CFC转中间元件方案概述

# 方案概述

语法检查通过后，按元件执行序号逐个元件生成中间语言

# 转中间语言头文件

## 文件名生成

获取POU名称，若不以数字起始，则文件名使用POU名称；若以数字起始，则命名规则为：

POU类型（PRG|FB|FUN）\_POU名称\_POU名称MD5值

注：函数名称与文件名称相同

## PRG头文件生成

1. 生成中间语言函数声明

例：

#ifndef \_\_CFCPRG\_H\_

#define \_\_CFCPRG\_H\_

#include <stdbool.h>

#include <stdint.h>

#include <math.h>

// Prg

#ifdef \_\_cplusplus

extern "C"

{

#endif

void CFCPRG();

#ifdef \_\_cplusplus

}

#endif

#endif

## FB头文件生成

1. 生成中间语言函数声明
2. 生成结构体定义（全部变量）
3. 生成Copy函数定义

例：

#ifndef \_\_CFCFB\_H\_

#define \_\_CFCFB\_H\_

#include <stdbool.h>

#include <stdint.h>

#include <math.h>

// FB

struct CFCFB

{

// IN

int16\_t\* p1;

bool\* p1Force;

// OUT

bool\* p2;

bool\* p2Force;

// INOUT

float\* p3;

bool\* p3Force;

// TEMP

uint16\_t\* p4;

bool\* p4Force;

};

#ifdef \_\_cplusplus

extern "C"

{

#endif

void CFCFB(void\* \_pHead);

#ifdef \_\_cplusplus

}

#endif

// Struct Copy

void CFCFB\_Copy(struct CFCFB\* pSrc, struct CFCFB\* pDesc);

#endif

## FUN头文件生成

1. 生成中间语言函数声明
2. 生成结构体定义（无局部变量，包含返回值变量）
3. 生成Copy函数定义

例：

#ifndef \_\_CFCFUN\_H\_

#define \_\_CFCFUN\_H\_

#include <stdbool.h>

#include <stdint.h>

#include <math.h>

// Fun

struct CFCFUN

{

// IN

bool\* p1;

// INOUT

int16\_t\* p2;

// RETURN

bool\* CFCFUN;

};

#ifdef \_\_cplusplus

extern "C"

{

#endif

bool CFCFUN(void\* \_pHead);

#ifdef \_\_cplusplus

}

#endif

// Struct Copy

void CFCFUN\_Copy(struct CFCFUN\* pSrc, struct CFCFUN\* pDesc);

#endif

# 转中间语言源文件

## FB\_Copy函数生成

判断是否强制，若未强制，则值赋值

例：

void CFCFB\_Copy(struct CFCFB\* pSrc, struct CFCFB\* pDesc)

{

if (!(bool)(\*(pDesc->p1Force))) {

\*(pDesc->p1) = \*(pSrc->p1);

}

if (!(bool)(\*(pDesc->p2Force))) {

\*(pDesc->p2) = \*(pSrc->p2);

}

if (!(bool)(\*(pDesc->p3Force))) {

\*(pDesc->p3) = \*(pSrc->p3);

}

if (!(bool)(\*(pDesc->p4Force))) {

\*(pDesc->p4) = \*(pSrc->p4);

}

}

## FUN\_Copy函数生成

直接值赋值

例：

void CFCFUN\_Copy(struct CFCFUN\* pSrc, struct CFCFUN\* pDesc)

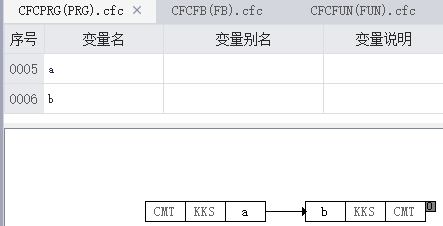
{

\*(pDesc->p1) = \*(pSrc->p1);

\*(pDesc->p2) = \*(pSrc->p2);

}

## PRG源文件生成



按元件执行次序生成各自中间语言

例：

#include "CFCPRG.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

void CFCPRG() {

// 0: Implementation of element b

if(!(bool)((\*(((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_B + H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET )))))){

(\*(((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_B)))) = (\*(((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_A))));

}

}

## FB源文件生成



按元件执行次序生成各自中间语言

例：

void CFCFB(void\* \_pHead) {

// Parameter assignment

struct CFCFB\* pHead = (struct CFCFB\*)\_pHead;

// 0: Implementation of element b

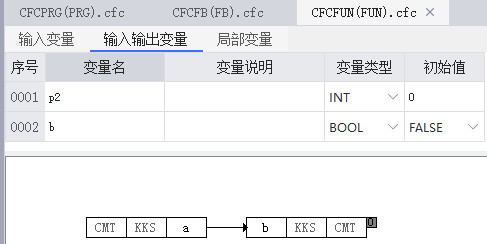
if(!(bool)((\*(((bool\*)(pHead->bForce)))))){

