DENSO

Crafting the Core

ADAS前方静止物感知功能的控制单元模型开发以及持续集成实践

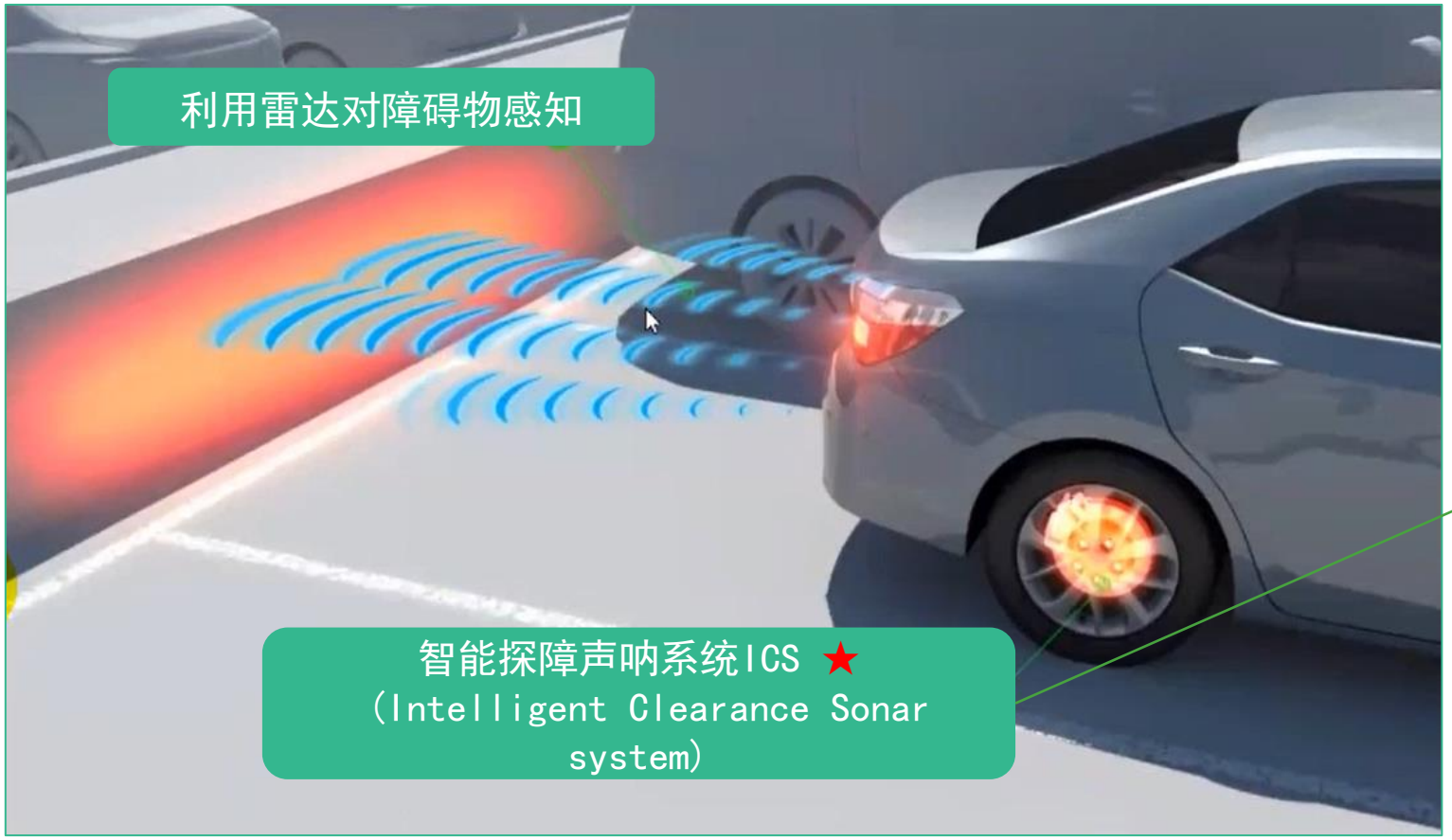
张强,电装智能科技（上海）有限公司

MATLAB EXPO

内容

- ADAS前后方静止物感知功能简介
- ADAS前后方静止物感知功能特性
- 面临的问题和挑战
- 基于模型的系统开发流程
- 基于模型的系统开发中我们面临的问题
- Jenkins上持续集成方法及自动进行验证(CI)
- 利用的工具一览
- 总结
- 了解更多

ADAS前后方静止物感知功能简介



智能探障声呐系统ICS

自动紧急刹车

自动泊车

侧方障碍提醒

数据融合，可信度判断

运行环境（RTE）

底层软件（BSW）

硬件

转载以下的链接内容
[トヨタ自動車、駐車場での安全支援技術 「インテリジェントクリアランスソナー（ICS）」の事故低減効果を発表 | Toyota Motor Corporation Official Global Website]

