基于模型设计
在机载安全关键领域中的应用

武方方

目录

1 技术应用背景

2 MBSE流程

3 工具建设

4 应用实例

基于模型设计广泛应用于航空航天领域

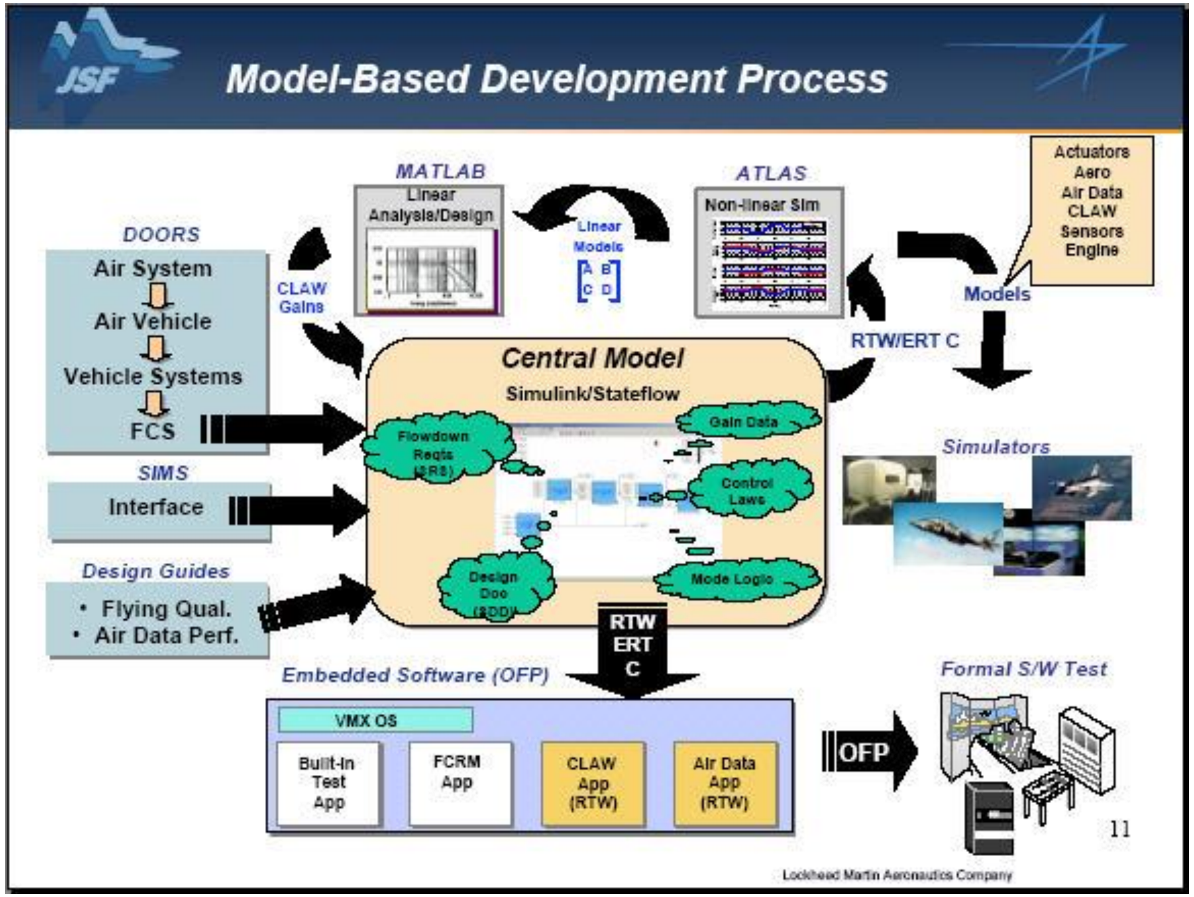
JSF-F35

Orion Spacecraft—猎户座

Space X

- 模型作为系统设计和开发的数据载体，在设计开发流程中进行传递，实现跨专业多部门的协同
- 统一的模型设计规范：标准，经验，知识
- 可复用的标准模型库：算法，对象，环境
- 集成一体的工程应用：流程，自动，易用

