

Simulink代码生成提高教程

原创 思想Goodman 古德曼汽车工业 2019-09-09

收录于话题 #车辆控制工程

16个

点击蓝字

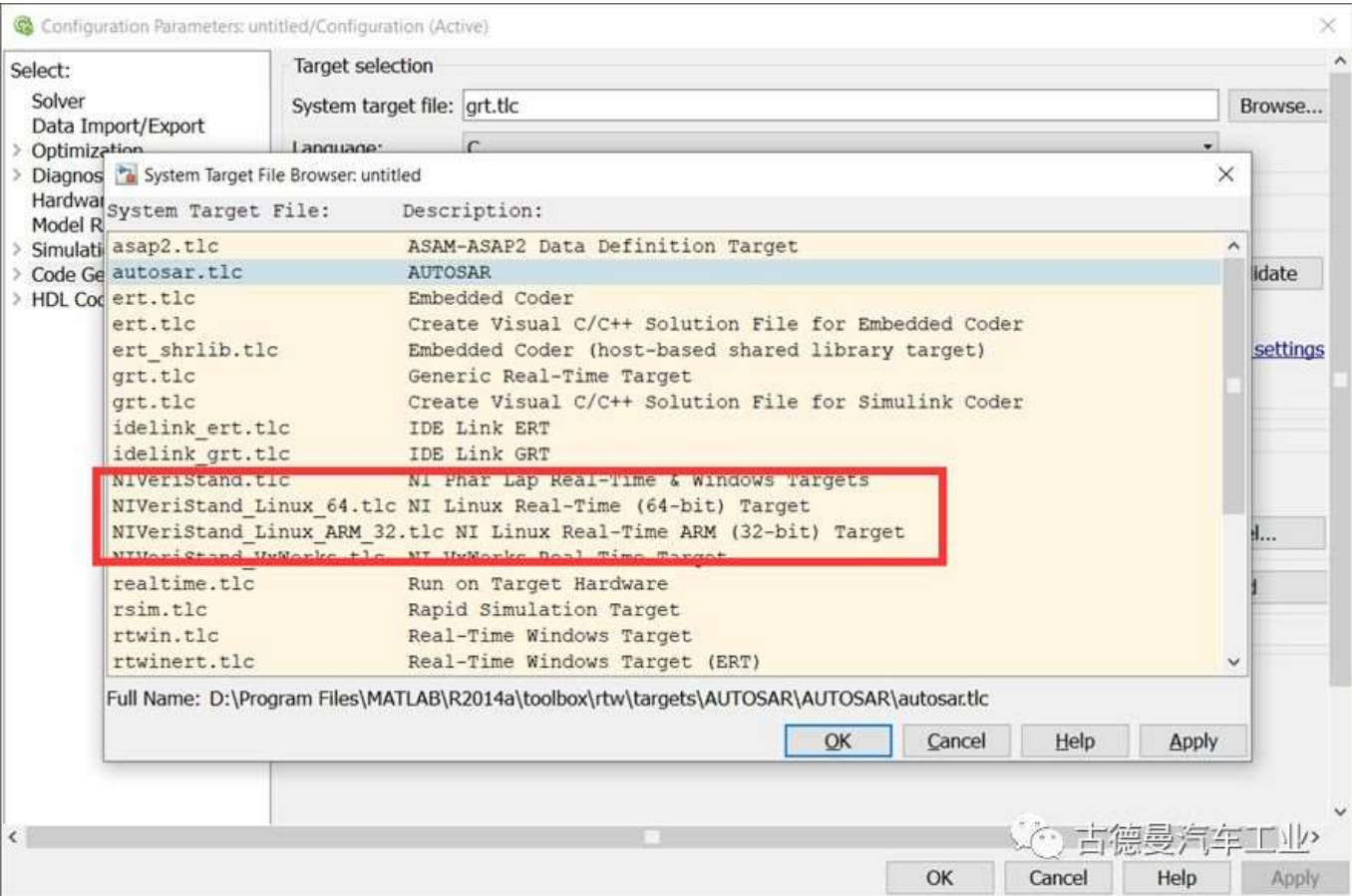
关注我们

01 前言

前两期文章《Simulink代码生成应用教程》、《Sinmulink代码生成基础体验教程》中思想介绍了如何使用Simulink生成嵌入式代码，以及生成的代码如何移植到嵌入式开发环境。本期文章内容将继续深入，介绍下如何在Simulink中直接对嵌入式芯片配置，做到无需手工编写底层的嵌入式程序。市场上的这类产品一般由快速原型的厂家（如：华海科技、海博瑞德等）提供，需要安装厂家自己的目标系统，本期就来聊聊如何自制一个目标系统。