ADAS前后方静止物感知功能特性

探测的信息由于外部复杂条件，需要甄别信息的可信度。

小型车、微型车、紧凑车型、中等车型、高级车型、豪华车型、三厢车型、CDV车型、MPV车型、SUV等

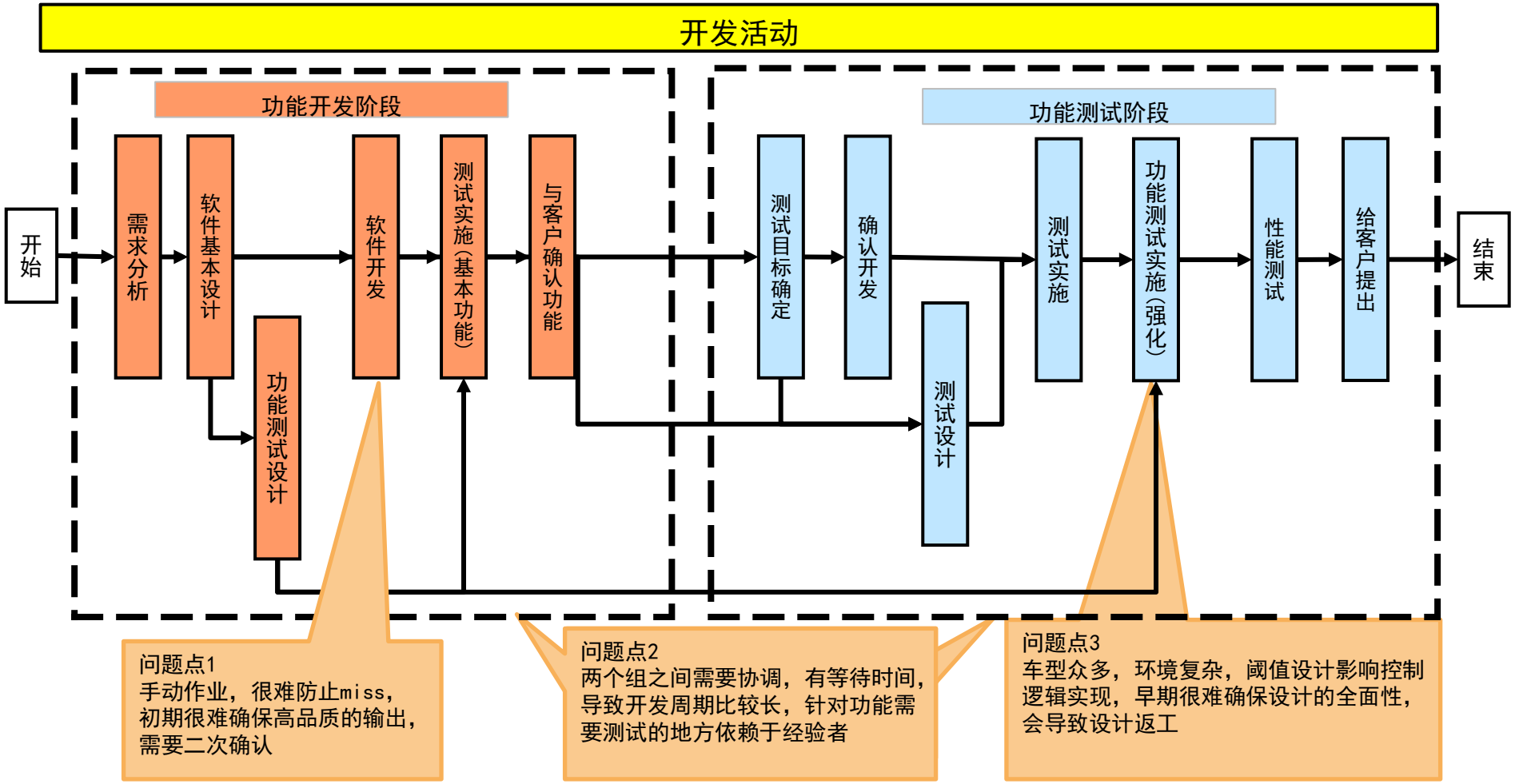
众多车型的的不同，感知带来的阈值也有区别

复杂的停车环境，对算法设计带来了挑战

图片源于网上搜索

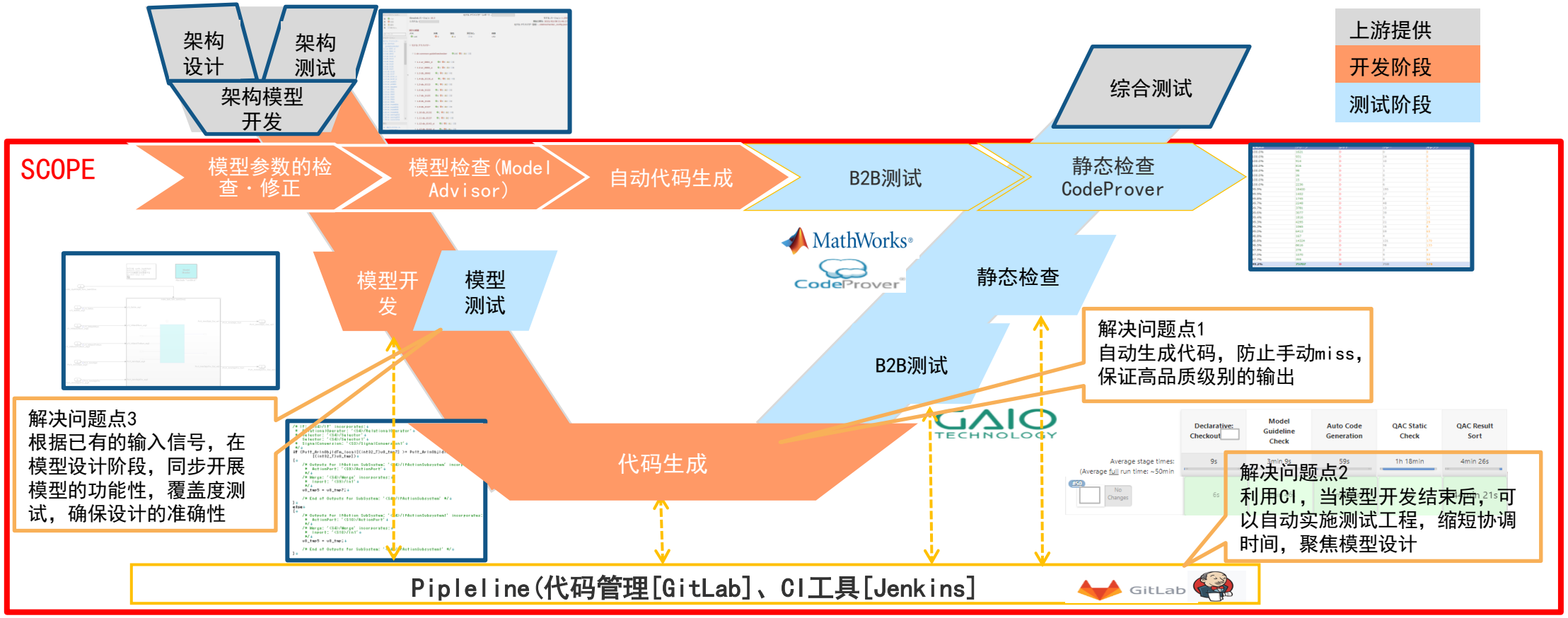
我们面临的问题和挑战

面对不同车企的众多车型，以及短周期的研发要求，如何早期验证产品是否满足客户的需求，以便能在短周期内交付高质量产品，成为了亟待解决的问题之一。



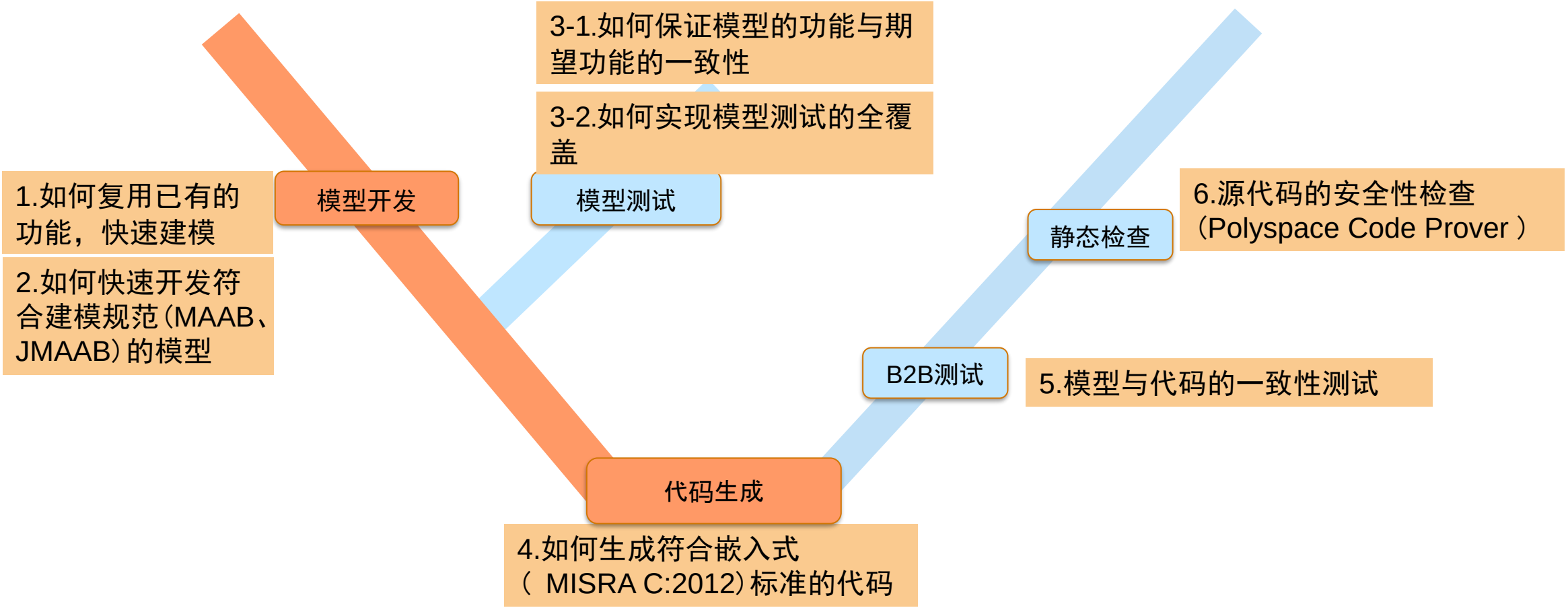
基于模型的系统开发流程

利用MATLAB，在已有算法的基础上，开发符合建模规范JMAAB的模型；在测试阶段，结合测试工具，确保软件的高质量；通过持续集成的方式，缩短开发周期，创建一个高可靠且可复用的研发环境。