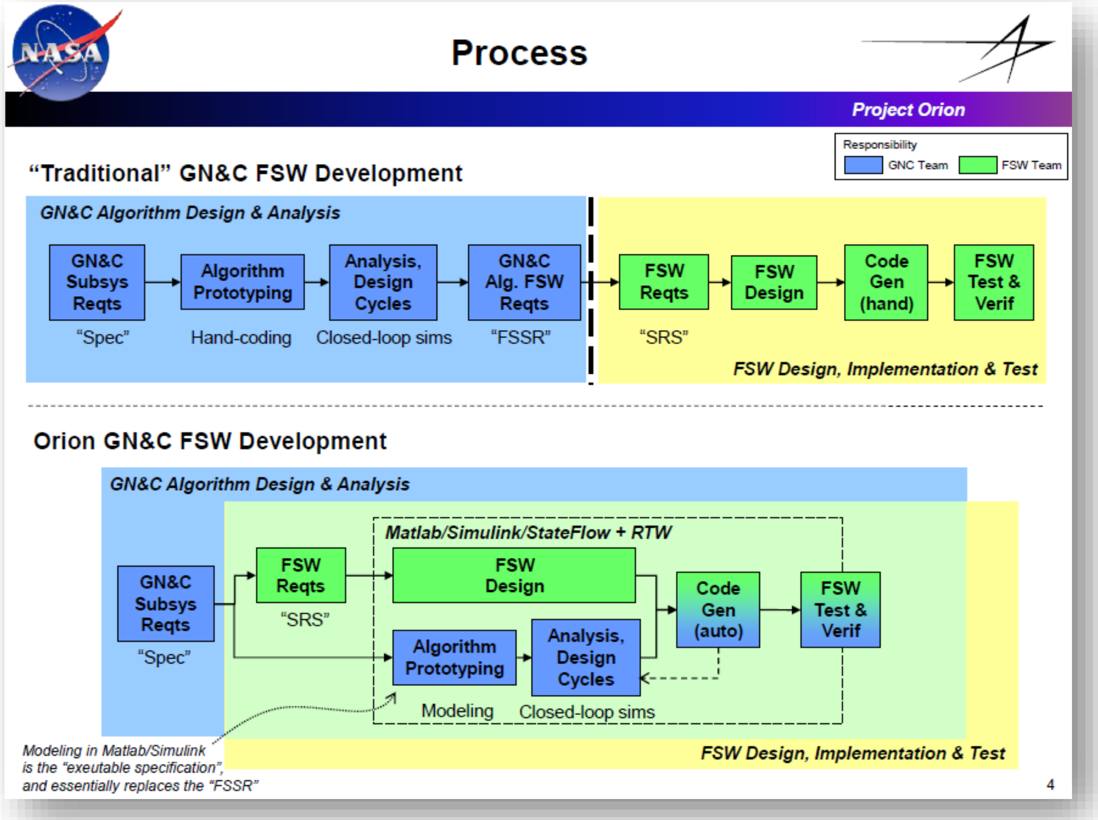
洛克希德马丁-JSF联合攻击机



- Library files ： **+ 266**
- Blocks： **16,143**
- Subsystem： **871**
- Instances of utility subsystems： **998**
- Logical code lines： **~47,000**
- Code files： **750**

- **Single Electronic Source for All Software Requirements, Design, and Implementation**
  - Graphical Representation of Software Design - No Paper Diagrams or Separate Block Diagrams
  - All Textual Documentation Embedded in Model
- **Automatic Code Generation Process to Eliminate Coding Defects**
  - Eliminate Errors Normally Incurred From Translating Requirements Into Design and Code
- **Model Thoroughly Evaluated in Analytical and Simulation Environment**
  - Code Supplied to Six DOF Simulation (ATLAS) for Dynamic Analysis and Piloted Simulator
  - Prototype Design Changes Rigorously Tested in Simulator with Test Pilots

NASA猎户座飞船项目的流程改进



Process

Project Orion

Responsibility: 🟦 GNC Team 🟩 FSW Team

"Traditional" GN&C FSW Development

GN&C Algorithm Design & Analysis

GN&C Subsys Reqts → Algorithm Prototyping → Analysis, Design Cycles → GN&C Alg. FSW Reqts → FSW Reqts → FSW Design → Code Gen (hand) → FSW Test & Verif

"Spec" | Hand-coding | Closed-loop sims | "FSSR" | "SRS"

FSW Design, Implementation & Test

Orion GN&C FSW Development

GN&C Algorithm Design & Analysis

Matlab/Simulink/StateFlow + RTW

GN&C Subsys Reqts → FSW Reqts → FSW Design → Code Gen (auto) → FSW Test & Verif

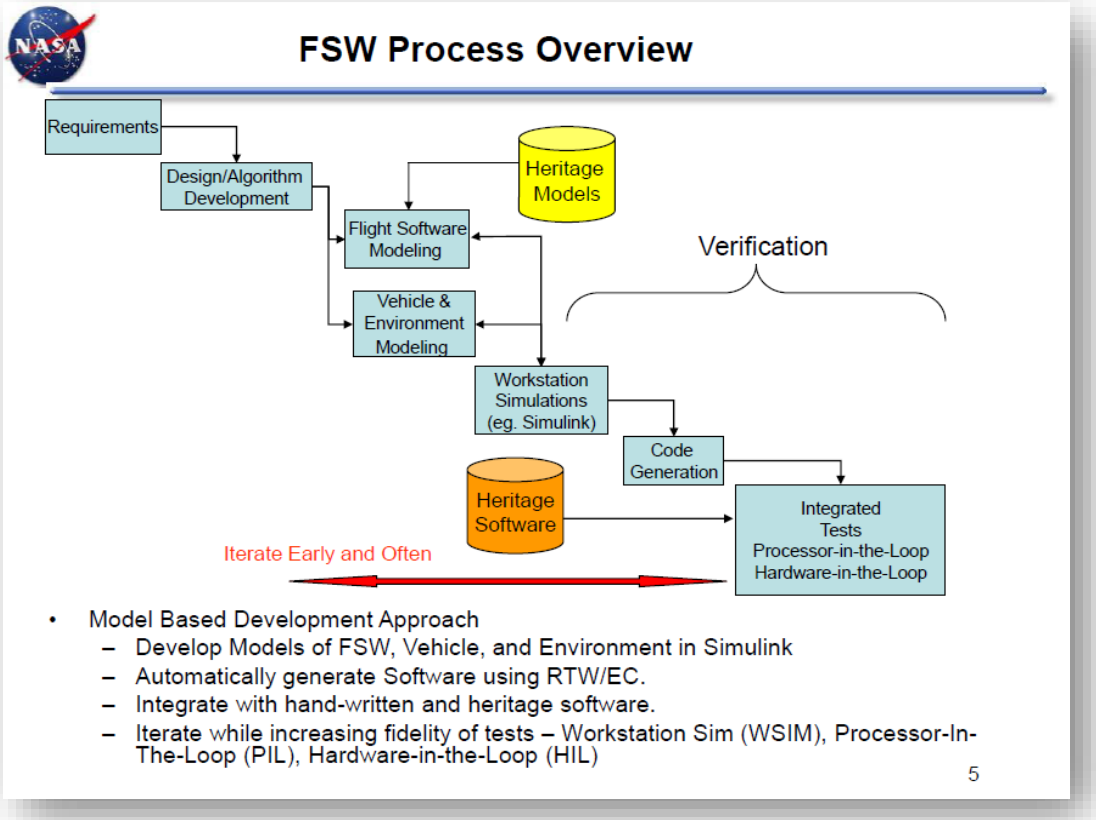
GN&C Subsys Reqts → Algorithm Prototyping → Analysis, Design Cycles

