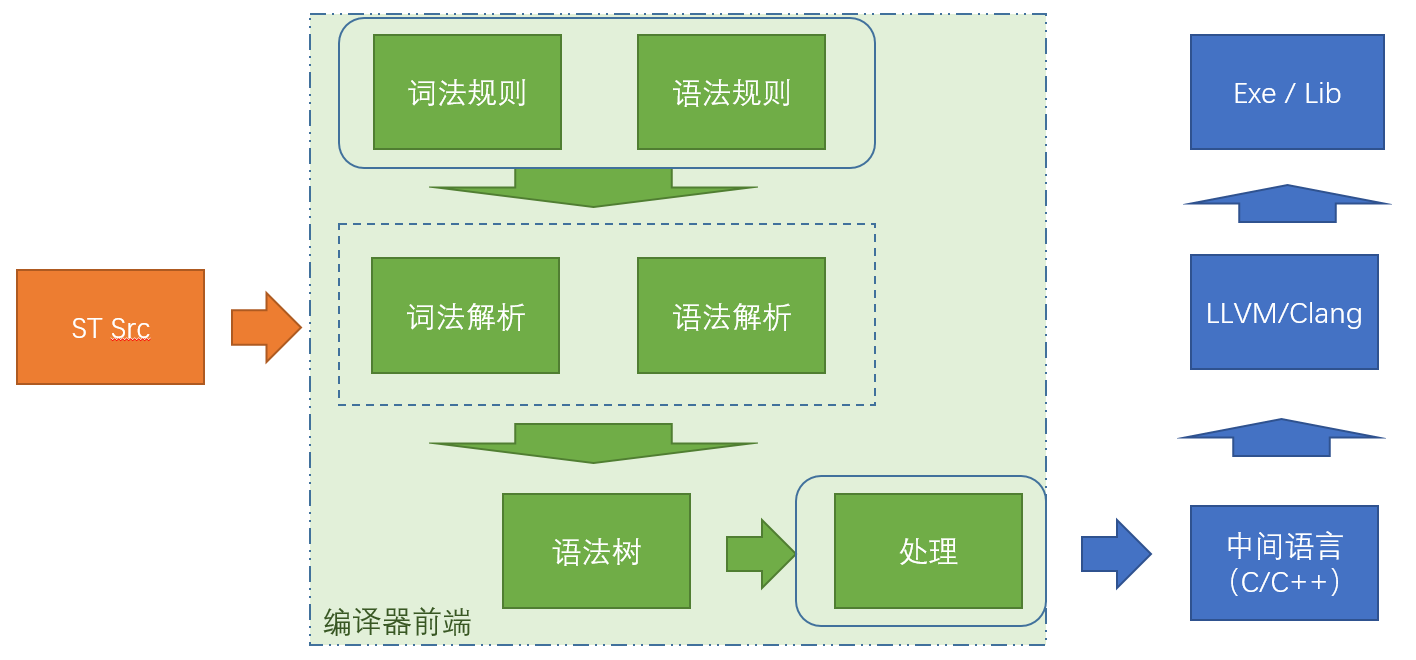
# M7-AT工作交接文档

## ST编译器（编译器前端）

### ST词法&语法解析

### 业务流程：



### 配置信息（设计文档&代码）说明：

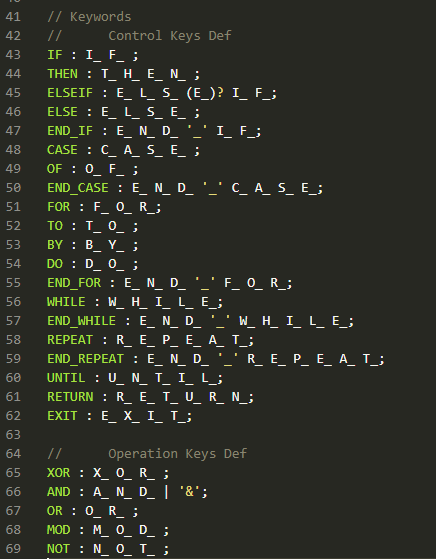
|  |  |  |
| --- | --- | --- |
| 名称 | TFS路径 | 说明 |
| M7产品AT平台ST语言编译器技术选型分析.pptx | $/MACS V7工业自动化平台开发/04\_系统设计/AT子系统设计/AT技术调研文档 | 选型Antlr方案分析文档 |
| antlr4-antlr4-master-4.8-1.zip | $/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/09\_ThirdParty/ANTLR/00\_3rdTools | 源码、工具 |
| 词法&语法解析规则代码 | $/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/09\_ThirdParty/ANTLR/01\_STLanguageRules | HolliAsSTLexer.g4（词法）  HolliAsSTParser.g4（语法）  crtcpp\_st.cmd（代码生成脚本） |
| Antlr4-cpp-runtime引用头文件 | $/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/03\_Platform/00\_Include/00\_Reference/antlr4-cpp-runtime | 使用Antlr4-cpp-runtime所需要引用的头文件 |
| 词法&语法解析功能CPP文件 | $/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/03\_Platform/00\_Include/00\_Reference/iec\_st | 依据词法&语法解析规则，通过生成脚本生成的CPP解析访问节点功能文件 |