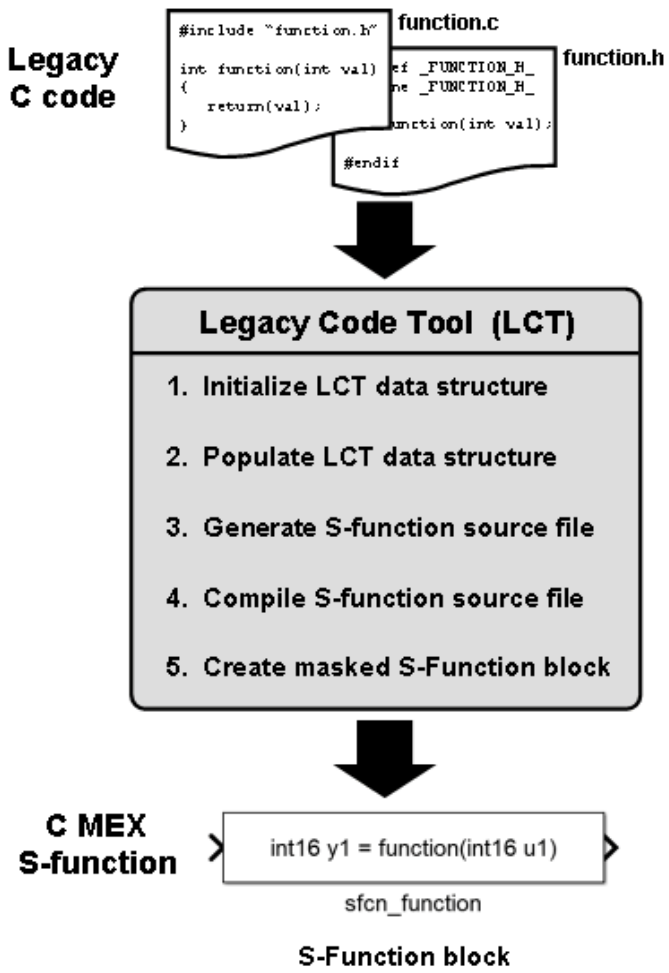
基于模型的系统开发中我们面临的问题

建模工作包含新增功能的实现和已有功能的模块化。在短周期开发的背景下，我们从模型开发到测试各个阶段都存在需要解决的课题



1. 如何复用已有的功能快速建模

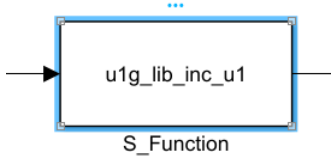
针对已有功能，MATLAB提供了丰富的集成方法，根据客户提供的指导意见说明，我们选取了S-Function，使用了legacy_code(代码继承工具)，实现已有功能的模型化转换。



- 使用的便利性 –
- 模型的易理解性 △
- 模块的独立性 ○
- ACG 代码的高效性 –
- MISRA-C 的遵守性 –
- B2B 测试效率实施性 ○

u1g_lib_inc_u1.c
u1g_lib_inc_u1.mexw64
u1g_lib_inc_u1.tlc

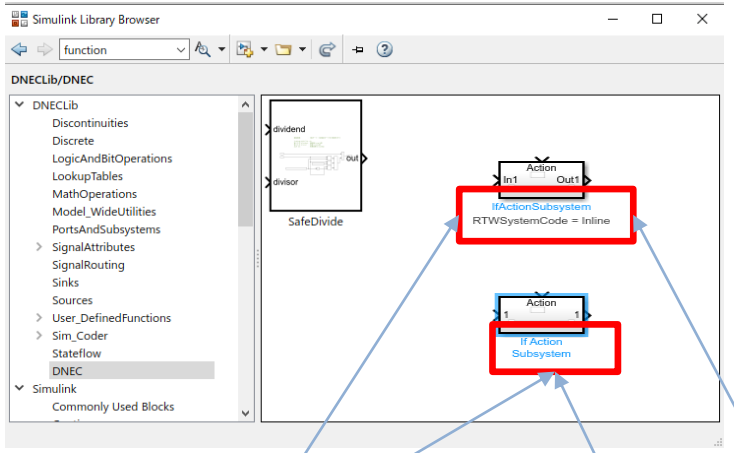
```
/**↓
 * ulg_lib_inc_ul.c↓
 *↓
 *    ABSTRACT:↓
 *      The purpose of this sfunction is to call a simple legacy↓
 *      function during simulation:↓
 *↓
 *        uint8 yl=ulg_lib_inc_ul(uint8 ul)↓
 *↓
 *    Simulink version          : 10.3 (R2021a) 14-Nov-2020↓
 *    C source code generated on : 25-Nov-2021 10:51:30↓
 *↓
 * THIS S-FUNCTION IS GENERATED BY THE LEGACY CODE TOOL AND MAY NOT WORK IF MODIFIED
 */↓
↓
/**↓
    %%%-MATLAB_Construction_Commands_Start↓
    def = legacy_code('initialize');↓
    def.SFunctionName = 'ulg_lib_inc_ul';↓
    def.OutputFcnSpec = 'uint8 yl=ulg_lib_inc_ul(uint8 ul)';↓
    def.HeaderFiles = {'libamt.h'};↓
    def.SourceFiles = {'libamt.c'};↓
    def.IncPaths = {'../inc'};↓
    def.SrcPaths = {'../src'};↓
    legacy_code('sfcn_cmex_generate', def);↓
    legacy_code('compile', def);↓
    %%%-MATLAB_Construction_Commands_End↓
 */↓
```




```
TLC 文件，用于在加速模式下或代码生成中进行仿真↓
%% file : ulg_lib_inc_ul.tlc↓
%% ↓
%% Description:↓
%%   Simulink Coder TLC Code Generation file for ulg_lib_inc_ul
%% ↓
%% Simulink version      : 10.3 (R2021a) 14-Nov-2020↓
%% TLC file generated on : 25-Nov-2021 10:51:36↓
↓
/%↓
    %%%-MATLAB_Construction_Commands_Start↓
    def = legacy_code('initialize');↓
    def.SFunctionName = 'ulg_lib_inc_ul';↓
    def.OutputFcnSpec = 'uint8 yl=ulg_lib_inc_ul(uint8 ul)';↓
    def.HeaderFiles = {'libamtopt.h'};↓
    def.SourceFiles = {'libamtopt.c'};↓
    def.IncPaths = {'../csr/inc'};↓
    def.SrcPaths = {'../csr/src'};↓
    legacy_code('sfcn_tlc_generate', def);↓
    %%%-MATLAB_Construction_Commands_End↓
%/↓
↓
%implements ulg_lib_inc_ul "C"↓
```

2. 如何快速开发符合建模规范(MAAB、JMAAB)的模型

在Simulink Library Browser的基础上，定制出符合我们开发需求的Library库，并让其在Simulink Library Browser中表示出来。



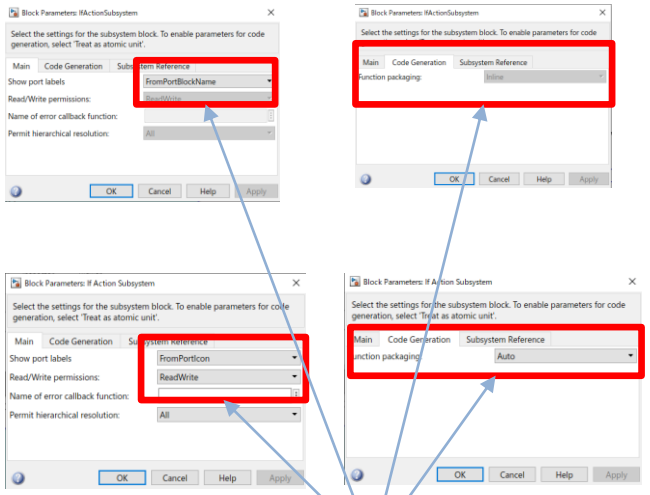
针对模块属性，在MATLAB提供的模块之上自建符合（ JMAAB ）要求的模块库

