**IEC后台处理逻辑**

目录

[公共部分 1](#_Toc42869959)

[变量强制 1](#_Toc42869960)

[异常处理(POUExceptionID枚举类型) 2](#_Toc42869961)

[除零保护 2](#_Toc42869962)

[溢出 3](#_Toc42869963)

[死循环 3](#_Toc42869964)

[Log负数 5](#_Toc42869965)

[Ln负数 5](#_Toc42869966)

[Square负数 5](#_Toc42869967)

[Expt异常 5](#_Toc42869968)

[Asin异常 5](#_Toc42869969)

[Acos异常 5](#_Toc42869970)

[判空处理 5](#_Toc42869971)

[数组下标越界处理 5](#_Toc42869972)

[调度周期 7](#_Toc42869973)

[类型安全 7](#_Toc42869974)

[语言相关部分 7](#_Toc42869975)

[ST语言 7](#_Toc42869976)

[CFC语言 7](#_Toc42869977)

[LD语言 7](#_Toc42869978)

[SFC语言 7](#_Toc42869979)

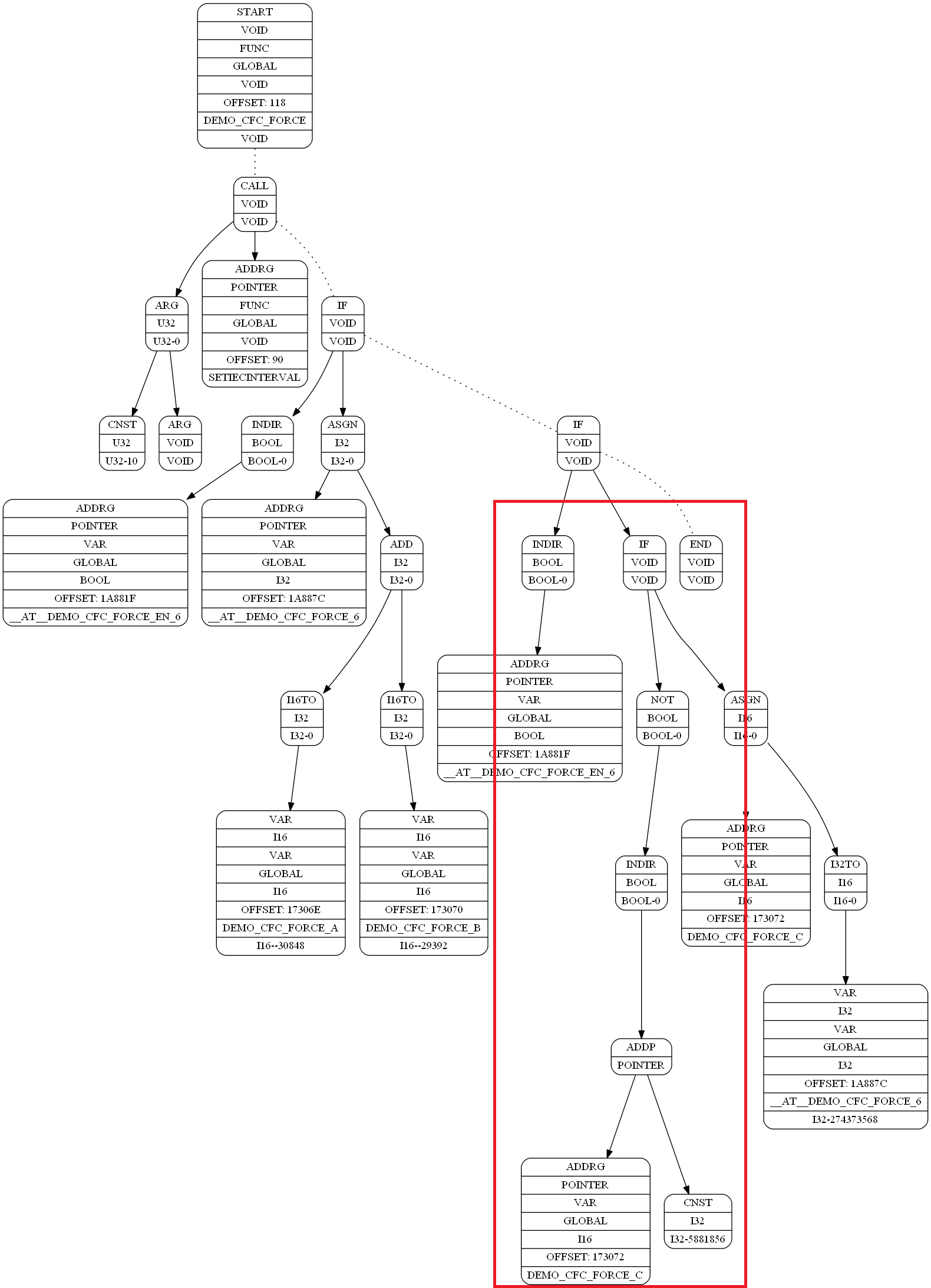
# 公共部分

## 变量强制

**强制变量**生成的HCC中间代码如下图红色框选的部分，要点如下：

1. 在**数据区**的后边，会另外分配一块**和数据去大小相同**的内存，用于存放**强制变量标记**，每个变量都有对应的**强制变量标记**。
2. 如果一个变量可强制，在生成代码时，会**判断强制变量标记**,如果**强制变量标记为FALSE**，**才可以给强制变量赋值**。