### ST词法&语法解析详细说明：

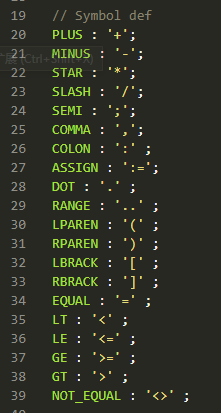
### 词法解析说明：



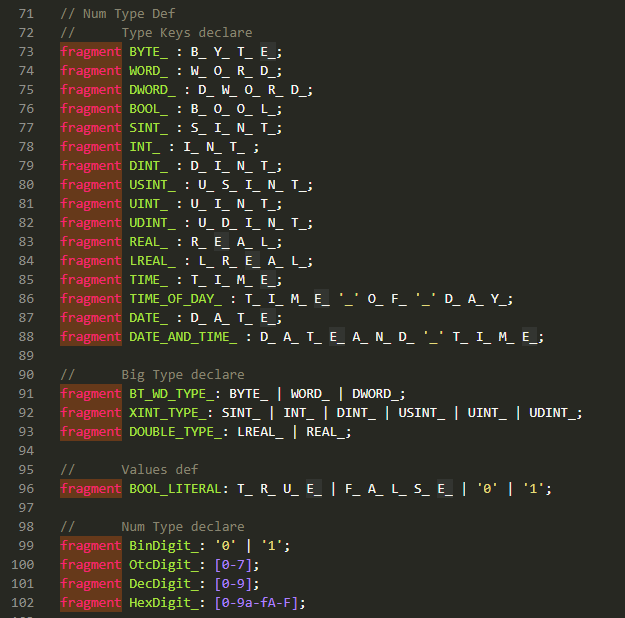
### 关键字词法Token



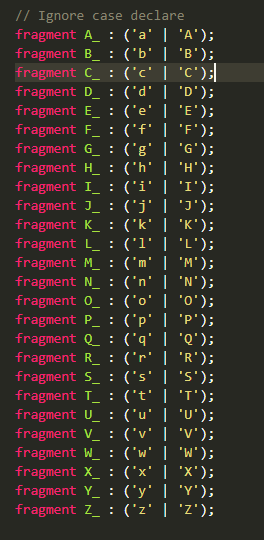
### 操作字符词法Token



### 内部Token：51个（以“fragment”定义，内部使用，外部不可见）

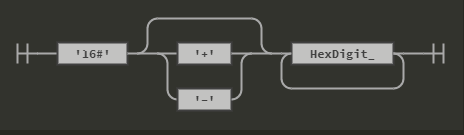


### 通过“fragment A\_ : ('a' | 'A');”方式忽略大小写（ST不区分大小写）

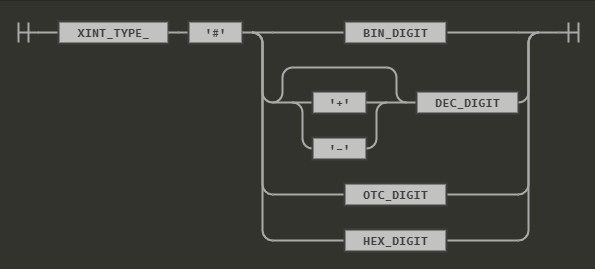


### 常量类型定义词法：

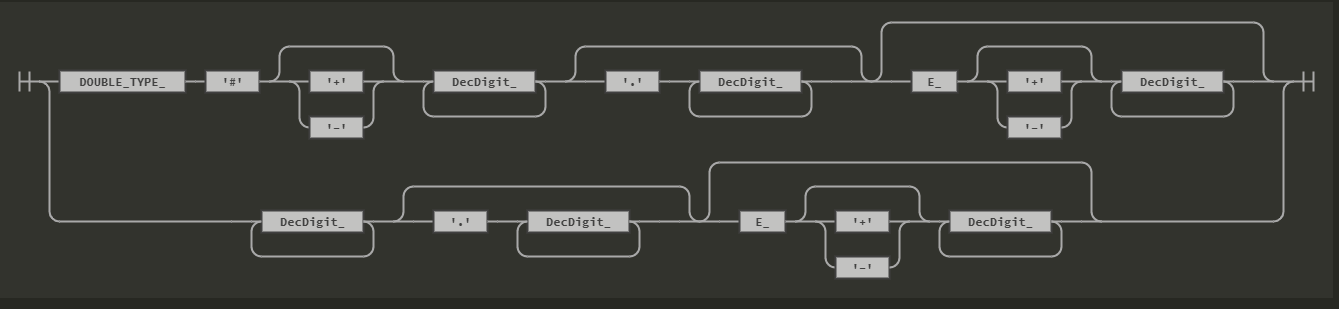
* HEX\_DIGIT：十六进制常量



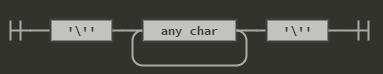
* DEC\_DIGIT：十进制常量
* OTC\_DIGIT：八进制常量
* BIN\_DIGIT： 二进制常量
* CONST\_XWD：类Word（BYTE、WORD、WORD）类型常量
* CONST\_BOOL：布尔常量
* CONST\_XINT：类整形（SINT、INT、DINT、USINT、UINT、UDINT）常量