"Spec" | "SRS" | Modeling | Closed-loop sims

Modeling in Matlab/Simulink is the "exeutable specification", and essentially replaces the "FSSR"

FSW Design, Implementation & Test

4



FSW Process Overview

Requirements → Design/Algorithm Development → Flight Software Modeling; Vehicle & Environment Modeling → Workstation Simulations (eg. Simulink) → Code Generation → Integrated Tests Processor-in-the-Loop Hardware-in-the-Loop

Heritage Models

Heritage Software

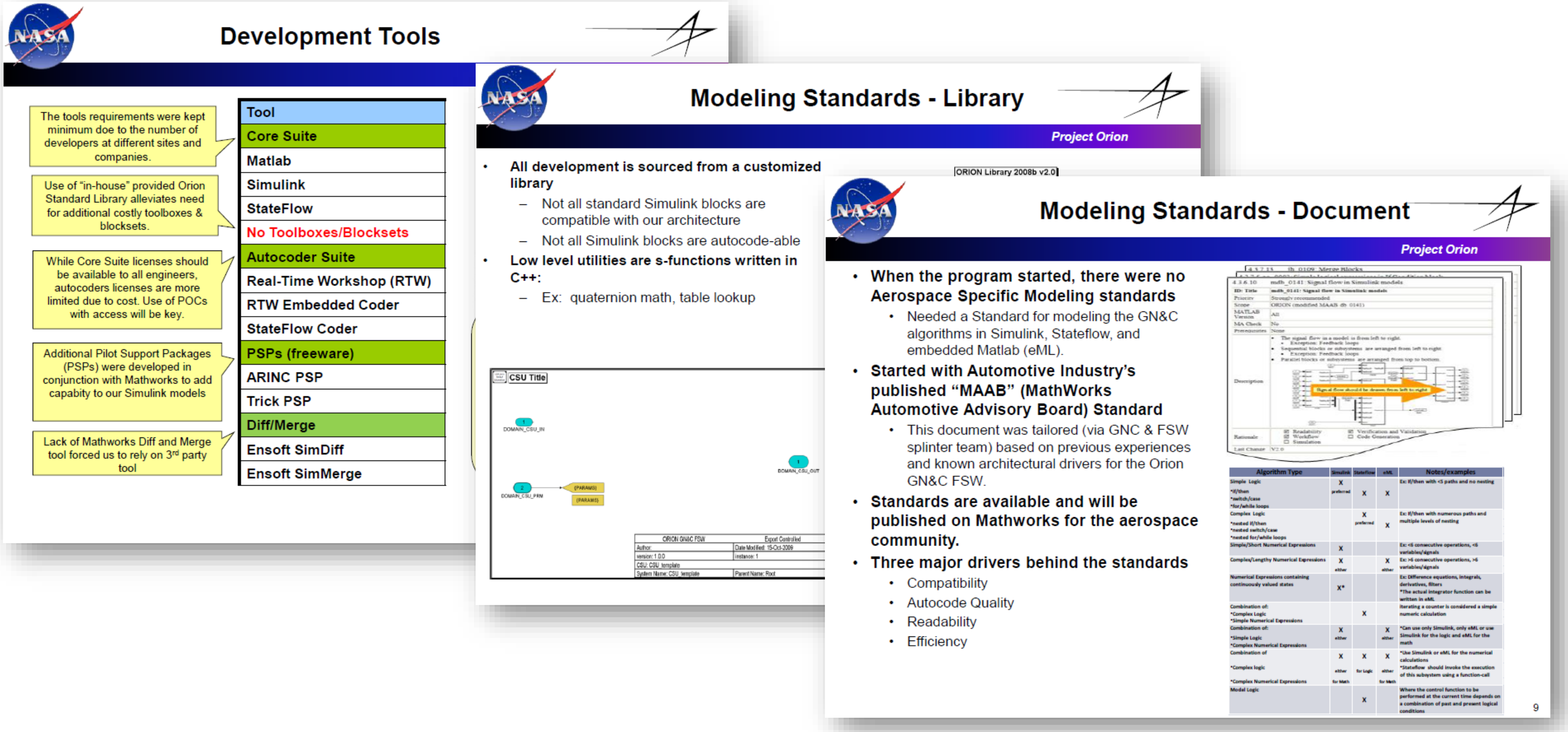
Verification

Iterate Early and Often

- Model Based Development Approach
  - Develop Models of FSW, Vehicle, and Environment in Simulink
  - Automatically generate Software using RTW/EC.
  - Integrate with hand-written and heritage software.
  - Iterate while increasing fidelity of tests – Workstation Sim (WSIM), Processor-In-The-Loop (PIL), Hardware-in-the-Loop (HIL)

5

NASA猎户座项目实施



Development Tools

The tools requirements were kept minimum doe to the number of developers at different sites and companies.

Use of “in-house” provided Orion Standard Library alleviates need for additional costly toolboxes & blocksets.

While Core Suite licenses should be available to all engineers, autocoders licenses are more limited due to cost. Use of POCs with access will be key.