02 什么是自定义目标系统

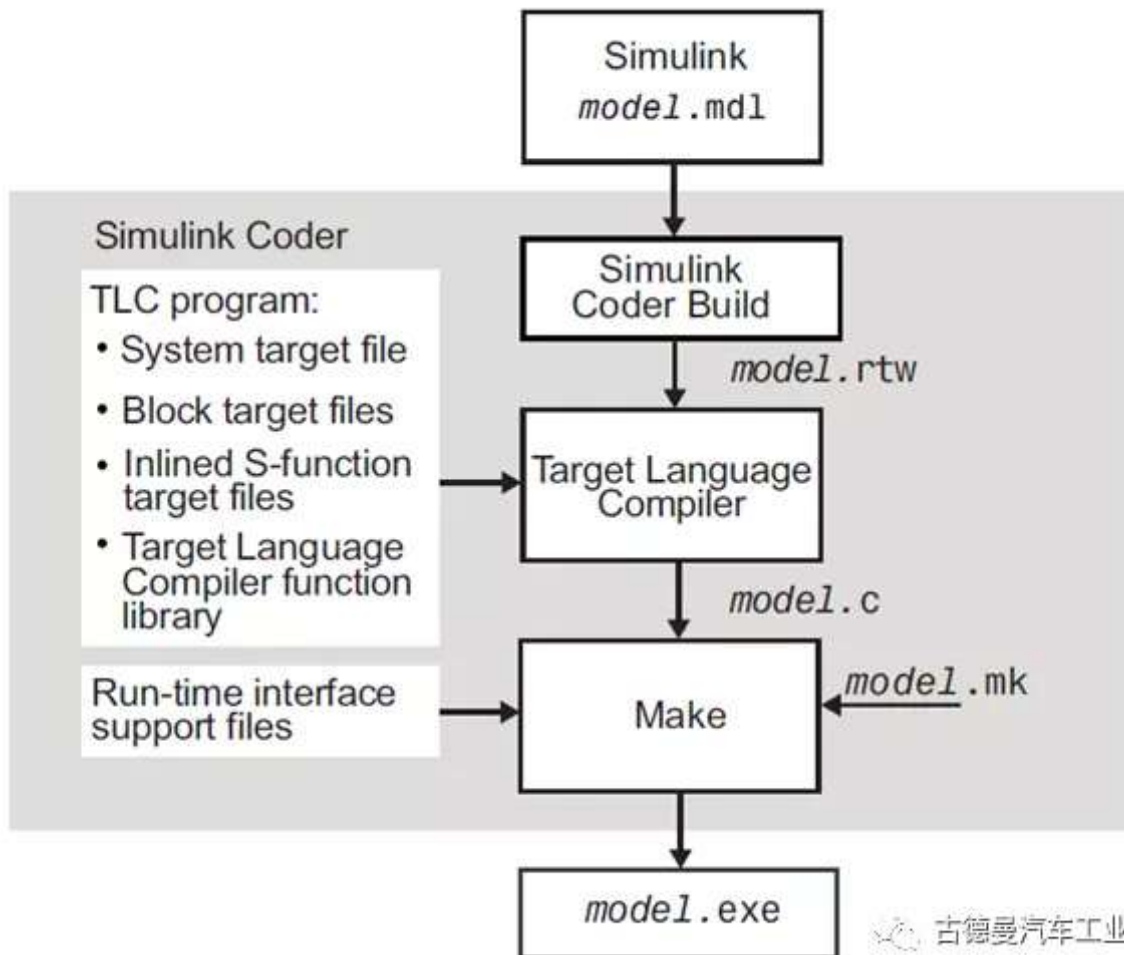
在前面介绍代码生成的时候，第一步就是选择目标【Target Selection】，Matlab已经自带了不少目标系统，例如之前用的ert.tlc。



如果你安装了NIVeriStand、CRUISE或其他快速原型产品，这里就会多出来一些目标系统可以选择。自定义目标系统是为了让Simulink生成的代码能根据用户的需要，与底层驱动做集成。

03 Simulink目标编译流程

制作自定义目标系统前，需要了解Simulink目标编译的流程，该部分内容来自Matlab内部帮助文件《Simulink Coder Target Language Compiler》。之前的内容中介绍了如何将Simulink模型生成C代码，下面思想就来围观一下，当我们按下【编译】按钮后Matlab究竟做了哪些事。