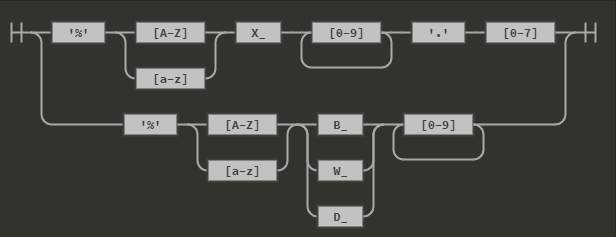
* CONST\_DOUBLE：浮点常量



* CONST\_TIME：时间常量
* CONST\_TIME\_OF\_DAY：TOD常量
* CONST\_DATE：DT常量
* CONST\_DATE\_AND\_TIME：DT类型常量
* CONST\_STRING：字符串常量



* CONST\_DARR：地址常量



### 变量标识符定义

* NET\_IDENTIFIER：网络变量标识符
* NORMAL\_IDENTIFIER：基础标识符

### 注释字符定义（投入Antlr隐藏通道）

* BLOCK\_COMMENT：块注释
* LINE\_COMMENT：行注释

### 非英语字符定义（投入自定义通道2，用于错误处理）

* OTHER\_CHARS

### 其他丢弃字符定义（投入Antlr丢弃通道，语法树不处理）

* WHITESPACE：空格
* LINE\_CONTINUATION：回车、换行、tab
* NEW\_LINE

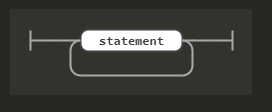
### 语法树节点：26个



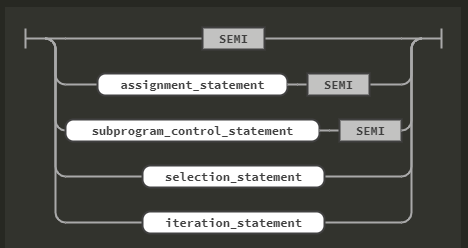
### Prog：语法树顶点节点（根）



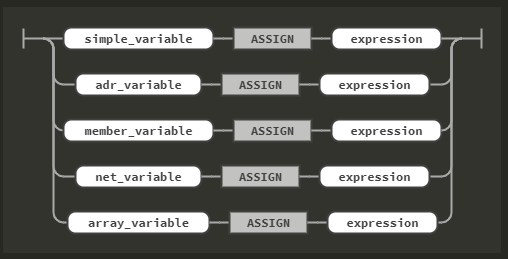
### Statementlist：语法树语句（抽象）列表节点（多语句）



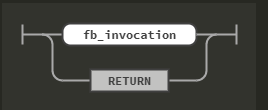
### Statement：语法树语句定义节点列表（类语句类型白名单）



### Assignment\_statement：赋值语句定义（变量:=表达式）



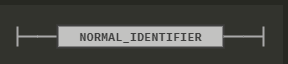
### Subprogram\_control\_statement：函数调用语句定义（Return也是一种特殊函数调用）



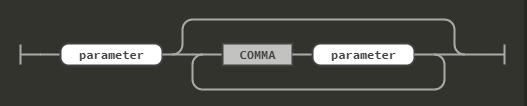
### Fb\_invocation：函数定义



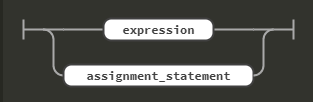
* + - Fb\_name：函数名定义（NORMAL\_IDENTIFIER为基础标识符）



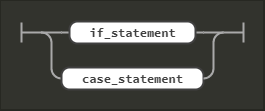
* + - Parameterlist：参数列表定义



* + - Parameter：参数定义



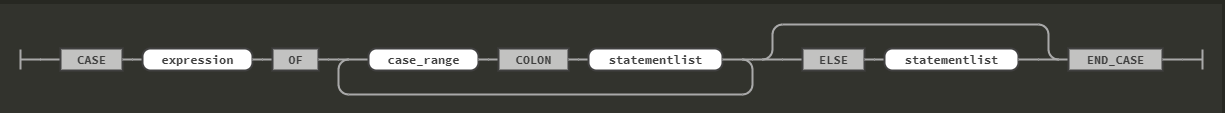
### Selection\_statement：条件语句总类定义（含IF、Case两类）



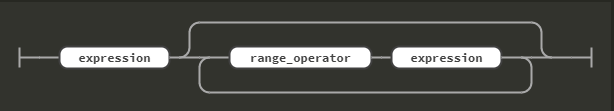
### If\_statement：IF语句条件表达式



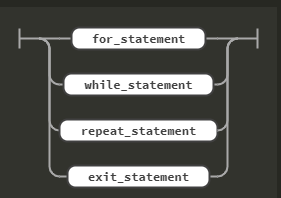
### Case\_statement：Case语句条件表达式



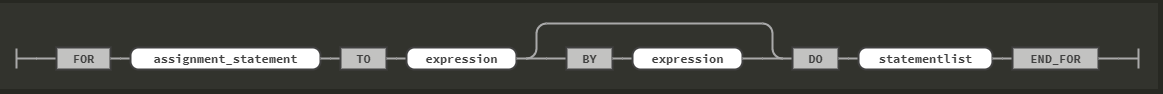
* + - Case\_range：Case条件范围表达式（Range\_operator为范围操作符：“..”、“,”）