Additional Pilot Support Packages (PSPs) were developed in conjunction with Mathworks to add capabity to our Simulink models

Lack of Mathworks Diff and Merge tool forced us to rely on 3rd party tool

Tool
Core Suite
Matlab
Simulink
StateFlow
No Toolboxes/Blocksets
Autocoder Suite
Real-Time Workshop (RTW)
RTW Embedded Coder
StateFlow Coder
PSPs (freeware)
ARINC PSP
Trick PSP
Diff/Merge
Ensoft SimDiff
Ensoft SimMerge

Modeling Standards - Library

Project Orion

- **All development is sourced from a customized library**
  - Not all standard Simulink blocks are compatible with our architecture
  - Not all Simulink blocks are autocode-able
- **Low level utilities are s-functions written in C++:**
  - Ex: quaternion math, table lookup

ORION Library 2008b v2.0

CSU Title

ORION GN&C FSW	Export Controlled
Author:	Date Modified: 15-Oct-2009
version: 1.0.0	instance: 1
CSU: CSU_template	
System Name: CSU_template	Parent Name: Root

Modeling Standards - Document

Project Orion

- **When the program started, there were no Aerospace Specific Modeling standards**
  - Needed a Standard for modeling the GN&C algorithms in Simulink, Stateflow, and embedded Matlab (eML).
- **Started with Automotive Industry’s published “MAAB” (MathWorks Automotive Advisory Board) Standard**
  - This document was tailored (via GNC & FSW splinter team) based on previous experiences and known architectural drivers for the Orion GN&C FSW.
- **Standards are available and will be published on Mathworks for the aerospace community.**
- **Three major drivers behind the standards**
  - Compatibility
  - Autocode Quality
  - Readability
  - Efficiency

4.3.7.13 jh_0109: Merge Blocks

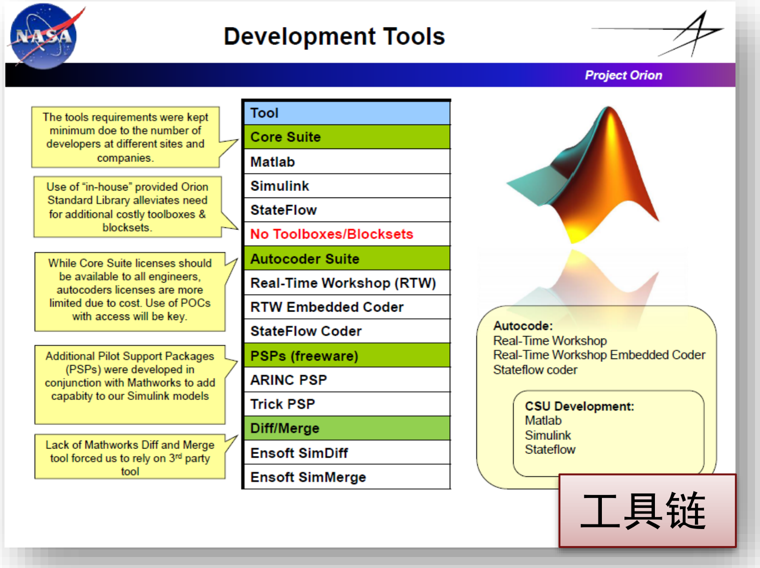
4.3.6.10 mdb_0141 Signal flow in Simulink models

ID: Title	mdb_0141: Signal flow in Simulink models
Priority	Strongly recommended
Scope	ORION (modified MAAB db_0141)
MATLAB Version	All
MA Check	No
Prerequisites	None
Description	• The signal flow in a model is from left to right. ▪ Exception: Feedback loops • Sequential blocks or subsystems are arranged from left to right ▪ Exception: Feedback loops • Parallel blocks or subsystems are arranged from top to bottom. Signal flow should be drawn from left to right
Rationale	☑ Readability ☑ Workflow □ Simulation ☑ Verification and Validation □ Code Generation
Last Change	V2.0

Algorithm Type	Simulink	Stateflow	eML	Notes/examples
Simple Logic •if/then •switch/case •for/while loops	X preferred	X	X	Ex: If/then with <5 paths and no nesting
Complex Logic •nested if/then •nested switch/case •nested for/while loops		X preferred	X	Ex: If/then with numerous paths and multiple levels of nesting
Simple/Short Numerical Expressions	X			Ex: <6 consecutive operations, <6 variables/signals
Complex/Lengthy Numerical Expressions	X either		X either	Ex: >6 consecutive operations, >6 variables/signals
Numerical Expressions containing continuously valued states	X*			Ex: Difference equations, integrals, derivatives, filters *The actual integrator function can be written in eML
Combination of: •Complex Logic •Simple Numerical Expressions		X		Iterating a counter is considered a simple numeric calculation
Combination of: •Simple Logic •Complex Numerical Expressions	X either		X either	*Can use only Simulink, only eML or use Simulink for the logic and eML for the math
Combination of •Complex logic •Complex Numerical Expressions	X either for Math	X for Logic	X either for Math	*Use Simulink or eML for the numerical calculations *Stateflow should invoke the execution of this subsystem using a function-call
Modal Logic		X		Where the control function to be performed at the current time depends on a combination of past and present logical conditions

9

NASA-猎户座飞船GNC系统设计



Development Tools

Project Orion

The tools requirements were kept minimum doe to the number of developers at different sites and companies.

Use of "in-house" provided Orion Standard Library alleviates need for additional costly toolboxes & blocksets.

While Core Suite licenses should be available to all engineers, autocoders licenses are more limited due to cost. Use of POCs with access will be key.