- 第一步，Simulink会生成一个rtw文件，这个文件非常重要。它保存了Simulink模型中的所有信息，包含模块，模型配置、S函数等。
- 第二步，使用TLC语言，读取rtw文件中的信息，根据预先编写好的模板重新组织生成的C语言代码。通过这种方式就能获得与嵌入式开发环境兼容的代码。
- 第三步，依托编译器（C/C++），将C代码编译成exe或者dll文件。这一步只会在一些仿真软件中用到，例如NIVeriStand、dSpace、CRUISE。本次内容只涉及到C代码生成，所以这部分内容不过多介绍。

```







D: > 05-SLX > slx > mygt > example > sfun_generation_7_mygt_rtw > sfun_generation_7.rtw
1  CompiledModel {
2      Name      "sfun_generation_7"
3      OrigName   "sfun_generation_7"
4      Version    "9.0 (R2018b) 24-May-2018"
5      SimulinkVersion "9.2"
6      ModelVersion "1.81"
7      GeneratedOn  "Mon Sep  2 22:46:19 2019"
8      HasSimStructVars  0
9      HasCodeVariants  0
10     HasInlineVariants  0
11     PreserveExternInFcnDecls 1
12     DataDictionary    ""
13     DataDictionarySet ""
14     ParallelExecutionInRapidAccelerator "0"
15     ParametersInRTMForMultiInstanceERT 1
16     ObserversInstrumentationInjection 0

```

上面是一个rtw文件，该文件会在编译过程中出现在代码生成的文件夹里。Matlab默认情况下这个文件在完成代码生成的时候会自行了断（被删除），如果有小伙伴们对rtw文件结构感兴趣，可以在Simulink模型配置中，把【Retain Rtw】选项勾选，这样完成代码生成动作后，这个文件就不会被删除。

04 自定义目标系统的文件组成

假设我们制作的目标系统名称为mygt，那就会包含如下文件：

Name	Date modified
 mygt.tlc	2018/1/18 9:35
 mygt.tmf	2018/1/18 9:59
 mygt_callback_handler.m	2018/1/18 9:47
 mygt_file_process.tlc	2018/1/19 13:10
 mygt_make_rtw_hook.m	2018/1/26 8:33
 mygt_stm32F407_srmain.tlc	2018/1/26 8:33

- 系统TLC文件【mygt.tlc】
- 主函数TLC文件【mygt_stm32F407_srmain.tlc】
- 用于处理文件的TLC文件【mygt_file_process.tlc】
- 回调函数文件【mygt_callback_handler.m】
- hook文件【mygt_make_rtw_hook.m】
- 编译模板【mygt.tmf】

每一个目标系统除了代码生成模板的设置外还会封装自己的Simulink模块。每个模块都会包含以下文件：

Name	Date modified	Type	Size
mcu_can_config.c	2018/1/26 13:37	C Source File	4 KB
mcu_can_config.mexw32	2018/1/30 15:35	MEXW32 File	93 KB
mcu_can_config.mexw64	2018/1/31 14:15	MATLAB MEX	19 KB
mcu_can_config.tlc	2018/1/30 13:22	TLC File	2 KB

- 编译前的C mex S函数文件【mcu_can_config.c】
- 编译后的S函数文件【mcu_can_config.mexw32】
- 模块的TLC文件【mcu_can_config.tlc】

05 系统TLC文件

刚刚提到了，生成代码的第一步就是配置目标【Target selection】，所以创建自定义目标的第一步也就是设计一个自己的tlc模板。这个模板可以参考ert.tlc来创建。

```
%% SYSTLC: This Is My Target TMF: none MAKE: make_rtw EXTMODE: ext_comm
```

```
%selectfile NULL_FILE
%assign CodeFormat = "Embedded-C"
%assign TargetType = "RT"
%assign Language = "C"
%assign AutoBuildProcedure = !GenerateSampleERTMain
%include "codegenentry.tlc"
```

```
/%
BEGIN_RTW_OPTIONS
rtwgensettings.BuildDirSuffix = '_mygt_rtw';
rtwgensettings.DerivedFrom = 'ert.tlc';
rtwgensettings.Version = '1';
rtwgensettings.SelectCallback = ['mygt_callback_handler(hDlg, hSrc)'];
END_RTW_OPTIONS
%/
```

古德曼汽车工业

系统TLC文件分为三个部分，顶部定义了自定义目标系统的名称，我们就叫它【This is My Target】。后面是编译选项，由于本次只生成C代码，所以编译模板选择【none】。中部的代码定义了生成代码的类型、语言等，直接照抄ert.tlc。底部定义了代码生成的文件夹命名格式、rtw设置直接引用ert.tlc、设置回调函数等。