### Iteration\_statement：循环语句总类表达式定义（含for、while、repeat、exit子句）



### For\_statement：For循环语句表达式定义



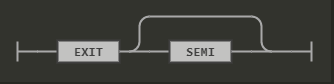
### While\_statement：While循环表达式定义：



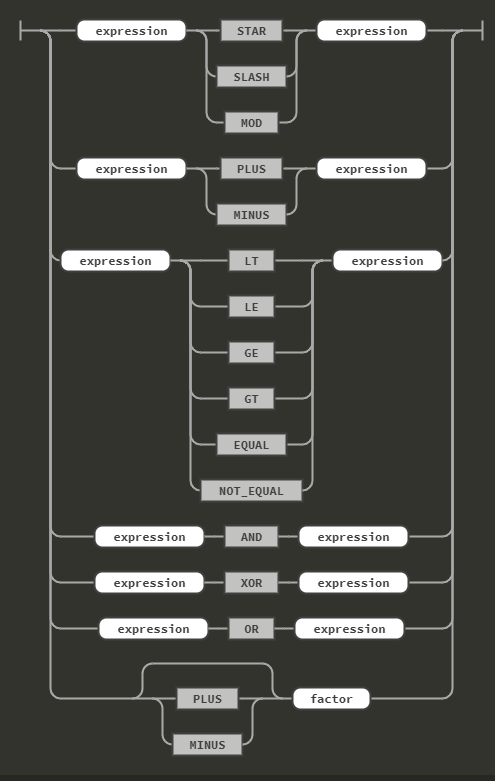
### Repeat\_statement：Repeat循环表达式定义



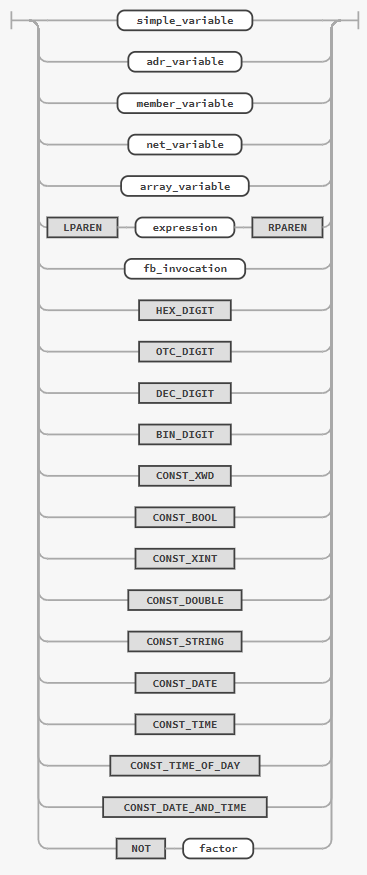
### Exit\_statement：退出语句定义



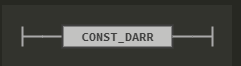
### Expression：表达式定义（递归嵌套）



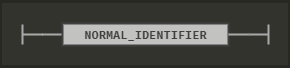
### Factor：表达式基础因子定义



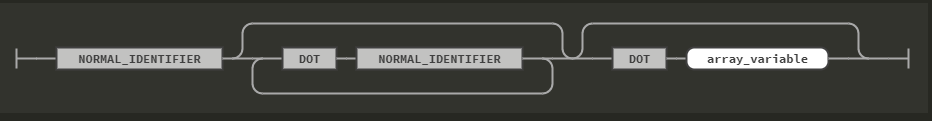
### Adr\_variable：地址变量定义



### Simple\_variable：简单变量定义



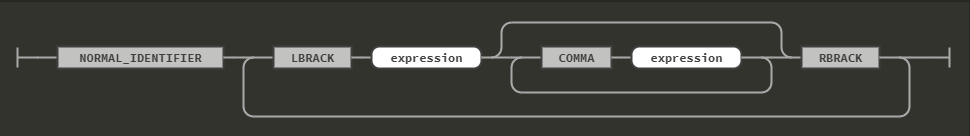
### Member\_variable：成员变量定义



### Net\_variable：网络变量定义

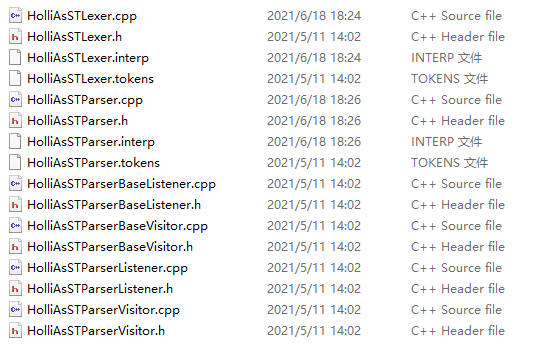


### Array\_variable：数组变量定义



### Antlr-cpp-runtime使用

### 规则生成解析器代码、与访问器代码（使用观察者模式）



### 关键API说明：

m\_pInput = new ANTLRInputStream(m\_sText); //加载待解析文本

m\_pLexer = new HolliAsSTLexer(m\_pInput); //装载词法解析器

m\_pLexer->addErrorListener(m\_pMyErrListener);//加载自定义错误处理类；

m\_pParser = new HolliAsSTParser(m\_pTokens); //装载语法解析器

m\_pParser->addErrorListener(m\_pMyErrListener); //加载自定义错误处理类；

tree::ParseTree\* tree = m\_pParser->prog();//语法解析入口

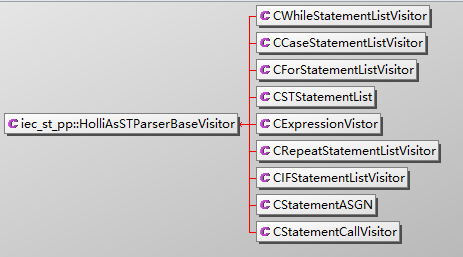
//语法树访问器构建（CSTStatementList继承于HolliAsSTParserBaseVisitor）