(\*(((bool\*)(pHead->b)))) = (\*(((bool\*)(pHead->a))));

}

}

## FUN源文件生成



按元件执行次序生成各自中间语言

例：

#include "CFCFUN.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

bool CFCFUN(void\* \_pHead) {

// Temp variable definition

int16\_t p3 = 0;

// Parameter assignment

struct CFCFUN\* pHead = (struct CFCFUN\*)\_pHead;

// Return variable assignment

(\*((bool\*)(pHead->CFCFUN))) = false;

// 0: Implementation of element b

(\*(((bool\*)(pHead->b)))) = (\*(((bool\*)(pHead->a))));

// Function return

return (\*((bool\*)(pHead->CFCFUN)));

}

# 结构体变量转中间语言

收集转C过程中用到的结构体变量，在IEC整体预编译完成后，生成结构体变量声明类，文件名命名格式：结构体名称\_Struct.h 结构体名称\_Struct.c，结构体名称同变量类名

例：

#ifndef \_\_TASK\_INFO\_H\_

#define \_\_TASK\_INFO\_H\_

struct TASK\_INFO

{

int32\_t\* dwCount;

bool\* dwCountForce;

int32\_t\* dwCycleTime;

bool\* dwCycleTimeForce;

int32\_t\* dwCycleTimeMin;

bool\* dwCycleTimeMinForce;

int32\_t\* dwCycleTimeMax;

bool\* dwCycleTimeMaxForce;

int32\_t\* dwCycleTimeAvg;

bool\* dwCycleTimeAvgForce;

uint16\_t\* wStatus;

bool\* wStatusForce;

uint16\_t\* wMode;

bool\* wModeForce;

};

struct TASK\_INFO\_TEMP

{

int32\_t dwCount;

int32\_t dwCycleTime;

int32\_t dwCycleTimeMin;

int32\_t dwCycleTimeMax;

int32\_t dwCycleTimeAvg;

uint16\_t wStatus;

uint16\_t wMode;

};

#endif

#include "TASK\_INFO\_STRUCT.h"

void TASK\_INFO\_Copy(struct TASK\_INFO\* pSrc, struct TASK\_INFO\* pDesc) {

if (!(\*((bool\*)pDesc->dwCountForce))) {

pDesc->dwCount = pSrc->dwCount;

}

if (!(\*((bool\*)pDesc->dwCycleTimeForce))) {

pDesc->dwCycleTime = pSrc->dwCycleTime;

}

if (!(\*((bool\*)pDesc->dwCycleTimeMinForce))) {

pDesc->dwCycleTimeMin = pSrc->dwCycleTimeMin;

}

if (!(\*((bool\*)pDesc->dwCycleTimeMaxForce))) {

pDesc->dwCycleTimeMax = pSrc->dwCycleTimeMax;

}

if (!(\*((bool\*)pDesc->dwCycleTimeAvgForce))) {

pDesc->dwCycleTimeAvg = pSrc->dwCycleTimeAvg;

}

if (!(\*((bool\*)pDesc->wStatusForce))) {

pDesc->wStatus = pSrc->wStatus;

}

if (!(\*((bool\*)pDesc->wModeForce))) {

pDesc->wMode = pSrc->wMode;

}

}

void TASK\_INFO\_TEMP\_Copy(struct TASK\_INFO\_TEMP\* pSrc, struct TASK\_INFO\_TEMP\* pDesc) {

pDesc->dwCount = pSrc->dwCount;

pDesc->dwCycleTime = pSrc->dwCycleTime;

pDesc->dwCycleTimeMin = pSrc->dwCycleTimeMin;

pDesc->dwCycleTimeMax = pSrc->dwCycleTimeMax;

pDesc->dwCycleTimeAvg = pSrc->dwCycleTimeAvg;

pDesc->wStatus = pSrc->wStatus;

pDesc->wMode = pSrc->wMode;

}

# AT变量类型与中间语言类型对应表

(DT\_BOOL "BOOL" "bool" "false")

(DT\_STRUCT "STRUCT" "struct" "0")

(DT\_ARRAY "ARRAY" "" "")

(DT\_STRING "STRING" "char" "0")

(DT\_FB "FB" "struct" "")

(DT\_POINTER "POINTER" "" "0")

(DT\_REF "REF" "" "")

(DT\_LREAL "LREAL" "double" "0.00")

(DT\_REAL "REAL" "float" "0.0")

(DT\_DT "DT" "uint32\_t" "0")

(DT\_TOD "TOD" "uint32\_t" "0")

(DT\_DATE "DATE" "uint32\_t" "0")

(DT\_TIME "TIME" "uint32\_t" "0")

(DT\_UDINT "UDINT" "uint32\_t" "0")