06 回调函数

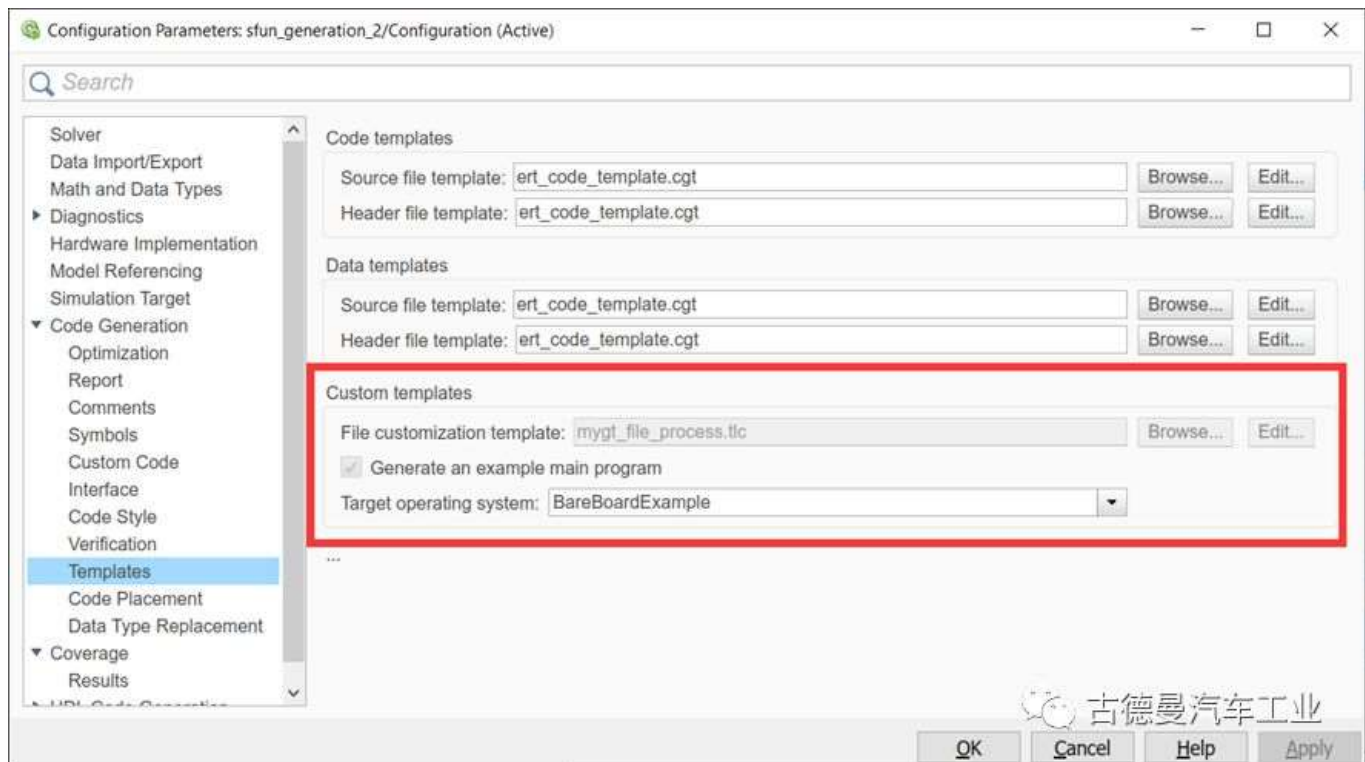
当我们在Target Selection中选择了系统TLC之后，会调用这个函数，对目标系统的常见配置进行初始化。


```

1 function mygt_callback_handler(hDlg, hSrc)
2
3 % Setup these options as desired and gray them out
4 slConfigUISetVal(hDlg, hSrc, 'GenerateSampleERTMain', 'on');
5 slConfigUISetEnabled(hDlg, hSrc, 'GenerateSampleERTMain', 0);
6
7 slConfigUISetVal(hDlg, hSrc, 'GenerateMakefile', 'off');
8 slConfigUISetEnabled(hDlg, hSrc, 'GenerateMakefile', 0);
9
10 slConfigUISetVal(hDlg, hSrc, 'ERTCustomFileTemplate', 'mygt_file_process.tlc');
11 slConfigUISetEnabled(hDlg, hSrc, 'ERTCustomFileTemplate', 0);