**生成代码时，如果一个变量是强制变量，需要生成强制逻辑。**

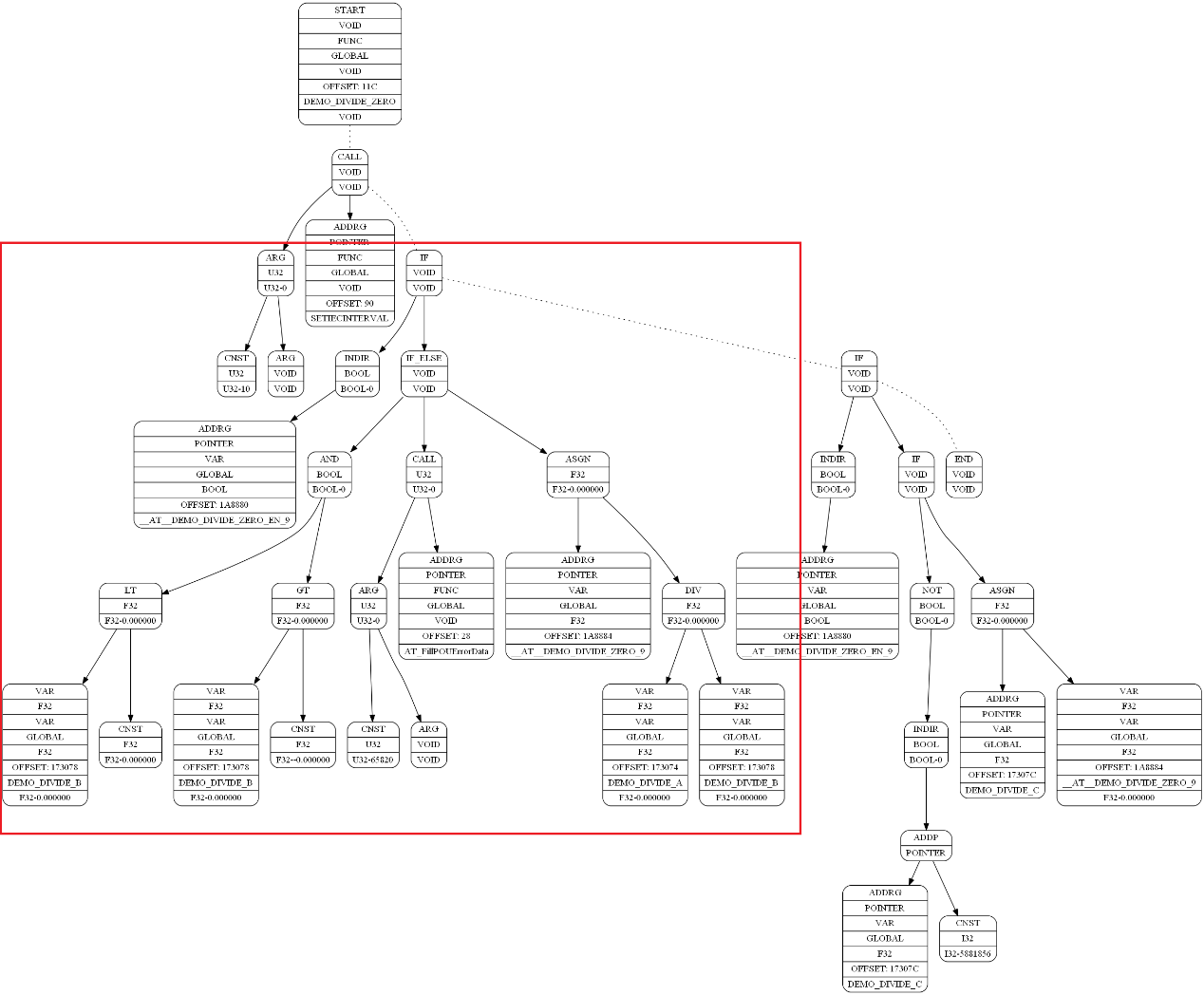


## 异常处理(POUExceptionID枚举类型)

### 除零保护

判断除数是否为0，如果是0，调用系统函数AT\_FillPOUErrorData进行出错处理；否则执行正常的计算逻辑。

**AT\_FillPOUErrorData需要和编译协商看怎么处理。