型定义部分默认由类型头说明和若干自定义类型说明体组成，对于用户自定义类型要加上Userdef type 的相关定义说明，目前自定义类型主要包括；数组、结构体（功能块），各自定义类型结构在文档后续有详细说明：

具体格式如下：

### 类型定义头 (Type Header)

类型定义头的格式如下：

struct SDBListHeader

{

ULONG m\_ulTag; // SDBT\_TYPETABLE 0x0002

ULONG m\_ulSize; // 类型定义部分的长度

ULONG m\_ulVersion; // 为 0x00;

ULONG m\_ulCount; // 定义变量类型的数目

};

### 类型定义体 (Type Element)

每个类型定义体的结构如下所示：

struct SDBTypeInfo

{

ULONG ulTag; // SDBT\_TYPE 0x0004

ULONG ulSize; // 该SDBTypeInfo的长度

ULONG ulTypeClass; //类型值

ULONG ulTypeSize; //该类型变量所占字节

USHORT uNameLen; //类型名称长度

char szName[1]; //类型名称，占ulNameLen字节

};

其中ulTypeClass的种类有：（取某一枚举值）

enum TypeClass

{

TYPE\_BOOL,

TYPE\_INT,

TYPE\_BYTE,

TYPE\_WORD,

TYPE\_DINT,

TYPE\_DWORD,

TYPE\_REAL,

TYPE\_TIME,

TYPE\_STRING,

TYPE\_ARRAY,

TYPE\_ENUM,

TYPE\_USERDEF,

TYPE\_BITORBYTE,

TYPE\_POINTER,

TYPE\_SINT,

TYPE\_USINT,

TYPE\_UINT,

TYPE\_UDINT,

TYPE\_DATE,

TYPE\_TOD,

TYPE\_DT,

TYPE\_VOID,

TYPE\_LREAL,

TYPE\_REF

TYPE\_STRUCT

};

对于自定义类型，除了包含上面类型定义的基本描述信息，针对不同的自定义类型还需要加入各自的说明信息：

### 枚举类型

如果ulTypeClass为TYPE\_ENUM，则该类型为枚举类型，还需要额外加入成员个数信息及每个成员的说明信息。

struct EnumDesc

{

ULONG ulElement; //Number of Vars in Enum

};

Number of Vars \* SDBEnumVarInfo(Var Element)

在枚举类型的定义体中，将该枚举类型中的所有成员按照枚举类型成员变量说明体的格式进行了描述，成员信息主要包括成员名和初始值，该部分格式说明在变量说明体中将详细介绍。

### 结构体类型

如果ulTypeClass为TYPE\_STRUCT，则该类型为结构体类型，还需要额外加入成员个数信息及每个成员的说明信息：

struct StructDesc

{

ULONG ulElement; // Number of vars in UserStruct

};

Number of vars \* SDBStrVarInfo (Var element)

在结构体类型的定义体中，将该结构体类型中所有成员按照结构体类型成员变量说明体的格式进行了描述，成员信息主要包括成员名、变量类型、变量所属区域、变量大小、变量相对偏移、变量属性，该部分格式说明在变量说明体中将详细介绍。

### 功能块类型

如果ulTypeClass为TYPE\_USERDEF，则该类型为功能块类型，还需要额外加入成员个数信息及每个成员的说明信息：

struct FBDesc

{

ULONG ulElement; //Number of vars in UserFB

};

Number of vars \* SDBFBVarInfo(Var element)

在功能块类型的定义体中，将该功能块类型中所有成员按照功能块类型成员变量说明体的格式进行了描述，成员信息主要包括成员名、变量类型、变量所属区域、变量大小、变量相对偏移、变量属性、变量所在的引脚、变量所在引脚中的位置，该部分格式说明在变量说明体中将详细说明。

### 数组类型

如果ulTypeClass为TYPE\_ ARRAY，则该类型为数组类型，还需要加入数组维数及每一维的上下标信息，结构如下：

struct ArrayInfo

{

ULONG ulTypeId; // 数据元素类型在类型表中的索引值

ULONG ulDimension; // 数组维数

};

Number of dimensions \* < ArrayBound >

ArrayBound结构如下所示：

struct ArrayBound

{

ULONG ulLower; // 下标

ULONG ulUpper; // 上标

};

## 变量定义部分

变量的类型定义部分由变量头说明和若干个变量说明体组成，每个变量体说明中主要包括变量对应的类型索引值（变量的类型对应在SDB类型说明部分中的相对位置）、变量所在区域信息、变量相对于所在区域的偏移、变量名称等信息，用于上位机通过变量名称获得变量的偏移信息，来完成对变量运算值的监视。

该部分具体格式如下：

### 变量定义头（Var Header）

变量定义头格式如下：

struct SDBListHeader

{

ULONG m\_ulTag; // SDBT\_VARTABLE 0x0003

ULONG m\_ulSize; // 变量定义部分的总长度

ULONG m\_ulVersion; // 为 0x00;

ULONG m\_ulCount; // 定义变量的数目

};

### 变量定义体（Var Element）

每个变量定义体的结构如下：

struct SDBVarInfo

{

BYTE btTag; // SDBT\_VAR 0x05

WORD wSize; // 该变量定义的长度

WORD wTypeId; // 该变量类型在类型表中的索引

BYTE btRefId; // 变量所属变量区

ULONG ulOffset; // 变量偏移

char cProperty; // 变量属性

BYTE btNameLen; // 变量名长度

char szName[]; // 变量名称占ulNameLen字节

}

为了压缩SDB表文件的大小，该部分内容较MACS6.5.0部分进行了修改，对于工程总控中的测点变量（总控中所谓的点），目前只保存外层变量的信息，不再保存其每个成员（项）的信息；若总控需要监视具体点项的信息，则需要自己通过类型描述的信息来得到点项的变量区域、变量大小、变量偏移等信息，由自己组织成为需要的点项信息。对于简单类型的变量还按照原来的方式处理。