- 可以防止手动设置和公司要求设置不一致的问题
- 可以早期明确使用的限制
- 可以通过sl_customization向Simulink 注册自定义项

1.命名规则不同

DNECLib

Simulink



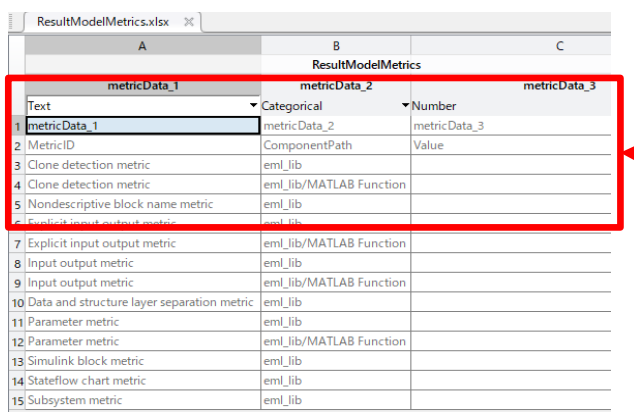
2.属性设置不同

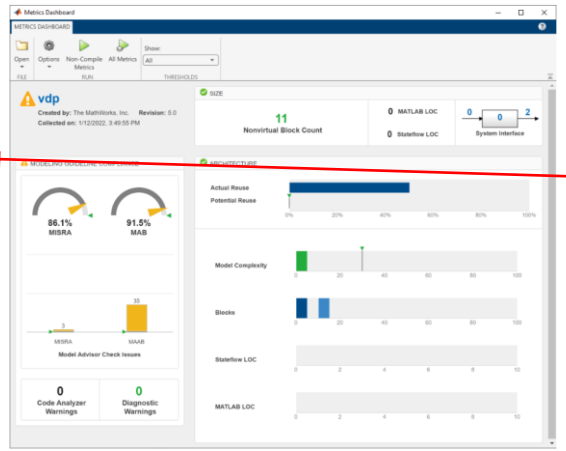
2. 如何快速开发符合建模规范（MAAB、JMAAB）的模型

1. 利用Model Advisor提供的API，自定制了公司规定的建模Model Advisor规范，统一了Config设置的检查
2. 利用Simulink Check提供的API， Model Metrics收集模型和测试工件的指标数据，确定模型是否符合大小、复杂度和可读性需求。