**

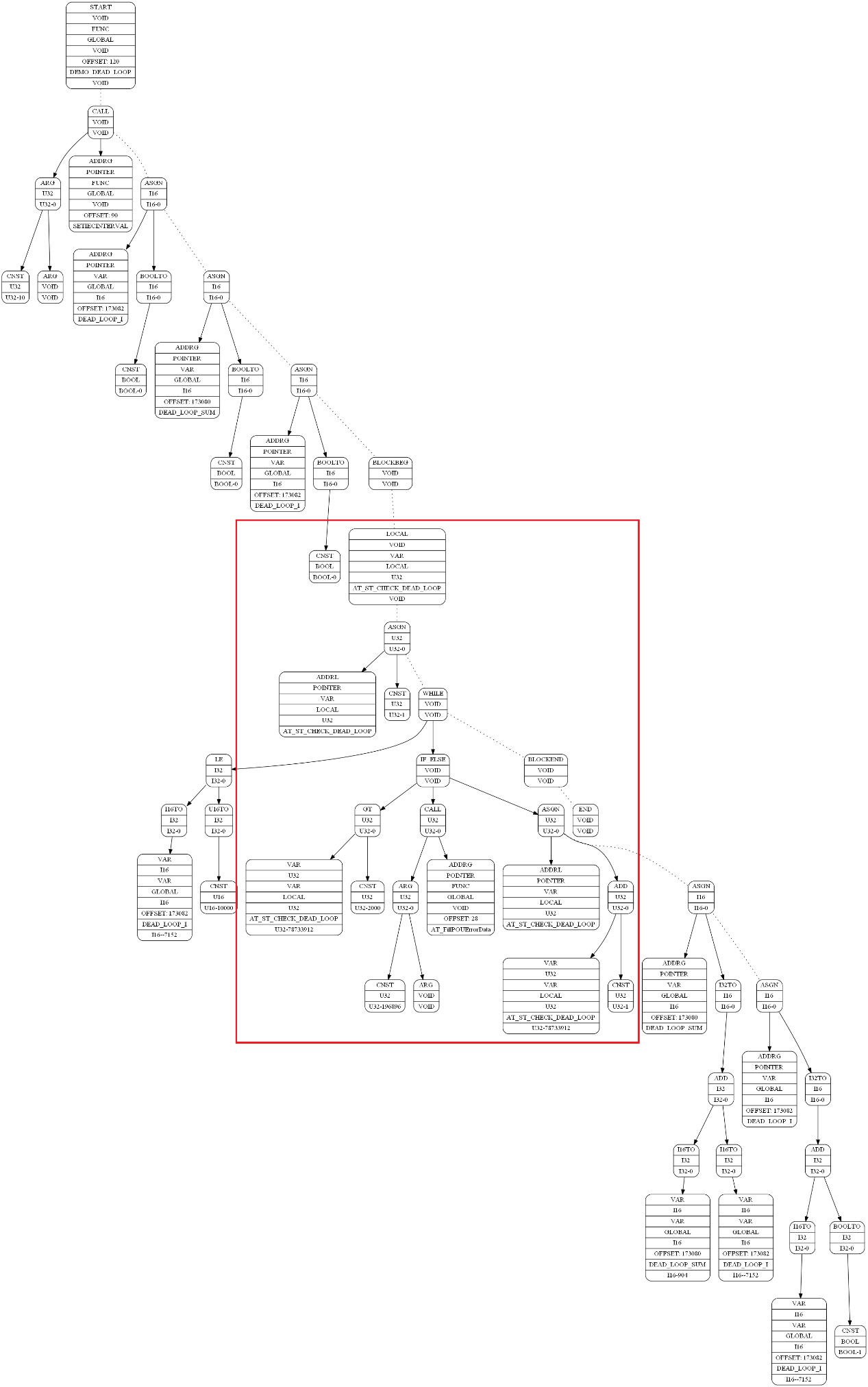


### 溢出

### 死循环

1. 创建一个局部变量AT\_ST\_CHECK\_DEAD\_LOOP用于循环计数。
2. 如果AT\_ST\_CHECK\_DEAD\_LOOP大于**2000**，调用系统函数AT\_FillPOUErrorData进行出错处理。
3. 在每一轮循环，AT\_ST\_CHECK\_DEAD\_LOOP + 1。