CSTStatementList zVisitorv(this);

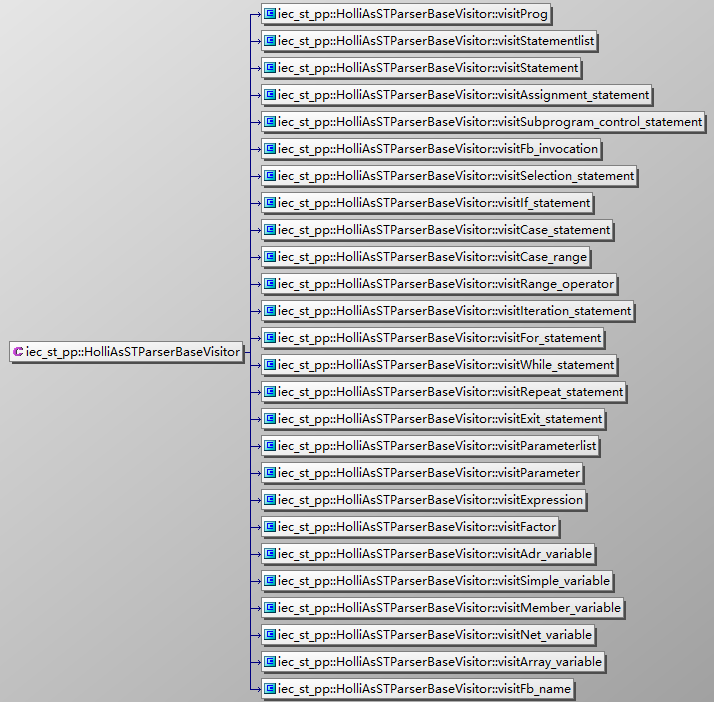
zVisitorv.visit(tree); //语法树访问器绑定语法树

【注】解析详细用法参看CSTPOU::CheckPOUErrors()方法

* 类说明
  + - 右侧为业务具体处理派生类，HolliAsSTParserBaseVisitor为语法树按访问器模式访问基类



* + - 类方法说明（成员访问方法与语法树节点一一对应）：



### 错误处理说明：

自定义错误处理类ice\_st\_err\_listener\_st继承于antlr4::BaseErrorListener，错误分类如下：

* + "missing " + expectedText + " at " + getTokenErrorDisplay(t);

e.g. missing ';' at '<EOF>'

* + "extraneous input " + tokenName + " expecting " + expecting.toString(recognizer->getVocabulary());

e.g. extraneous input 'CASE' expecting OF

* + "rule " + ruleName + " " + e.what();

e.g.

* + "no viable alternative at input " + escapeWSAndQuote(input);

e.g. no viable alternative at input 'g3='

* + "mismatched input " + getTokenErrorDisplay(e.getOffendingToken())

+ " expecting " + e.getExpectedTokens().toString(recognizer->getVocabulary());

e.g. mismatched input ';' expecting

{'+', '-', '(', NOT, HEX\_DIGIT, DEC\_DIGIT, OTC\_DIGIT, BIN\_DIGIT, CONST\_XWD, CONST\_BOOL,

CONST\_XINT, CONST\_DOUBLE, CONST\_TIME, CONST\_TIME\_OF\_DAY, CONST\_DATE, CONST\_DATE\_AND\_TIME,

CONST\_STRING, CONST\_DARR, NET\_IDENTIFIER, NORMAL\_IDENTIFIER}

* + "token recognition error at: '" + getErrorDisplay(text) + "'"

e.g. line 14:0 token recognition error at: '@'

* + "reportAmbiguity d=" + decision + ": ambigAlts=" + conflictingAlts.toString() + ", input='" + text + "'"
  + "reportAttemptingFullContext d=" + decision + ", input='" + text + "'"
  + "reportContextSensitivity d=" + decision + ", input='" + text + "'";

【注】

syntaxError 函数：line 是错误所在行数，charPositionInLine 是所在列，msg 是详细的错误信息，offendingSymbol 是错误出现的 Token

### 外部库

位于$/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/09\_ThirdParty/LLVM/各个平台命名目录/compile/lib/hls子目录，按库路径为目录路径结构划分。