Additional Pilot Support Packages (PSPs) were developed in conjunction with Mathworks to add capabity to our Simulink models

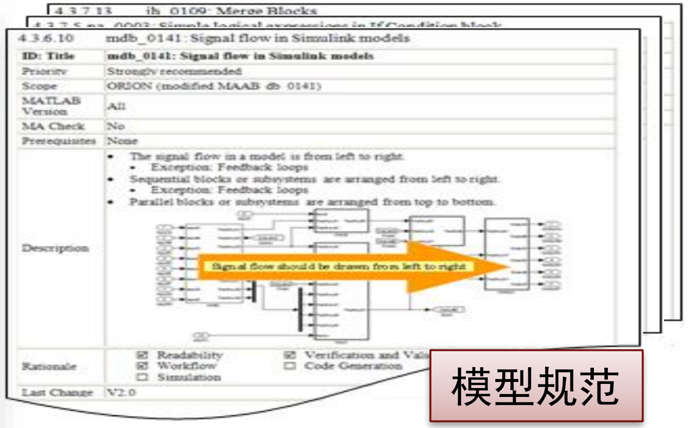
Lack of Mathworks Diff and Merge tool forced us to rely on 3rd party tool

Tool
Core Suite
Matlab
Simulink
StateFlow
No Toolboxes/Blocksets
Autocoder Suite
Real-Time Workshop (RTW)
RTW Embedded Coder
StateFlow Coder
PSPs (freeware)
ARINC PSP
Trick PSP
Diff/Merge
Ensoft SimDiff
Ensoft SimMerge

Autocode:
Real-Time Workshop
Real-Time Workshop Embedded Coder
Stateflow coder

CSU Development:
Matlab
Simulink
Stateflow

工具链



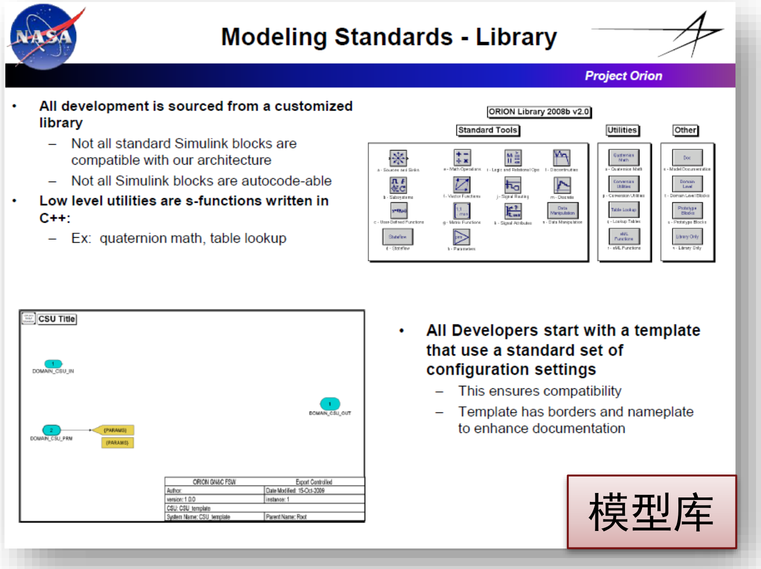
4.3.7.13 jh_0109: Merge Blocks

4.3.7.5 na_0003: Simple logical expressions in If Condition block

4.3.6.10 mdb_0141: Signal flow in Simulink models

ID: Title	**mdb_0141: Signal flow in Simulink models**
Priority	Strongly recommended
Scope	ORION (modified MAAB db_0141)
MATLAB Version	All
MA Check	No
Prerequisites	None
Description	- The signal flow in a model is from left to right. - Exception: Feedback loops - Sequential blocks or subsystems are arranged from left to right. - Exception: Feedback loops - Parallel blocks or subsystems are arranged from top to bottom. Signal flow should be drawn from left to right
Rationale	☒ Readability ☒ Workflow ☐ Simulation ☒ Verification and Vali… ☐ Code Generation
Last Change	V2.0