**注：目前循环的次数限制为2000，能够满足目前的需求；但是未来在高级算法，云仿真使用时，可能会碰到瓶颈，M7在设计时，需要将这个数值设置为可配置。**



### Log负数

### Ln负数

### Square负数

### Expt异常

### Asin异常

### Acos异常

## 判空处理

## 数组下标越界处理

1. 访问数组时，会生成检查下标的代码。
2. 如果超出数组的LOWER BOUND和UPPER BOUND，调用系统函数AT\_FillPOUErrorData进行出错处理。
3. 否则，执行正常的数组访问。

**ARRAY\_VALUE[ARRAY\_LOOP]:=ARRAY\_LOOP; 生成C代码需要考虑，访问数组都要处理，不能越界。**

enum POUExceptionID

{

    DiV\_By\_0 = 1,

    OverFlow,

    DeadCycle,

    Log\_By\_Neg,

    Ln\_By\_Neg,

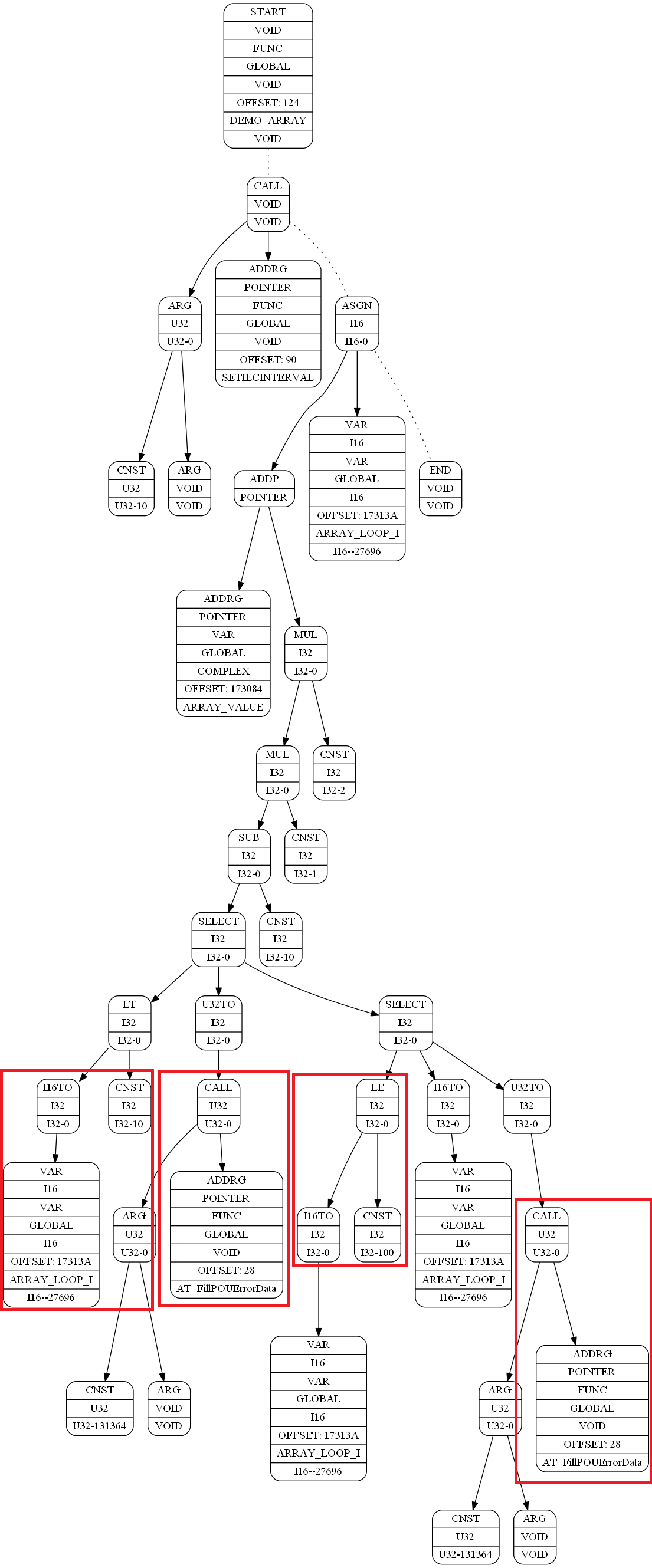
    Square\_By\_Neg,

    Expt\_Exception,

    Asin\_Exception,

    Acos\_Exception

};



## 调度周期

1. 将POU的cycle作为参数。
2. 调用系统函数SETIECINTERVAL。
3. 只有PRG类型的POU生成调度周期代码。**生成C代码时不能漏掉。**

## 类型安全

# 语言相关部分

## ST语言

## CFC语言

## LD语言

## SFC语言

动作