1. 统一Config设定。检查出不符合设定的项目，自动进行强制修改。

2. 提示出需要确认的设置信息。自行调整，不进行强制修改。

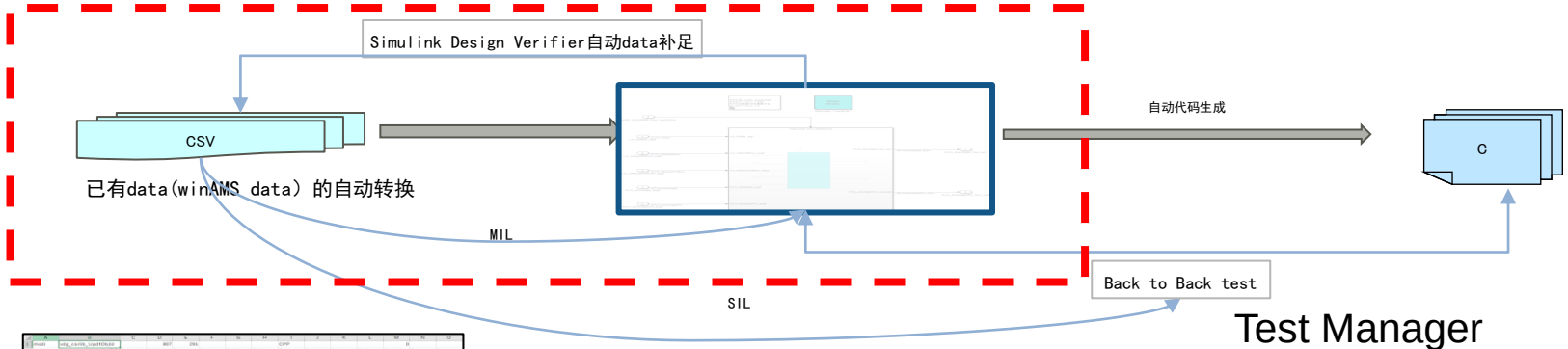




3. 为了做出易懂，可再利用的模型，对模型的复杂度进行了check。
–使用模型指标 API 创建自定义模型指标，计算指标结果，并导出指标数据。

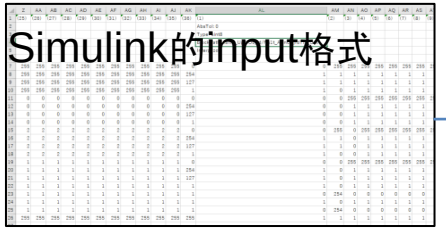
3-1. 如何保证模型的功能与期望功能的一致性

利用Test Manager进行一致性测试（模型与已有功能MIL）

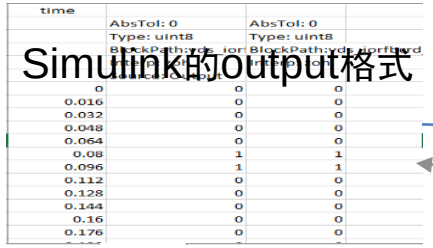


已有data（winAMS自动Testdata格式）

Simulink的input格式

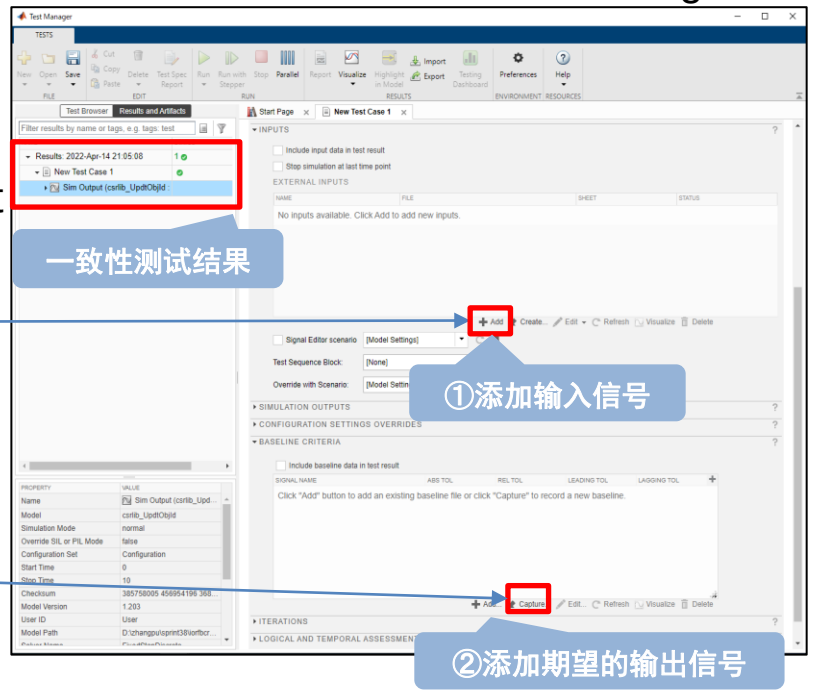


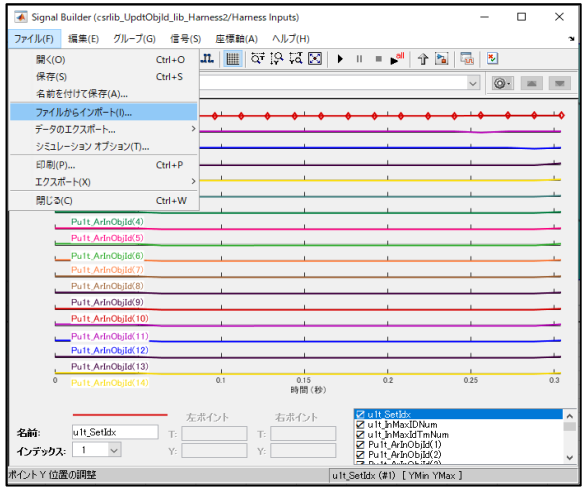
Simulink的output格式



自动转化Simulink的input/output

Test Manager

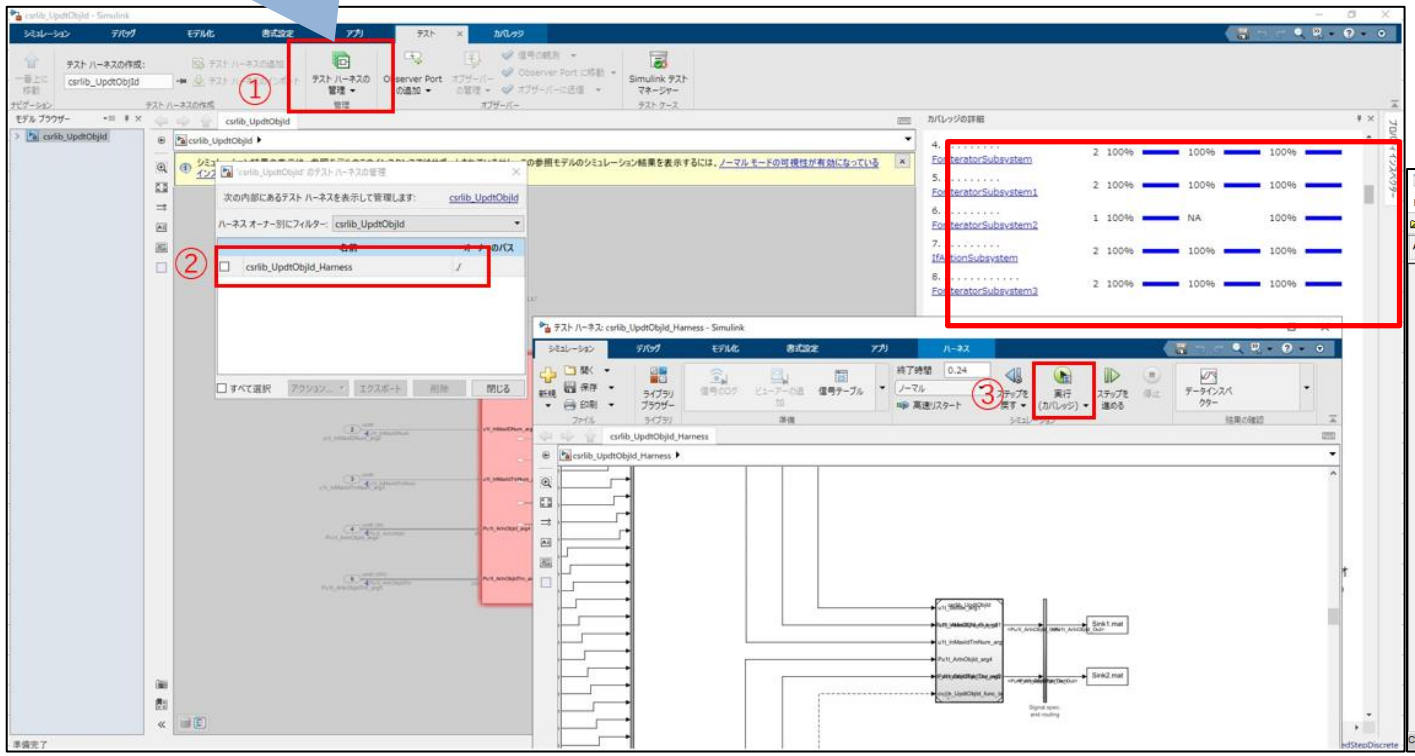




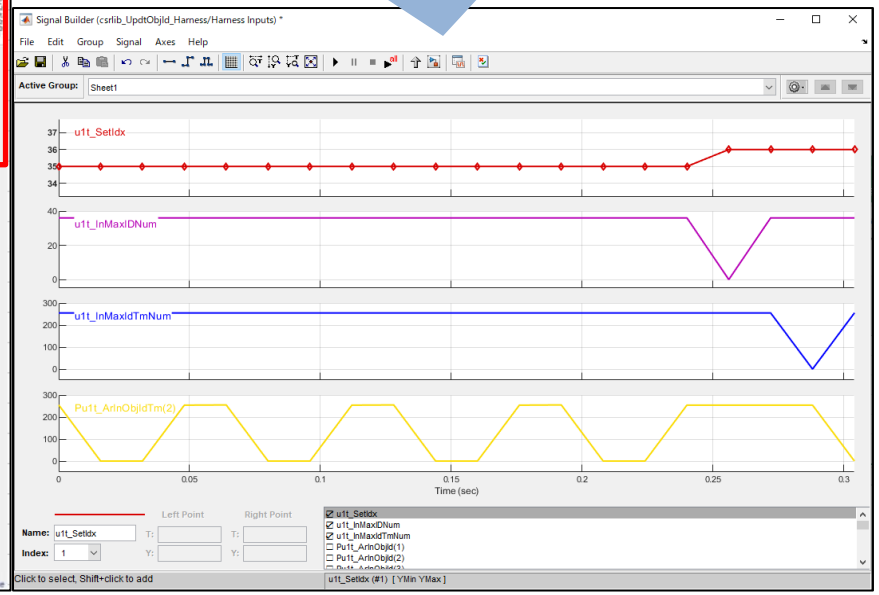