模型规范



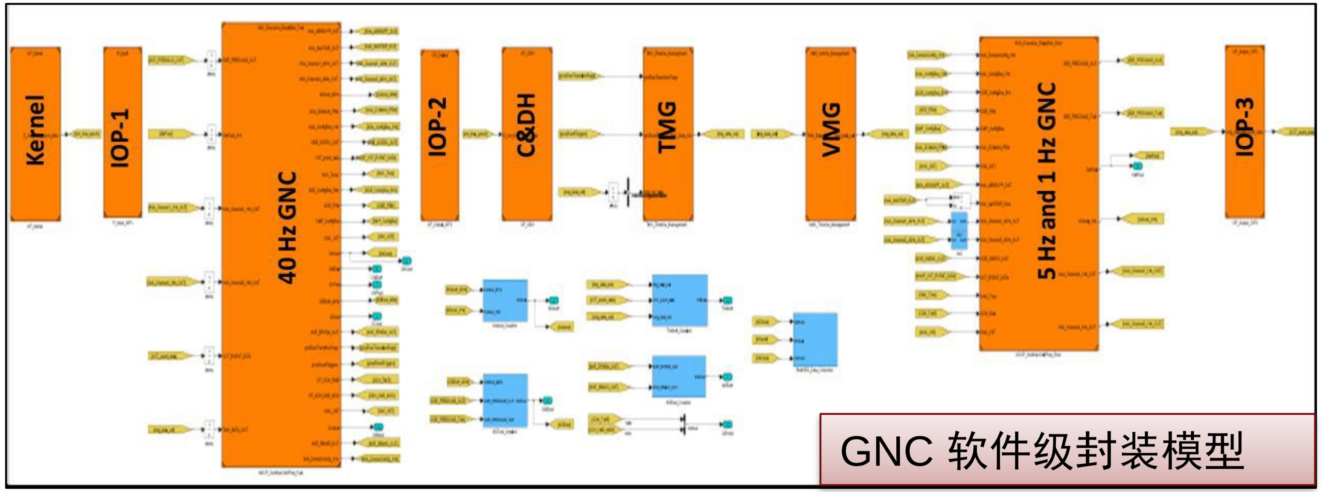
Modeling Standards - Library

Project Orion

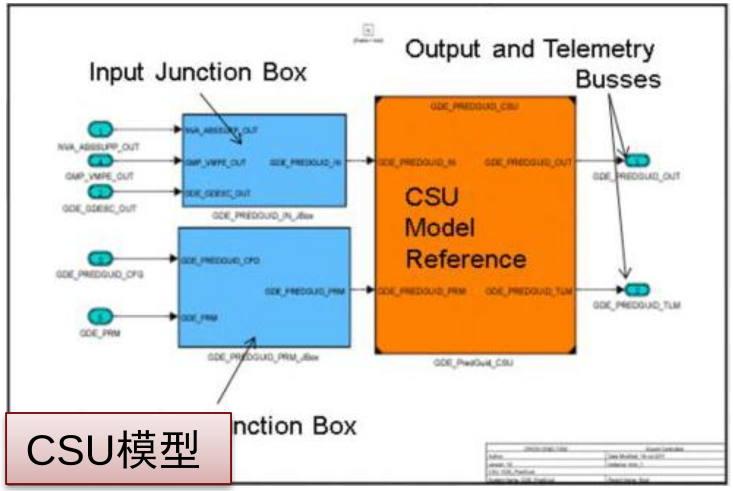
- **All development is sourced from a customized library**
  - Not all standard Simulink blocks are compatible with our architecture
  - Not all Simulink blocks are autocode-able
- **Low level utilities are s-functions written in C++:**
  - Ex: quaternion math, table lookup

- **All Developers start with a template that use a standard set of configuration settings**
  - This ensures compatibility
  - Template has borders and nameplate to enhance documentation