(DT\_UINT "UINT" "uint16\_t" "0")

(DT\_USINT "USINT" "uint8\_t" "0")

(DT\_SINT "SINT" "int8\_t" "0")

(DT\_DWORD "DWORD" "uint32\_t" "0")

(DT\_DINT "DINT" "int32\_t" "0")

(DT\_WORD "WORD" "uint16\_t" "0")

(DT\_BYTE "BYTE" "uint8\_t" "0")

(DT\_ENUM "ENUM" "int16\_t" "0")

(DT\_INT "INT" "int16\_t" "0")

(DT\_BIT "BIT" "bool" "0")

(DT\_ZO "ZO" "int8\_t" "0")

# 元件转中间语言

无执行次序的元件，不参与转中间语言，涉及的元件有：输入元件、网络输入元件、注释元件

## Jump元件

## Label元件

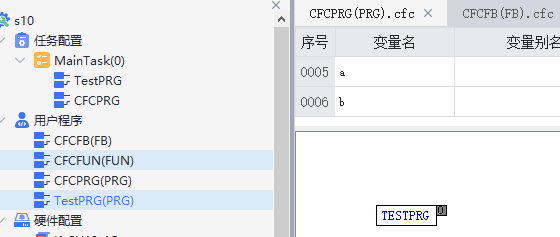
## Return元件

## Output元件

## PRG块元件

1. 添加PRG引用头文件
2. 转C直接调用被调PRG函数名称
3. 记录POU引用关系

例：



#include "CFCPRG.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

#include "TESTPRG.h"

void CFCPRG() {

// 0: Implementation of element TESTPRG

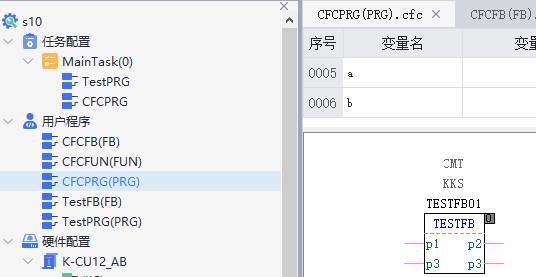
TESTPRG();

}

## FB块元件

### PT\_PRG中调用FB块元件

#### 全局变量



#include "CFCPRG.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

#include "TESTFB.h"

void CFCPRG() {

// 0: Implementation of element TESTFB

struct TESTFB\* TESTFB01 = (struct TESTFB\*)(gl\_ulStactAddress+0);

// Assign address to input variable

TESTFB01->p1 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P1));

TESTFB01->p1Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P1+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to output variable

TESTFB01->p2 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P2));

TESTFB01->p2Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P2+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to inout variable

TESTFB01->p3 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P3));

TESTFB01->p3Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P3+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to temp variable

TESTFB01->p4 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P4));

TESTFB01->p4Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P4+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

bool \_\_AT\_\_CFCPRG\_EN\_1 = true;

if((bool)\_\_AT\_\_CFCPRG\_EN\_1){

// Assign value to input pin variable

TESTFB(TESTFB01);

}

}

#### 局部变量

#include "CFCPRG.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

#include "TESTFB.h"

void CFCPRG() {

// 0: Implementation of element TESTFB

struct TESTFB\* TESTFB02 = (struct TESTFB\*)(gl\_ulStactAddress+0);

// Assign address to input variable

TESTFB02->p1 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P1));

TESTFB02->p1Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P1+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to output variable

TESTFB02->p2 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P2));

TESTFB02->p2Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P2+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to inout variable

TESTFB02->p3 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P3));

TESTFB02->p3Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P3+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to temp variable

TESTFB02->p4 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P4));

TESTFB02->p4Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_CFCPRG\_TESTFB02\_P4+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

bool \_\_AT\_\_CFCPRG\_EN\_1 = true;

if((bool)\_\_AT\_\_CFCPRG\_EN\_1){

// Assign value to input pin variable

TESTFB(TESTFB02);

}

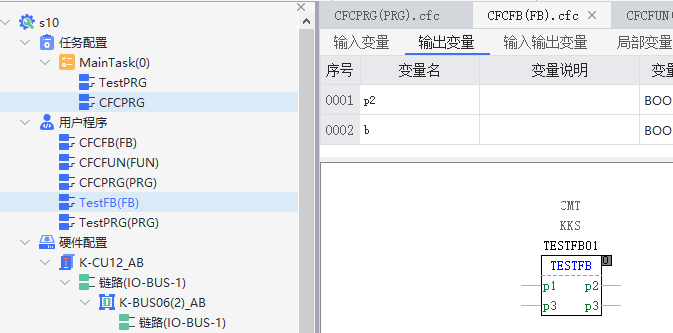
}

### PT\_FB中调用FB块元件

#### 全局变量

通过任务获取全局变量地址，并赋值变量，之后调FB函数

例：



#include "CFCFB.h"