目前工程中变量的属性分为：掉电保持、同步、掉电并且同步、不掉电也不同步，分别对应不同的RefID，对应关系如下表所示：

表3-1 变量属性与RefID的对应关系

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RefID | 7 | 3 | 8 | 4 |
| 变量属性 | 掉电保持 | 同步 | 掉电且同步 | 不掉电也不同步 |

针对此变化每一个测点变量至多会生成4个变量，分别对应4种属性，若该测点变量中不具有某种属性的成员，则AT不会生成该属性对应的变量，该4个变量名字相同，以RefID进行区分；工程总控构建完整点项信息时，需要根据类型中成员变量的属性来判断选择哪一个外层变量进行构建。

下面举例进行简单说明：

10号控制站工程中有AVI类型的测点变量PN10\_AVI\_01，则AutoThink首先会在SDB表中生成AVI类型的说明信息，里面包括每个成员的VarInfo结构；同时变量定义部分会生成四个变量体描述信息，名称相同均为”.PN10\_AVI\_01”但分别对应4个RefID；假设用户需要监视PN10\_AVI\_01的AV引脚，需要通过AVI类型说明部分查找到AV引脚，并确认其对应的RefID为3，则此处需要使用RefID为3的PN10\_AVI\_01变量，再将类型信息中AV引脚的相对偏移与RefID为3的PN10\_AVI\_01的偏移相加，则得到PN10\_AVI\_01.AV项的信息。

对于功能块类型存在嵌套时，其类型说明部分的功能块子成员信息采用同样的方式，类型中存在4个同名的引脚，但RefID作为区分的标志，其生成规则与变量说明体相同，此处不再赘述。

### 枚举类型的成员变量定义体

枚举类型的成员变量定义体的结构如下所示：

struct SDBEnumVarInfo

{

BYTE btTag; // SDBT\_ENUMVAR 0x06

WORD wSize; // 该变量定义的长度

int iInitValue; // 变量初始值

BYTE btNameLen; // 变量名长度

char szName[]; // 变量名称占ulNameLen字节

}

### 结构体类型的成员变量定义体

结构体类型的成员变量定义体的结构如下所示：

struct SDBStrVarInfo

{

BYTE btTag; // SDBT\_STRVAR 0x07

WORD wSize; // 该变量定义的长度

WORD wTypeId; // 该变量类型在类型表中的索引

BYTE btRefId; // 变量所属变量区

ULONG ulOffset; // 变量偏移

char cProperty; // 变量属性

BYTE btNameLen; // 变量名长度

char szName[]; // 变量名称占ulNameLen字节

}

### 功能块类型的成员变量定义体

功能块类型的成员变量定义体的结构如下所示：

struct SDBFBVarInfo

{

BYTE btTag; // SDBT\_FBVAR 0x08

WORD wSize; // 该变量定义的长度

WORD wTypeId; // 该变量类型在类型表中的索引

BYTE btRefId; // 变量所属变量区

ULONG ulOffset; // 变量偏移

char cProperty; // 变量属性

BYTE btPinType; // 功能块成员所在引脚，1为temp，2

// 为input，3为output，4为inout，

// 5为cpl

ULONG ulPinLocation; // 功能块成员所在的引脚中的位置，原本// 为BYTE型

BYTE btNameLen; // 变量名长度

char szName[]; // 变量名称占ulNameLen字节

}

btPinType和ulPinLocation是为了在还原SDB表时，将功能块类型中的成员变量正确还原到temp、input、output、inout或cpl引脚。

## POU索引信息部分

MACS6.5.1在SDB表文件中的增加POU名及索引号的对应信息，用于工程总控解析POU异常信息。

该部分具体格式如下：

### POU索引信息头（POU Index Header）

POU索引部分头结构如下：

struct SDBPOUIndexHeader

{

ULONG m\_ulTag; //SDBT\_POUTABLE 0x0006

ULONG m\_ulSize; //POU索引信息长度

ULONG m\_ulVersion //目前为0

ULONG m\_ulPOUName //POU错误信息个数

}

### POU索引定义体（POU Info Element）

每个POU的索引信息结构如下：

struct SDBPOUInfo

{

USHORT usSize; //该POU信息长度

USHORT usIndex; //该POU对应索引号

BYTE bNameLen; //该POU名字长度

char szName[]; //该POU名称，名字占usNameLen字节，

}

工程总控通过RTS上传的信息来进行POU运算异常处理报错，该信息目前最多支持16组，每组异常信息由3部分组成，第一个部分为开始的两字节，标示POU异常的错误类型，目前AT支持的错误类型如表3-2所示，后面两字节为发生异常的POU的索引号，最后4字节预留，作为后续报错信息扩展。

表3-2 POU异常信息描述

|  |  |
| --- | --- |
| **错误码** | **错误信息** |
| 1 | 除法、取余分母为零 |
| 2 | 数组下表越界 |
| 3 | 同一语句执行2000次认为是死循环 |
| 4 | log对数操作数为负数 |
| 5 | LN操作数为负数 |
| 6 | 开方操作数为负数 |
| 7 | 指数操作时指数为负数，且底数为零 |

工程总控通过上传信息解析得到POU的异常类型及其对应的索引值，通过查找POU索引定义体得到错误POU的名称，进行界面报错处理。