3-2. 如何实现模型测试的全覆盖

针对模型进行覆盖率测试，以确保模型所有分支的全覆盖

①打开TestHarness

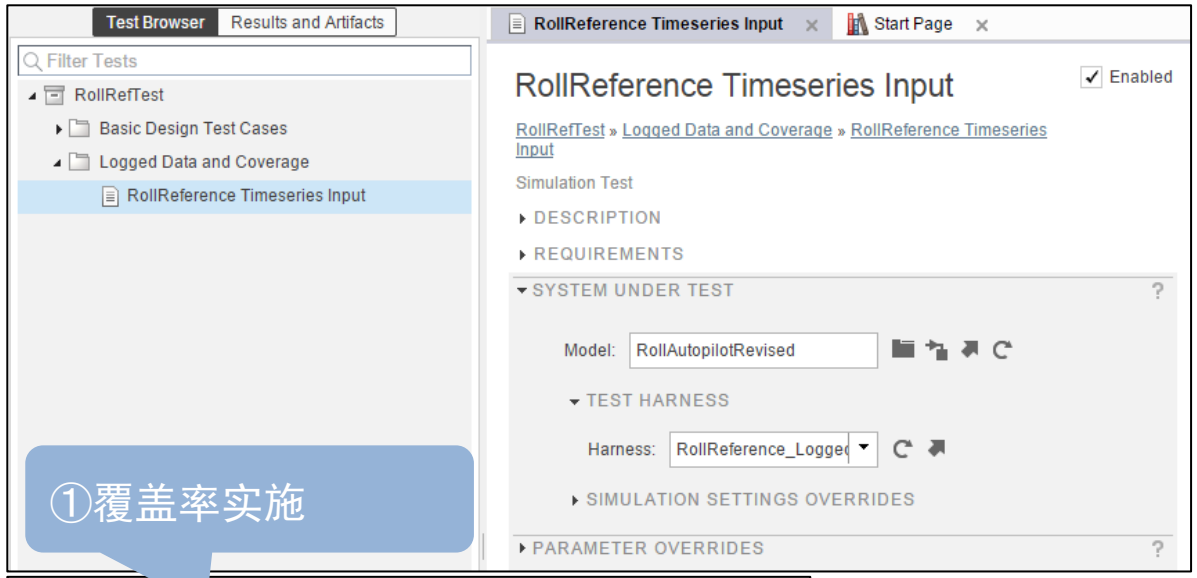


②testdata输入

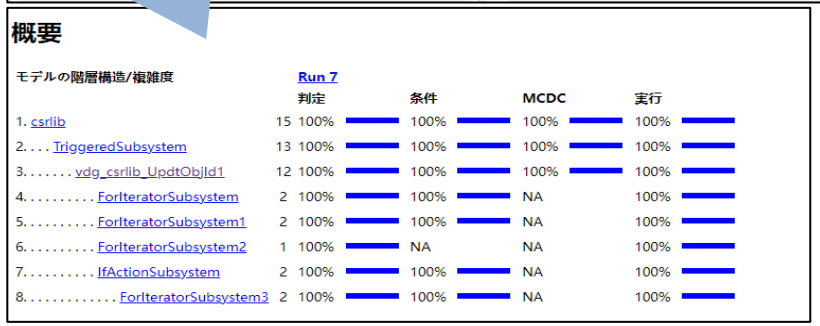


3-2. 如何实现模型测试的全覆盖

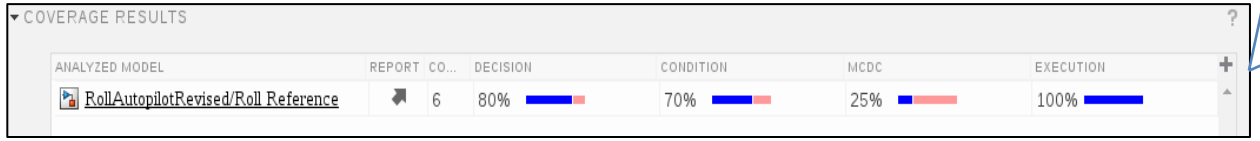
Simulink Design Verifier：扩展已有的测试用例，补充模型的覆盖率。



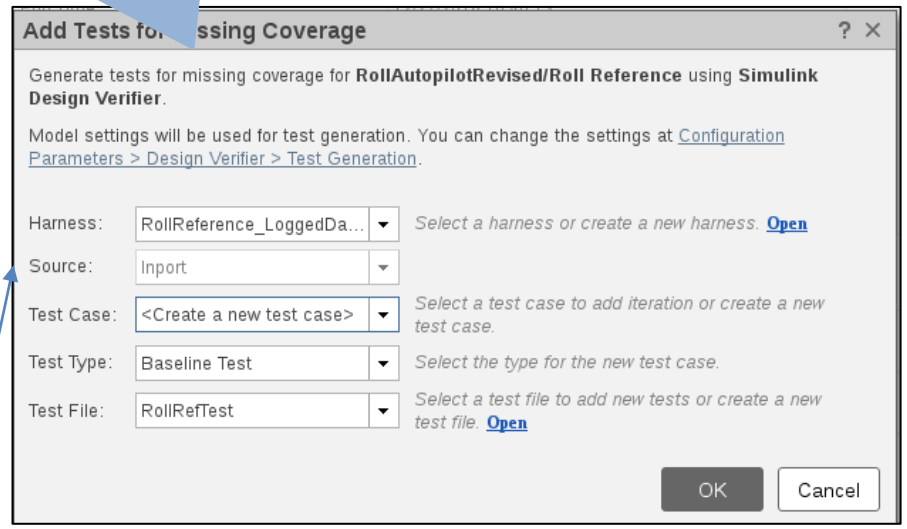
①覆盖率实施



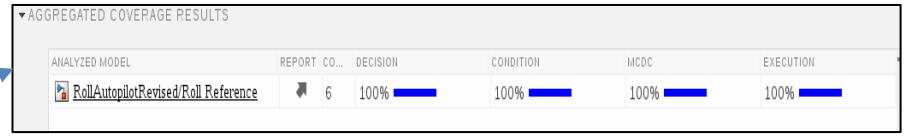
C0/C1＆MC/DC不足



②覆盖率的补充

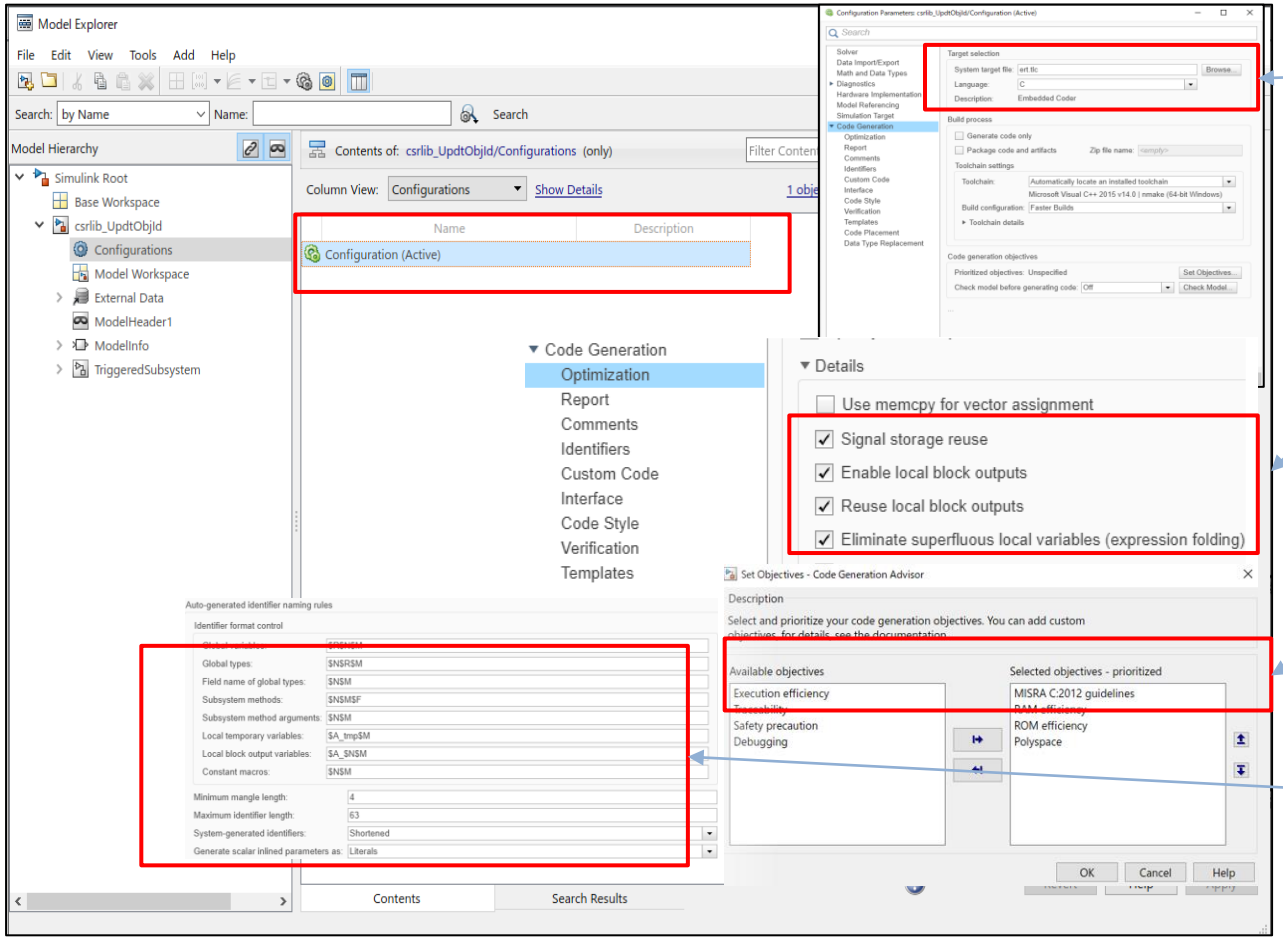


补全后覆盖率为100%



4. 如何生成符合嵌入式（ MISRA C:2012）标准的代码

Embedded Coder针对代码生成，提供了大量配置选项。其中通过设定Code Generation属性可以生成公司规定风格并符合（ MISRA C:2012 ）标准的C Code