模型库

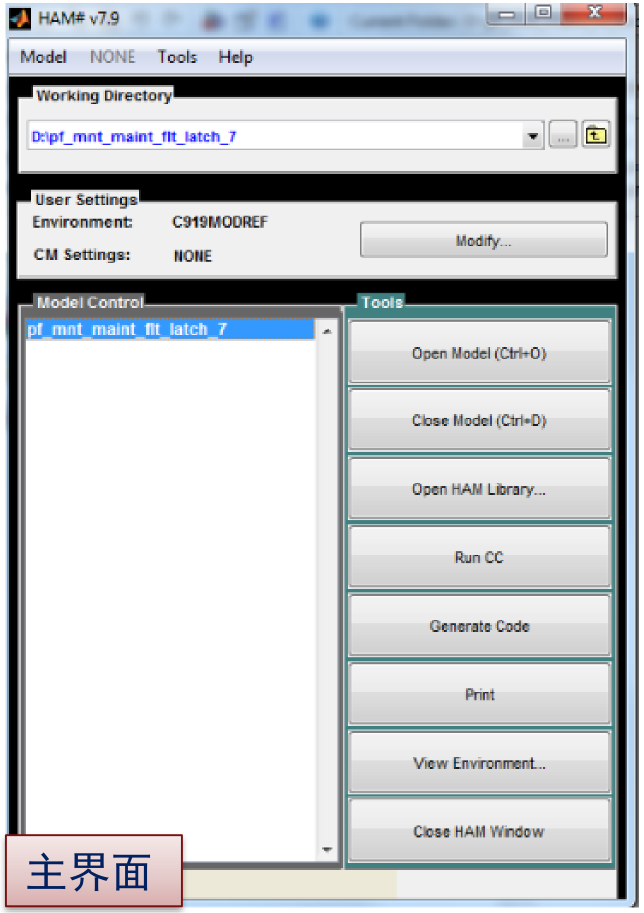


GNC 软件级封装模型



CSU模型

Honeywell-安全子集库



Working Directory

D:\pf_mnt_maint_flt_latch_7

User Settings

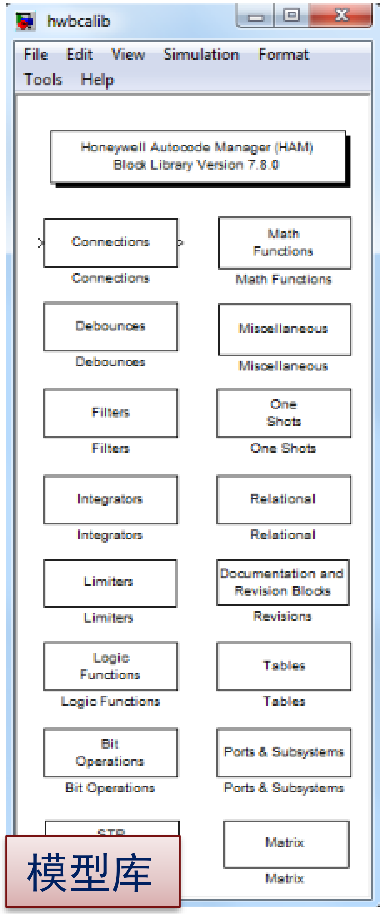
Environment: C919MODREF

CM Settings: NONE

Model Control

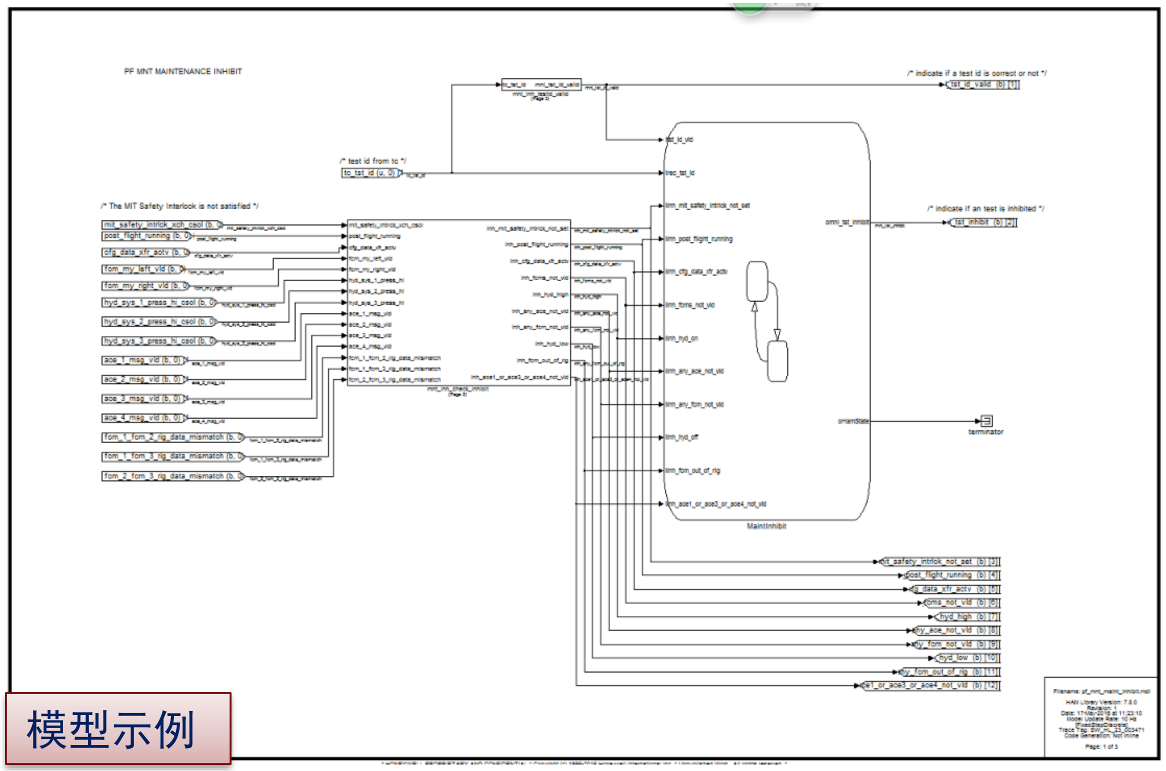
pf_mnt_maint_flt_latch_7

主界面



Honeywell Autocode Manager (HAM)
Block Library Version 7.8.0

模型库


PF MNT MAINTENANCE INHIBIT
/* indicate if a test id is correct or not */
tst_id_valid (b) [1]
/* test id from tc */
tc_tst_id (u, 0)
/* The MIT Safety Interlock is not satisfied */
mit_safety_intrlck_xch_csol (b, 0)
post_flight_running (b, 0)
cfg_data_xfr_actv (b, 0)
fcm_my_left_vld (b, 0)
fcm_my_right_vld (b, 0)
hyd_sys_1_press_hi_csol (b, 0)
hyd_sys_2_press_hi_csol (b, 0)
hyd_sys_3_press_hi_csol (b, 0)
ace_1_msg_vld (b, 0)
ace_2_msg_vld (b, 0)
ace_3_msg_vld (b, 0)
ace_4_msg_vld (b, 0)
fcm_1_fcm_2_rig_data_mismatch (b, 0)
fcm_1_fcm_3_rig_data_mismatch (b, 0)
fcm_2_fcm_3_rig_data_mismatch (b, 0)
mnt_inh_check_inhibit
/* indicate if an test is inhibited */
tst_inhibit (b) [2]
omnt_tst_inhibit
oHamState
terminator
MaintInhibit
mit_safety_intrlck_not_set (b) [3]
post_flight_running (b) [4]
cfg_data_xfr_actv (b) [5]