* + - * 各个平台命名目录：arm\_linux、linux、windows
      * 在ExtLibTypeDef.h中定义调用数据结构体（依据TRS提供结构体）
      * 在exlibfun.h中定义映射所需要调用RTS的函数指针，以及在exlibfun.c中初始化指针为空（0），在运行期由编译模块进行初始化，挂载真实函数地址；
      * 每个外部库在系统中对应一个相应的FB或FUN以便上层ST调用；
      * 外部库列表如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **From RTS C File** | | **IEC** | | | | | |
| **RTS\_FUN\_ReturnType** | **FUN\_NAME** | **外部库函数（POU\_Name）** | **return\_type** | **pou\_type** | **C fun\_point\_name** | **功能** | **Lib\_Name** |
| VOS\_VOID | DiagGetCPULoad | CPULoad | NA | FB | fpDiagGetCPULoad | 获取控制器负荷 | HSSyslib |
| VOS\_VOID | CTD | CTD | NA | FB | fpCTD | 递减计数器 | STANDARD |
| VOS\_VOID | CTU | CTU | NA | FB | fpCTU | 递增计数器 | STANDARD |
| VOS\_VOID | CTUD | CTUD | NA | FB | fpCTUD | 递增/递减计数器 | STANDARD |
| VOS\_VOID | DPDiagGetMasterState | DPDiagGetMasterState | NA | FB | fpDPDiagGetMasterState | 获取主卡运行状态 | HSDPdiagEXT |
| VOS\_VOID | F\_TRIG | F\_TRIG | NA | FB | fpF\_TRIG | 下降沿检测触发器 | STANDARD |
| VOS\_VOID | MemLoad | MemLoad | NA | FB | fpMemLoad | 获取内存使用率 | HSSyslib |
| VOS\_VOID | DiagGetNetLoad | NetLoad | NA | FB | fpDiagGetNetLoad | 获取控制器网络负荷 | HSSyslib |
| VOS\_VOID | R\_TRIG | R\_TRIG | NA | FB | fpR\_TRIG | 上升沿检测触发器 | STANDARD |
| VOS\_VOID | RS | RS | NA | FB | fpRS | 复位优先双稳态器 | STANDARD |
| VOS\_VOID | RTC | RTC | NA | FB | fpRTC | 设定时间计时 | STANDARD |
| VOS\_VOID | RTSTIME | RTSTIME | NA | FB | fpRTSTIME | 获取控制器时间 | HSSyslib |
| VOS\_VOID | SR | SR | NA | FB | fpSR | 置位优先双稳态器 | STANDARD |
| VOS\_VOID | SysGetIECInfo | SYSGETIECINFO | NA | FB | fpSysGetIECInfo | 获取IEC详细周期 | HSSyslib |
| uint32\_t | SysGetIECPeriod | SYSGETIECPERIOD | NA | FB | fpSysGetIECPeriod | 获取IEC周期 | HSSyslib |
| uint32\_t | GetUsTimeStamp | SYSGETUSTIMESTAMP | NA | FB | fpGetUsTimeStamp | 获取系统实时US值 | HSSyslib |
| VOS\_VOID | TOF | TOF | NA | FB | fpTOF | 关延时定时器 | STANDARD |
| VOS\_VOID | TON | TON | NA | FB | fpTON | 开延时定时器 | STANDARD |
| VOS\_VOID | TP | TP | NA | FB | fpTP | 普通定时器 | STANDARD |
| VOS\_INT8 | DevGetTaskState | DevGetTaskState | BOOL | FUN | fpDevGetTaskState |  | HSDPdiagEXT |
| VOS\_INT8 | DPDiagGetRawDiag | DPDiagGetRawDiag | BOOL | FUN | fpDPDiagGetRawDiag | 获取模块原始诊断细 | HSDPdiagEXT |
| VOS\_INT8 | DPDiagHartData | DPDiagHartData | BOOL | FUN | fpDPDiagHartData | 获取是否有Hart数据信息 | HSDPdiagEXT |
| VOS\_INT8 | DPGetHartData | DPGetHartData | BOOL | FUN |  | 从DP主卡获取HART数据 | HSDPdiagEXT |
| VOS\_VOID | DPReadDRam | DPReadDRam | BOOL | FUN |  |  | HSDPdiagEXT |
| VOS\_INT8 | DPReadSlaveStationCnetFlags | DPReadDramDPAB | SINT | FUN |  | 从双口RAM中读取当前所组态从站中CNET双网的状态 static VOS\_INT8 DPReadSlaveStationCnetFlags(VOS\_UINT8\* pBuf) 注：此函数未实现 ，需AT明确需求。 | HSDPdiagEXT |
| VOS\_INT8 | DpTestErr | DpTestErr | BYTE | FUN |  |  | HSDPdiagEXT |
| VOS\_UINT8 | GetFCUState | GetFCUState | BYTE | FUN | fpGetFCUState |  | HSSyslib |
| VOS\_INT8 | GetHartData | GetHartData | BOOL | FUN |  | 将Hart仪表数据传递给算法VOS\_INT8 GetHartData(VOS\_UINT8 ucDpMasterIndex, VOS\_UINT8 ucMasterCardType, VOS\_UINT8 ucDevAddr, VOS\_UINT8\* pDATARCV)  注：使用者有问题可以咨询尚文轩工。 | HSDPdiagEXT |
| VOS\_UINT32 | GetIECInterval | GetIECInterval | DWORD | FUN | fpGetIECInterval |  | HSSyslib |
| VOS\_UINT32 | GetIECTaskInterval | GetIECTaskInterval | DWORD | FUN |  | 获取当前系统时间 | HSSyslib |
| VOS\_UCHAR | HSAddP | HSAddP | POINTER TO BYTE | FUN | fpHSADDP |  | HSSyslib |
| VOS\_UCHAR | HSMemcpy | HSMemcpy | BYTE | FUN | fpHSMemcpy |  | HSSyslib |
| VOS\_UINT16 | DiagSetTaskSYN | TaskSYN | WORD | FUN |  | diagnose更新IEC任务同步号函数 | HSSyslib |