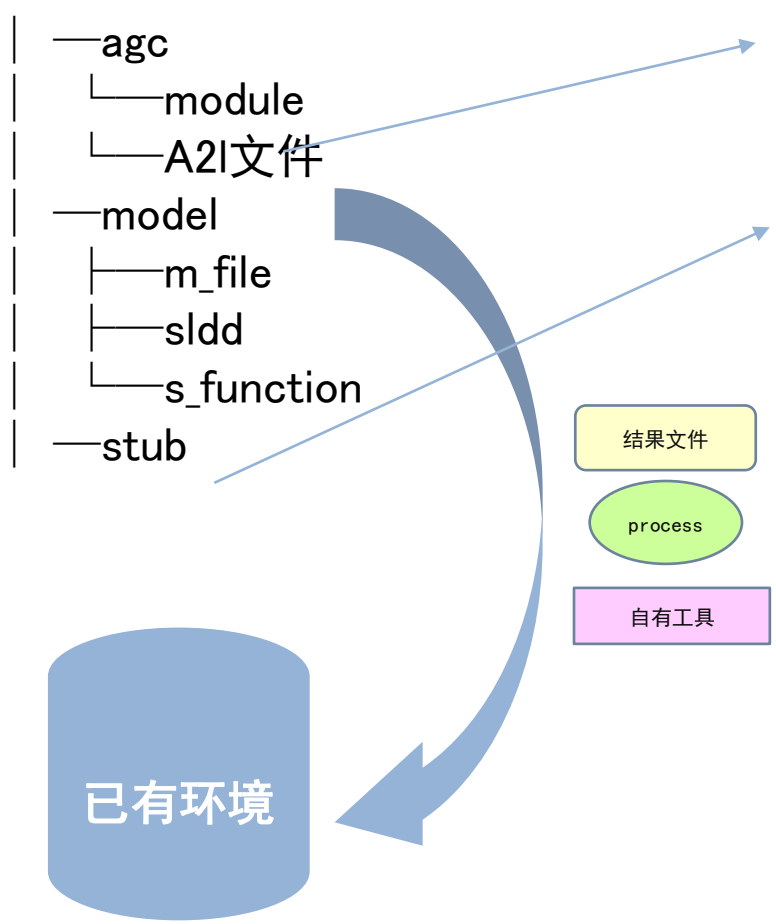
ert.tlc(embedded coder)
嵌入式系统优化的C代码

代码的复用性，可以缩减ROM/RAM量

Code Generation Advisor检查代码符合MISRA C:2012标准

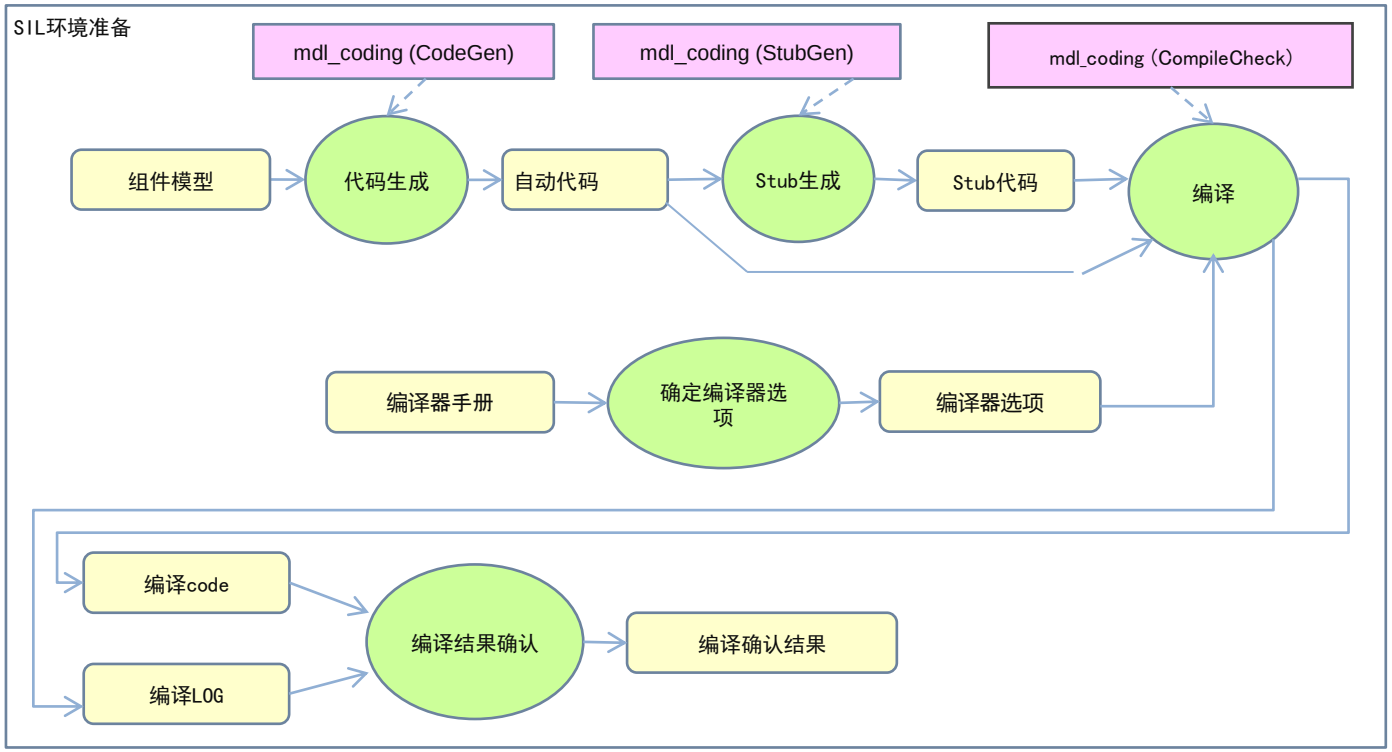
设定符合自己命名规范的代码

5. 模型与代码的一致性测试



ASAP2 defines the description format of the internal ECU variables used in measurement and calibration.

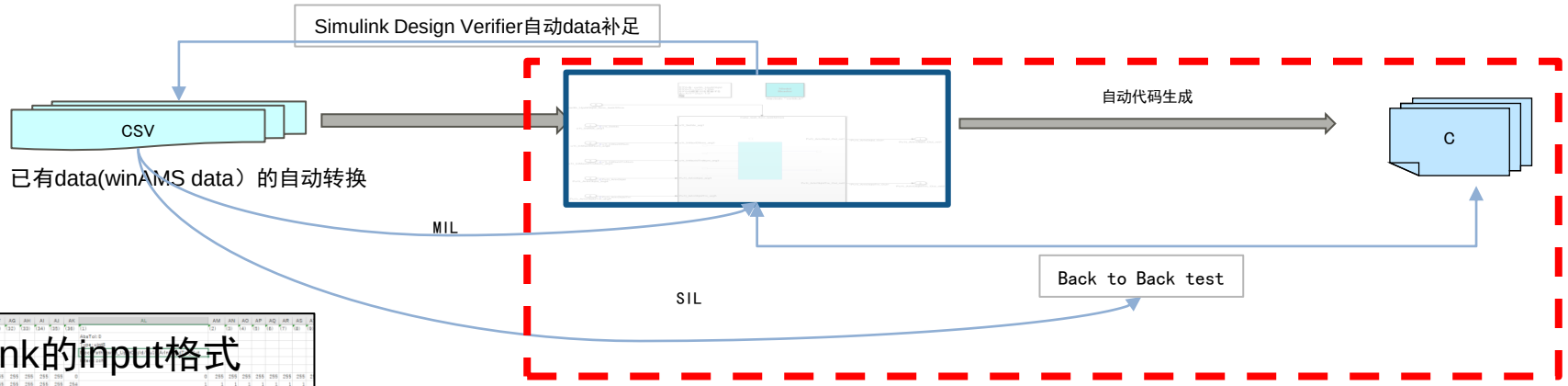
StubGen
模型生成的自动代码，在编译/单体实施的时候，StubGen可以生成必要的Stub函数。