#include "HOLLYSYS-M7-AT-TASK-DEFINE.h"

#include "TESTFB.h"

void CFCFB(void\* \_pHead) {

// Parameter assignment

struct CFCFB\* pHead = (struct CFCFB\*)\_pHead;

// 0: Implementation of element TESTFB

bool \_\_AT\_\_CFCFB\_EN\_1 = true;

if((bool)\_\_AT\_\_CFCFB\_EN\_1){

struct TESTFB\* TESTFB01 = (struct TESTFB\*)(gl\_ulStactAddress+0);

// Assign address to input variable

TESTFB01->p1 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P1));

TESTFB01->p1Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P1+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to output variable

TESTFB01->p2 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P2));

TESTFB01->p2Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P2+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to inout variable

TESTFB01->p3 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P3));

TESTFB01->p3Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P3+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign address to temp variable

TESTFB01->p4 = ((bool\*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P4));

TESTFB01->p4Force = ((bool \*)(gl\_ulDatRangeHeadAdress + H\_LrVarHAdr + H\_\_TESTFB01\_P4+ H\_FrVarHAdr - HD\_EDITABLE\_DATA\_OFFSET ));

// Assign value to input pin variable

TESTFB(TESTFB01);

}

}

#### 局部变量

void CFCFB(void\* \_pHead) {

// Parameter assignment

struct CFCFB\* pHead = (struct CFCFB\*)\_pHead;

// 0: Implementation of element TESTFB

bool \_\_AT\_\_CFCFB\_EN\_1 = true;

if((bool)\_\_AT\_\_CFCFB\_EN\_1){

TESTFB(&(pHead->TESTFB02));

}

}

### PT\_FUN中调用FB块元件

同PT\_FB中调用FB块元件

## FUN块元件

## Operator块元件

# 其它

## FB或FUN中包含复杂变量，转C结构体原则

除了FB和Struct变量外，其它都用指针

例：

struct PRGSTEP

{

// IN

bool\* IN;

bool\* INForce;

bool\* FBK;

bool\* FBKForce;

bool\* JUMP;

bool\* JUMPForce;

bool\* STRFAIL;

bool\* STRFAILForce;

bool\* RUN;

bool\* RUNForce;

bool\* NRST;

bool\* NRSTForce;

bool\* INJUMP1;

bool\* INJUMP1Force;

bool\* WT;

bool\* WTForce;

// OUT

bool\* LRST;

bool\* LRSTForce;

bool\* RUNOUT;

bool\* RUNOUTForce;

bool\* FALOUT;

bool\* FALOUTForce;

uint8\_t\* STEPOUT;

bool\* STEPOUTForce;

bool\* ENDOUT;

bool\* ENDOUTForce;

bool\* JMPOUT;

bool\* JMPOUTForce;

// INOUT

// TEMP

bool\* INJUMP2;

bool\* INJUMP2Force;

uint32\_t\* CYC;

bool\* CYCForce;

uint8\_t\* BB;

bool\* BBForce;

struct RS RS1;

struct RS RS2;

struct RS RS3;

struct RS RS4;

struct HSTON TON1;

};

// FB

struct PRGSTART

{

// IN

bool\* FBK;

bool\* FBKForce;

bool\* PEN;

bool\* PENForce;

bool\* MAN;

bool\* MANForce;

bool\* AUT;

bool\* AUTForce;

bool\* MOFFRST;

bool\* MOFFRSTForce;

bool\* AOFFRST;

bool\* AOFFRSTForce;

bool\* FALIN;

bool\* FALINForce;

uint8\_t\* OUTAD;

bool\* OUTADForce;

bool\* INJUMP;

bool\* INJUMPForce;

// OUT

bool\* POUT;

bool\* POUTForce;

bool\* OUT;

bool\* OUTForce;

bool\* FALOUT;

bool\* FALOUTForce;

bool\* OUTJMP;

bool\* OUTJMPForce;

// INOUT

// TEMP

struct RS RS1;

struct RS RS2;

struct RS RS3;

struct HSTP VOPTP;

struct HSTON OTTON;

struct HSTP TP1;

struct HSTP TP2;

};

struct NWRITERAWDIAG\_NM

{

uint8\_t \*DiagAdd;

bool\* DiagAddForce ;

struct STCSLDIAGNOR SlDiag;

bool\* SlDiagForce ;

uint8\_t\* devDiag;

bool\* devDiagForce ;

uint8\_t\* modDiag;

bool\* modDiagForce ;

uint16\_t\* bModeNo;

bool\* bModeNoForce ;

uint8\_t\* bChErr;

bool\* bChErrForce ;

uint8\_t\* bErr;

bool\* bErrForce ;

uint8\_t\* NWRITERAWDIAG\_NM;

};