```

例如这三部分分别为：设置ERT生成主函数、设置不进行编译、设置用户代码模板



上述回调函数对应到模型的参数配置中就是上图红色部分。当然你可以通过Matlab帮助找到你想设置参数的变量名。

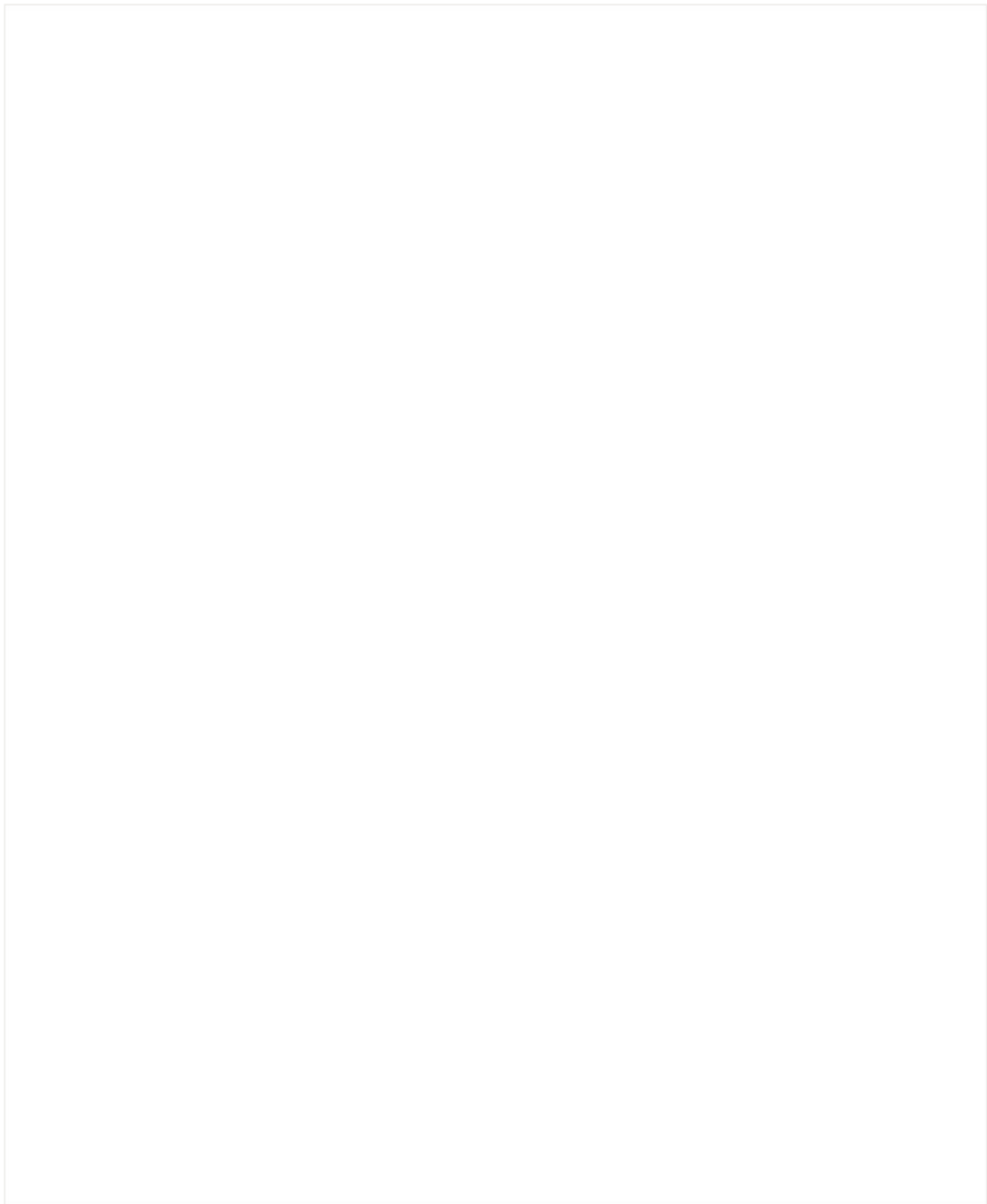
07 用于操作文件的TLC文件

该文件在本例中命名为mygt_file_process.tlc，通过这个文件来选取使用哪个代码模板。

```
1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  %% $RCSfile: mygt_file_process.tlc,v $
3  %% $Revision: 1.1.6.3 $
4  %% $Date: 2006/10/10 02:35:49 $
5  %%
6  %% Abstract:
7  %%   Example Real-Time Workshop Embedded Coder custom file processing template.
8  %%
9  %% Copyright 1994-2006 The MathWorks, Inc.
10 %%
11 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
12 %selectfile NULL_FILE
13
14 %assign ERTCustomFileTest = TLC_TRUE
15
16 %if EXISTS("ERTCustomFileTest") && ERTCustomFileTest == TLC_TRUE
17
18     %% Need to set the template compliance flag before you can use the API
19     %<LibSetCodeTemplateComplianceLevel(1)>
20     %include "mygt_stm32F407_srmain.tlc"
21     %<FcnSingleTaskingMain()>
22 %endif
```



上图可知，本目标系统将调用mygt_stm32F407_srmain.tlc这个代码模板文件，这是一个STM32的Keil工程模板。如果你的项目使用的是CodeWarrior，就需要另外制作一个模板。