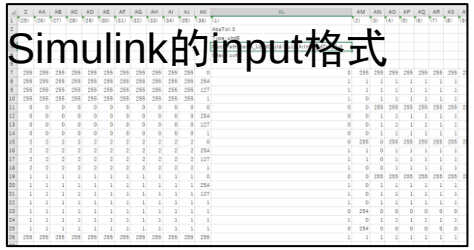
编译过后，merge到编译环境里面

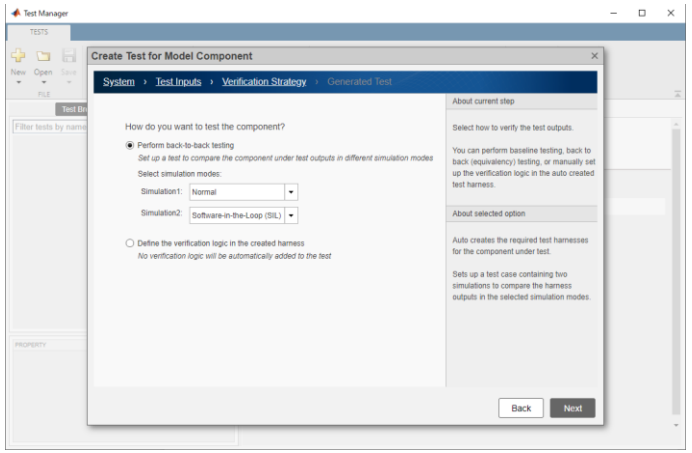
5. 模型与代码的一致性测试

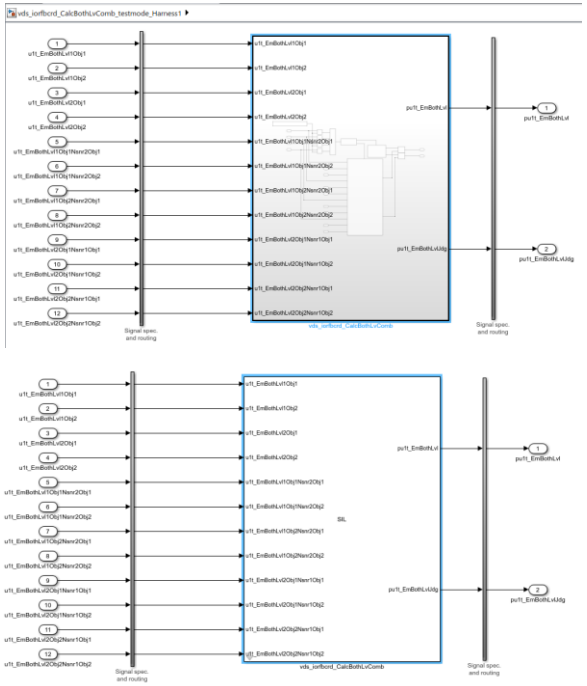
利用Test Manager功能进行一致性测试（模型与已有功能SIL）

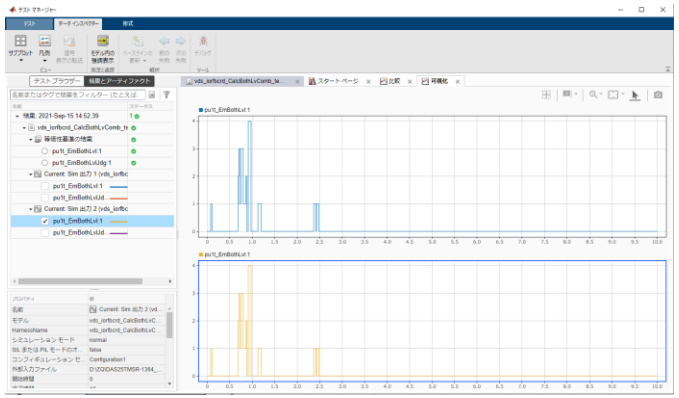


Simulink的input格式





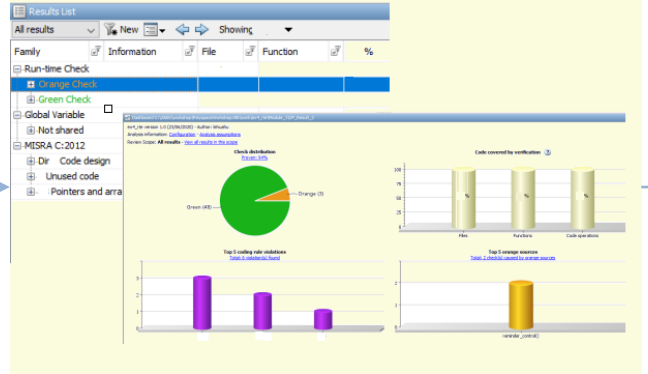




6. 源代码的安全性检查（Polyspace Code Prover ）

利用Polyspace Code Prover工具，可以静态检查代码存在的问题，并生成分析结果报告书

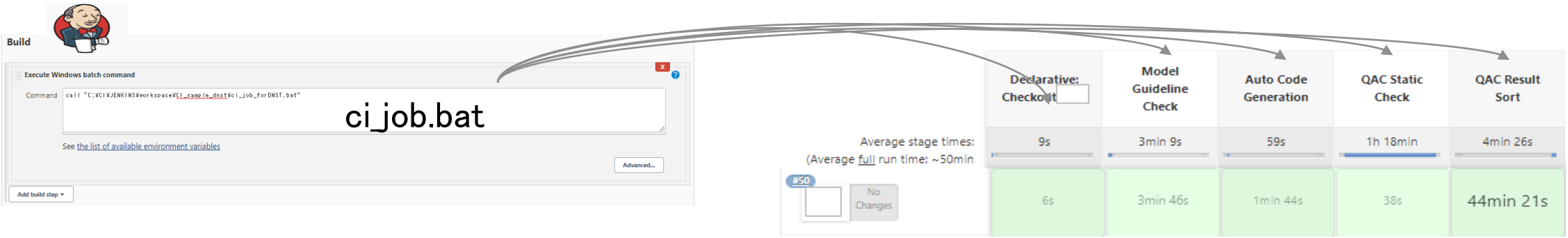
Git环境



- 代码规范性检查
- 冗余代码检查
- 初始化检查
- 值域越界检查
- 死逻辑检查
- 不正确的指针访问检查

ID	警告	函数	行	Col	详细	Jus	重大度	status
73813	overflow	vds_aaa_test_func()	1821	28	Warning: operation [conversion from int32 to int16] on scalar may overflow (on MIN or MAX bounds of INT16)	No	Unset	Unreviewed

Jenkins上持续集成方法及自动进行验证(CI)



API(M語言)	功能	报告文件
mdl_design	模型设置，建模规范检查，模型复杂度测量	建模规范设定报告书 建模规范检查报告书 模型复杂度测量报告书
mdl_coding	自动代码生成，stub函数生成，检查编译	自动代码，Stub代码，编译器选项，编译LOG，编译结果
mdl_static_check	静态检查，静态检查结果统计（Polyspace Code Prover）	Code Prover警告报告书 新加警告比较结果报告书
Mdl_design_verify	B2B测试，一致性检查结果	测试数据、一致性测试结果

利用的工具一览

开发阶段 \ MATLAB/Simulink	MATLAB	Simulink	Stateflow	Embedded Coder	Simulink Coder	MATLAB Coder	Simulink Check	Simulink Design Verifier	Simulink Coverage	Simulink Test	Polyspace Code Prover
模型开发	○	○	○								
模型测试							○	○	○	○	
代码生成				○	○	○					
静态检查											○
B2B测试	○	○	○					○	○	○	

工具概要	工具名	版本
MATLAB上的编译工具	MinGW:https://www.mathworks.com/support/compilers	-
ACG代码编译工具	Green Hills Software, Compiler	RH850_V.R954, TRI_V.R9151
ACG代码静态检查	Polyspace Code Prover	R2019a(10.0)
ACG代码测试工具	WinAMS	7.0

总结

- MATLAB可以让测试左移，早期验证设计
- MATLAB可以很好的与Jenkins相结合，利用其公开的API方便实现个人定制
- 通过良好的软件和建模实践节省了项目的测试时间，保证了高品质级别的输出

了解更多

- PIL实施准备
- 关于指针与模型联动的方法的研究
- 基本功能实现模块化，根据自身功能特点有效利用MATLAB关于数学计算方面的优势
- MATLAB内部也提供AUTOSAR的接口，更有效的实现接口的标准化

MATLAB EXPO

Thank you

MathWorks®

© 2022 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See *mathworks.com/trademarks* for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.