## 数据导入&导出框架

### 配置信息



### 数据流定义



### 业务时序定义



### 架构模块定义



* 1. 导入流程定义（内部流）



### 导出流程定义（内部流）



### 第三方库使用

* + 1. rapidxml-1.13：xml解析
    2. antlr4-cpp-runtime：M6变量定义语句解析

### M6导出语法解析规则

* + - 归档路径：

$/MACS\_Platform\_V7/01\_dev/01\_Platform/03\_AT/03\_Platform/15\_ImpExpLib/02\_Source/01\_SrcPaser/01\_M6Parser/02\_M6VarPaser/02\_IECVarAnltrGRuleFile



### 导入导入子系统关键类说明

### CIECXmlIOInterface对外门面接口类

|  |  |
| --- | --- |
| 接口 | 说明 |
| static CIECXmlIOInterface\* GetInstance(); | 对外单例获取接口，此类为导出子系统对外门面接口类 |
| int getPousName(\_IN const char\* strXmlPath, \_OUT std::list<std::string>& lstPouName); | 功能: 获取待导入文件POU名称列表(不支持异步调用扩展)  输入参数:strXmlPath导入XML文件路径  输出参数:lstPouName POU名称列表  返回值: 执行码（0：成功） |
| int impXMLDat(\_IN const char\* strXmlPath, \_IN IMP\_FLG emImpFlg); | 功能: 导入XML(支持异步调用扩展)  输入参数: strXmlPath导入XML文件路径；emImpFlg操作标识码  返回值: 执行码（0：成功） |
| int regSavePouFun(\_IN FP\_SavePouInfFun pCbFun); | 功能: 注册存储模块POU信息组导入接口的接口  输入参数: pCbFu储模块POU信息组导入接口函数指针  返回值: 执行码（0：成功） |
| int expXMLDat(\_IN const void\* pPOUInfos, \_OUT std::string& strXmlPath); | 功能: 导出XML(支持异步调用扩展)  输入参数: pPOUInfos ：导出POU列表  输出参数: strXmlPath：导出XML文件路径；strErrJson : 错误消息JSON结构字符串  返回值: 执行码（0：成功） |

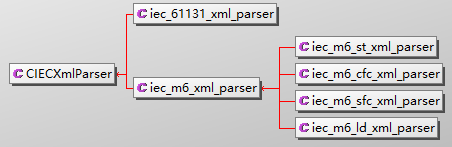
### CIECXmlTskSchMg任务调度类

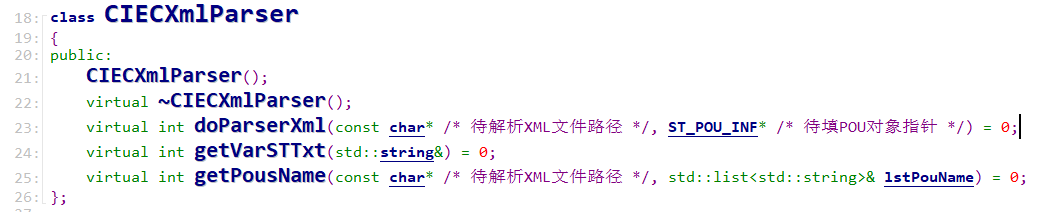
|  |  |
| --- | --- |
| 接口 | 说明 |
| static CIECXmlTskSchMg\* GetInstance(); | 对外单例获取接口 |
| int DoTask(CIECXmlIOTask\*); | 任务处理（线程安全） |

### CIECXmlParserSelector解析器模式选择类

|  |  |
| --- | --- |
| 接口 | 说明 |
| static CIECXmlParserSelector\* GetInstance(); | 对外单例获取接口 |
| CIECXmlParser\* getIecXmlParser(const char\* strXmlPath, int& iExeRtId); | 功能: M7平台XML解析器实例获取，依据XML特征选择相应解析器  输入参数: strXmlPath : 待解析XML文件路径  输出参数:iExeRtId : 执行返回码（成功：0）  返回值: XML解析器实例指针 |