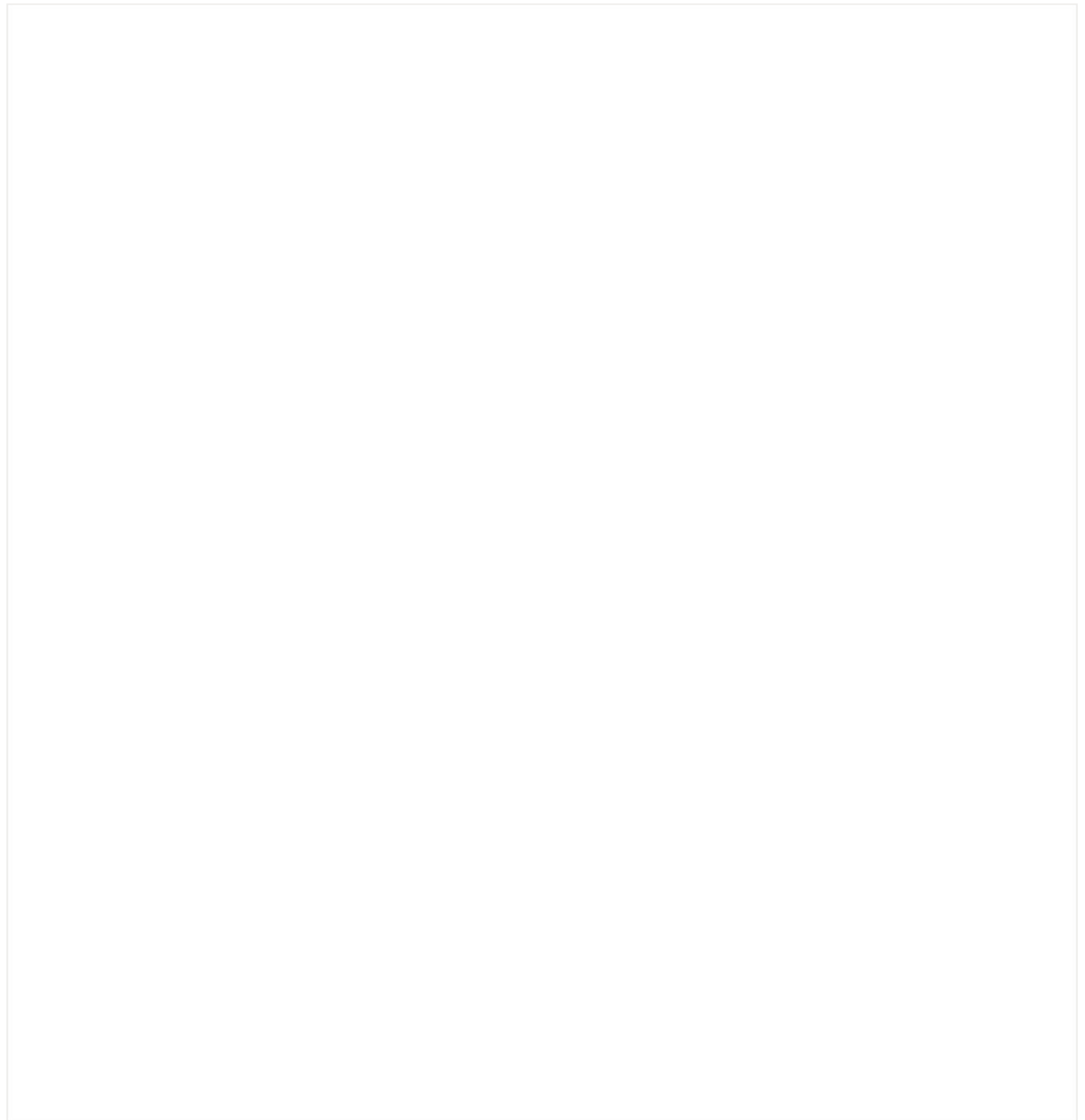
打开mygt_stm32F407_srmain.tlc我们可以看到这个文件会把生成的代码组织成一个keil工程的主.c格式。图中包含了需要引用的头文件，函数等等。这里用到很多tlc函数，这些函数具体什么可以去《Simulink Coder Target Language Compiler》查找。

08 Hook文件

顾名思义就是一个钩子函数，它用来将代码生成的各阶段中断出来，插入自己想要加入的操作。



通过流程图，我们可以在每一个步骤进入前【before】、进入时【entry】、完成后【after】进行自定义操作。



这就是一个Hook文件的结构，最常用就是在【exit】的时候，我们把生成的代码与IDE（集成开发环境）做集成。完成代码生成后可以直接用ukeil打开工程，甚至直接烧写单片机。这里调用了make_exit_hook函数。



这内容有点复杂，大致逻辑就是将生成的C文件与H文件复制到keil工程目录下，并把生成C文件、H文件写入工程描述中。

09 模块文件

上面主要介绍的是自定义目标系统的工作流程、包含哪些文件、具体有什么用。那大部分的自定义目标系统都会封装自己的Simulink模块库，用来对单片机的硬件进行配置



例如：本次使用的例子是基于STM32F407芯片，所以就用S函数封装了一个CAN的初始化函数。对应的S函数名称就是MCU_Can_Config.mwx64。这部分如何创建S函数，调用S函数，Mask参数的设计。这部分在之前的《S-Funciton应用实例》中已经介绍过，这里不重复。

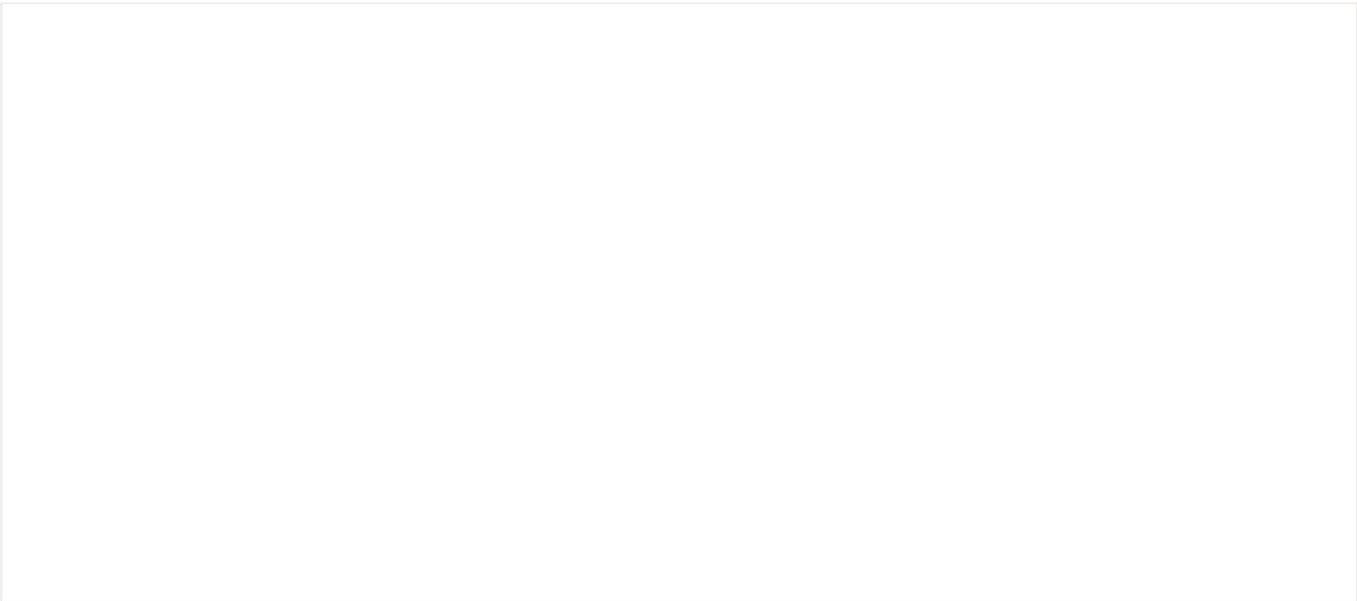


之前介绍S函数时没有介绍mdlRTW这个函数，该函数的功能是把S函数的参数、端口等信息写入RTW文件。后面通过模块的TLC文件来读取写入RTW文件中的相关信息



在模块TLC文件中，定义该模块生成代码的格式，包含引用的外部函数，申明变量。另外这里的TLC函数还会到RTW文件中读取配置参数，配合模板生成代码。

10 大功告成



这是一个利用自定义目标系统开发的应用模型，使用了两个封装模块，分别用来配置CAN总线和接收CAN总线报文。



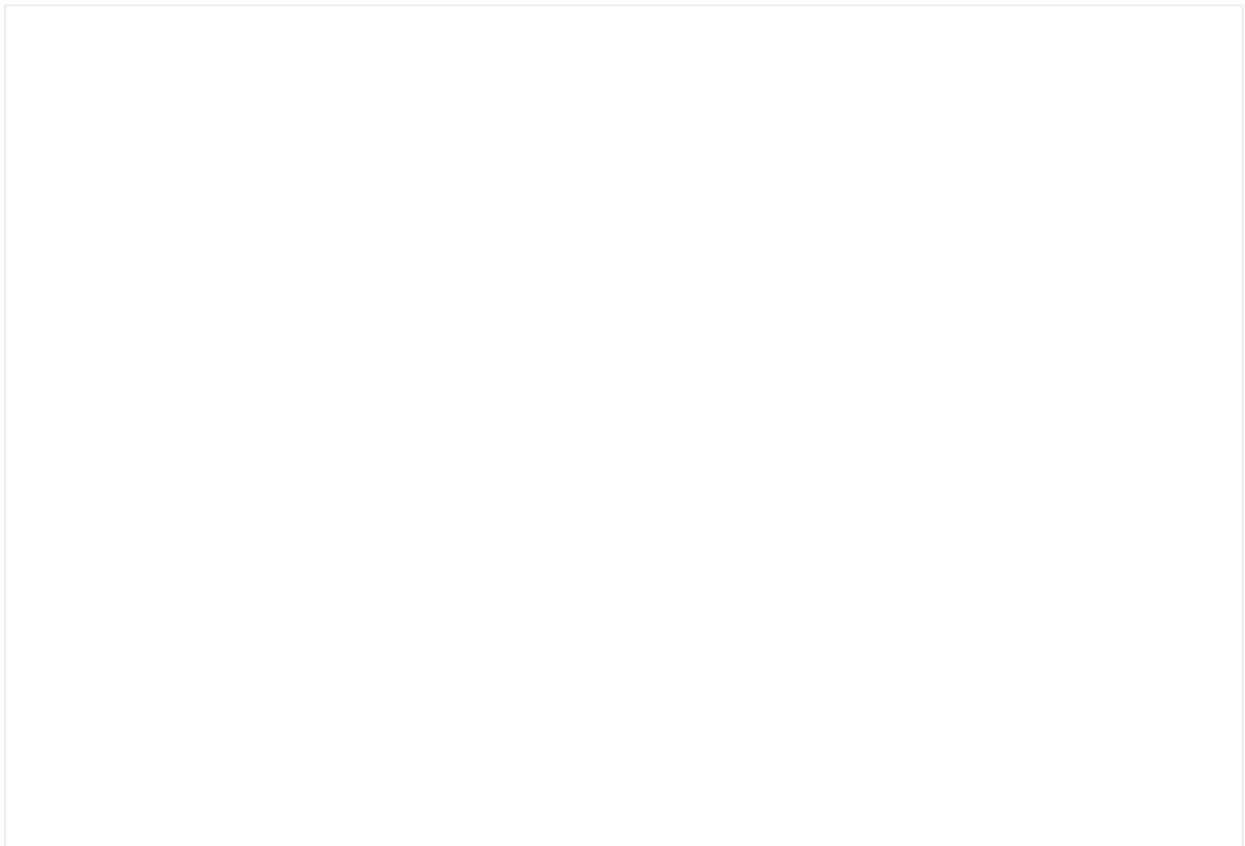
点击【编译】按钮就会调用keil打开工程或者直接写入单片机



打开keil就能看到集成到开发环境的Simulink代码

11 最后