### 业务解析器类





### XML解析器类（观察者模式）





### M6导出对象定义ST语法解析类CIECVarVisitor



### CIECVarDtDef.h内部数据结构定义

|  |  |  |
| --- | --- | --- |
| 数据结构名称 | 成员 | 说明 |
| ST\_POU\_INF：M6 POU导出适配数据结构 | std::string m\_strPouName; | POU名称 |
| std::string m\_strPouPath; | POU路径 |
| std::string m\_strPou2ndName; | POU别名 |
| std::string m\_strPouDesc; | POU描述 |
| int m\_iPouFlg; | POU预留标记 |
| int m\_iPouCyc; | POU周期 |
| int m\_iPouAutoSort; | POU数据流顺序。0:自动；1：手动 |
| std::string m\_strExpTime; | POU导出时间 |
| std::string m\_strAmEndtime; | POU最后修改时间 |
| std::string m\_downloadtime; | POU下装时间 |
| std::string m\_modifier; | 修改人 |
| std::string m\_strPouPaperSize; | 页面尺寸（A[0-4] / B5 / Ax） |
| EM\_POU\_PRT\_TYPE m\_emPouPrintType; | 打印类型（横向：0，纵向：1） |
| int m\_iPouExcuteOrder; | POU执行顺序 |
| ST\_VAR\_DAT\_TYPE m\_stFunRtType; | POU返回类型（函数型POU） |
| ST\_VAR\_GP m\_stVarGp; | POU变量列表 |
| EM\_POU\_LOGIC\_TYPE m\_emPouLogicType; | POU逻辑语言类型（ST \ CFC \ SFC \ LD） |
| ST\_POU\_DATA m\_stPouDat; | POU逻辑语言数据体 |
| ST\_VAR\_DAT\_TYPE  ：POU返回类型（函数型POU） | EM\_VAR\_DAT\_TYPE m\_emVarDtType; | 类型 |
| int m\_iArrCount | 变量长度（数组类型变量元素个数 / 字符串长度）  e.g. [0..3,0..2,0..3]：4\*3\*4 = 48  e.g. p14(2117): STRING(80) -》 80 |
| std::vector<ST\_ARR\_RANGE\*>m\_arrRangeInf; | 数组变量维度上下限列表 |
| std::string m\_strVarUdfName; | 自定义类型名称 |
| EM\_VAR\_DAT\_TYPE m\_emPointer2DtType; | 指向类型  1. Pointer to 类型（非指针类型的其他类型）  2. ARRAY 类型的基础类型（只能为基本类型） |
| EM\_VAR\_DAT\_TYPE m\_emPtArr2DtType | Pointer to ARRAY 的基础类型 |
| std::string m\_strPointer2DtName; | Pointer to 类型名称（为外部定义时必填） |
| ST\_VAR\_GP：POU变量列表数据结构 | std::string m\_strPouName; | pou名称 |
| EM\_PUT\_TYPE m\_emPouType; | pou类型 |
| std::unordered\_map<std::string, ST\_VAR\*> m\_mpVars; | 变量列表(key:变量名），重名检测索引 |
| std::list<std::pair<std::string, ST\_VAR\*>> m\_lstVars; | 保证排序索引 |
| ST\_POU\_DATA：POU逻辑语言数据体 | std::string m\_strStlTxt; | ST语言对象定义文本（FB、FUN、VAR） |
| std::list<CFCElementData\*> m\_lstCFCLogic; | CFC元件定义列表 |

## 增量编译

### 增量POU处理流程



### POU变更处理流程



### 增量C编译处理流程

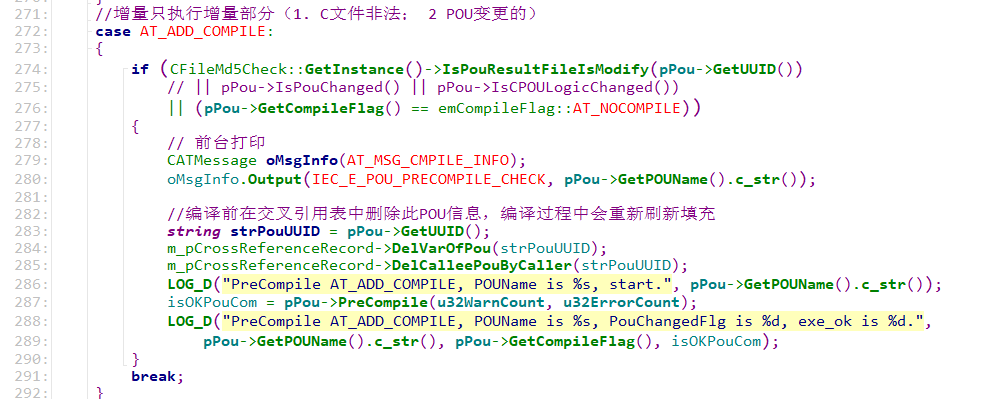


### POU变更判定

* 变量变更：
* 修改：类型，名称，区域，成员（引脚、局部变量）
* 增、删：
* 逻辑变更：
* ST： 待编译语法树变更
* CFC：前台组态非注释元件变更（顺序，名称、增、删、改）

### 关键代码说明

* 增量编译入口：CPOUContainer::PreCompile



* 文件指纹合法检测：

“CFileMd5Check::GetInstance()->IsPouResultFileIsModify(pPou->GetUUID())”接口检测文件

* POU编译标识检测（增量）

(pPou->GetCompileFlag() == emCompileFlag::AT\_NOCOMPILE)

* POU逻辑变更虚接口（各个具体类型POU依赖自己实现）

virtual bool CPOU::IsCPOULogicChanged()

* + CSTPOU::IsCPOULogicChanged()：待编译语法树变更
  + CFC：前台组态非注释元件变更（顺序，名称、增、删、改）
* 变量变更通知接口（参数为变更变量列表）

CPOUContainer::VarUpdateNoticPOUs(map<string, VarUpInfo>& mpVarChangeInfos)



## 远程控制系统

详细信息见系统分析文档：“$/MACS V7工业自动化平台开发/01\_项目管理/09\_项目资料/技术调研报告/远程控制维护系统分析”