Simulink代码生成专题共三篇就此结束，最后这个提高篇涉及的内容比较多，可能很多人看了还不是很明白如何实现的。最后就来总结下一些知识点：S函数、TLC语言、M语言。

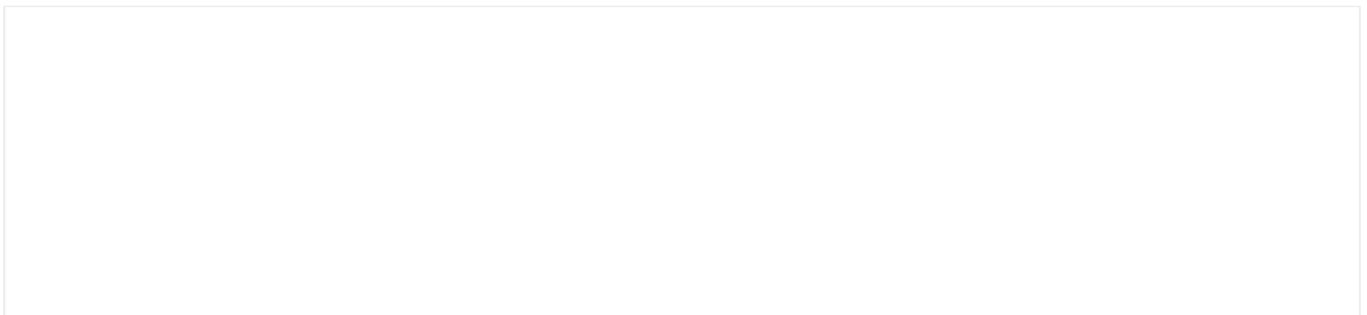


【思想】收集到的Matlab培训资料，及一个自定义目标系统的例子。有兴趣的可以分享本文或以下推荐文章到1个专业群和朋友圈，截图上传微信公众号即可获得。

----- RECOMMEND -----

推荐阅读

- S-Function应用实例
- 汽车工程师眼中的C#
- 工况路谱的采集与数据处理
- 混合动力节油的秘密-发动机万有特性
- AVL-CRUISE纯电动仿真策略提高教程
- AVL-CRUISE纯电动模型仿真策略
- Simulink代码生成应用教程
- Simulink代码生成基础体验教程
- 燃料电池车(FCHEV)动力经济性建模与仿真



培训、咨询、项目开发，请与【思想】联系