模型示例

目录

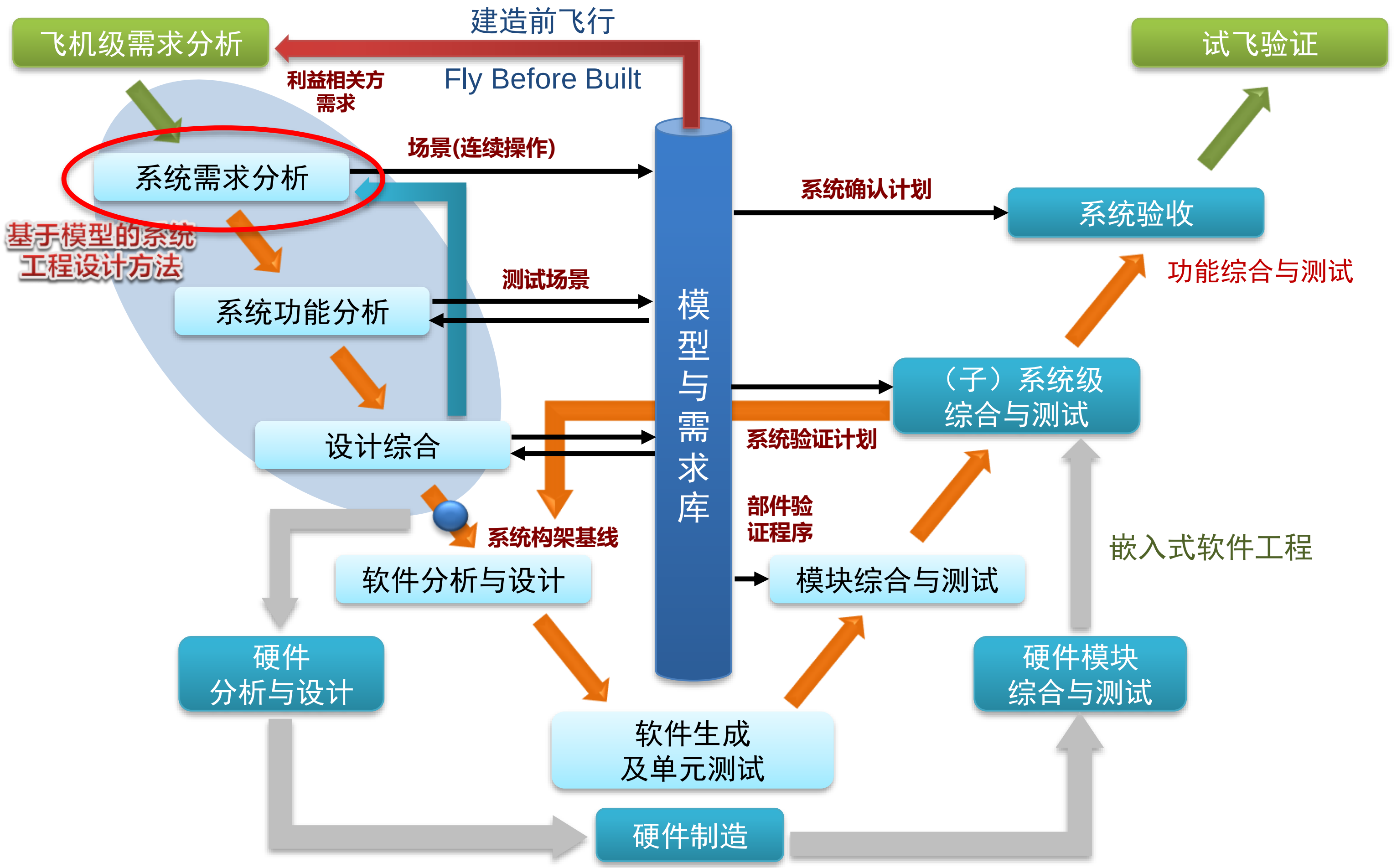
1 技术应用背景

2 MBSE流程

3 工具建设

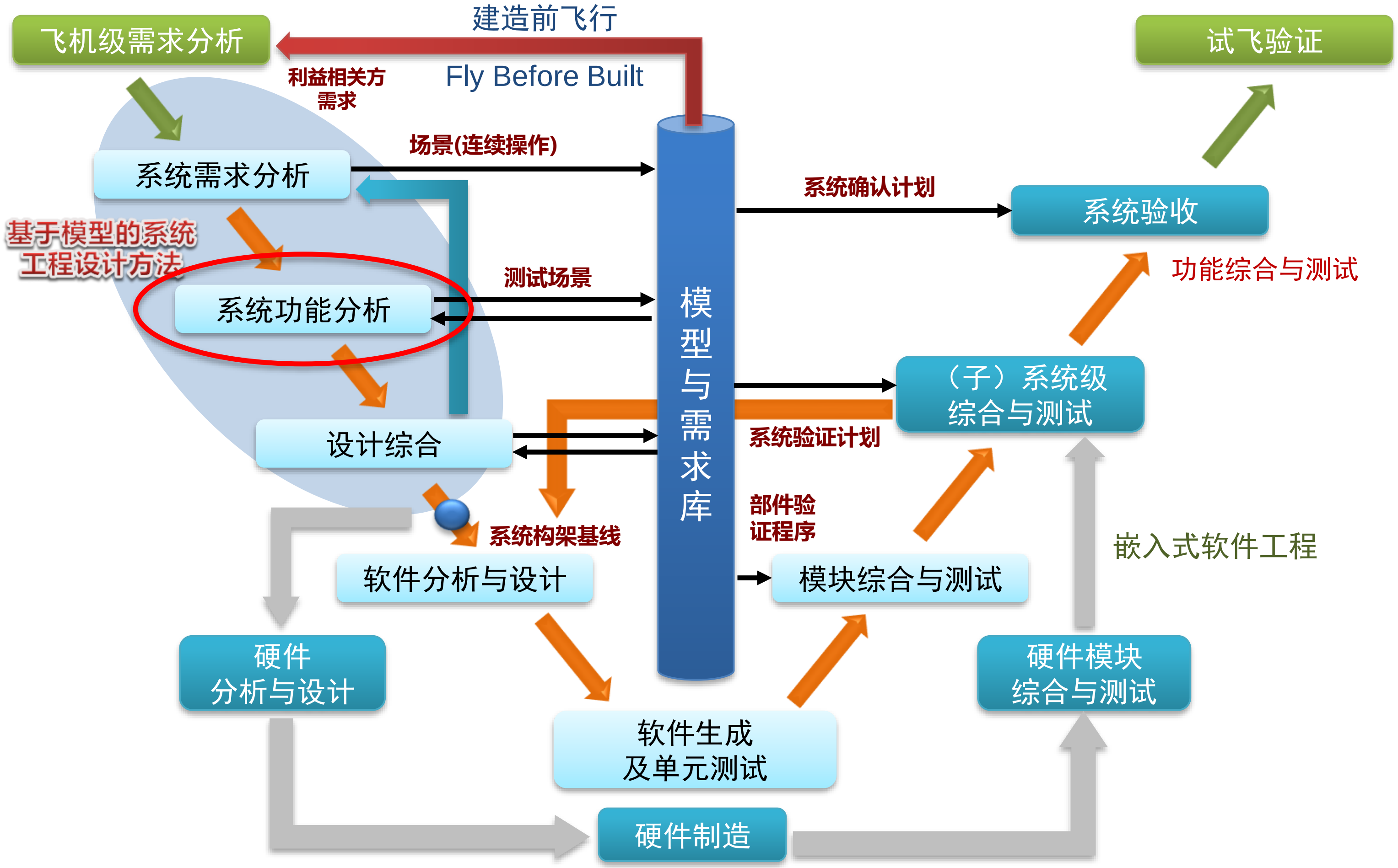
4 应用实例

MBSE——功能性分析



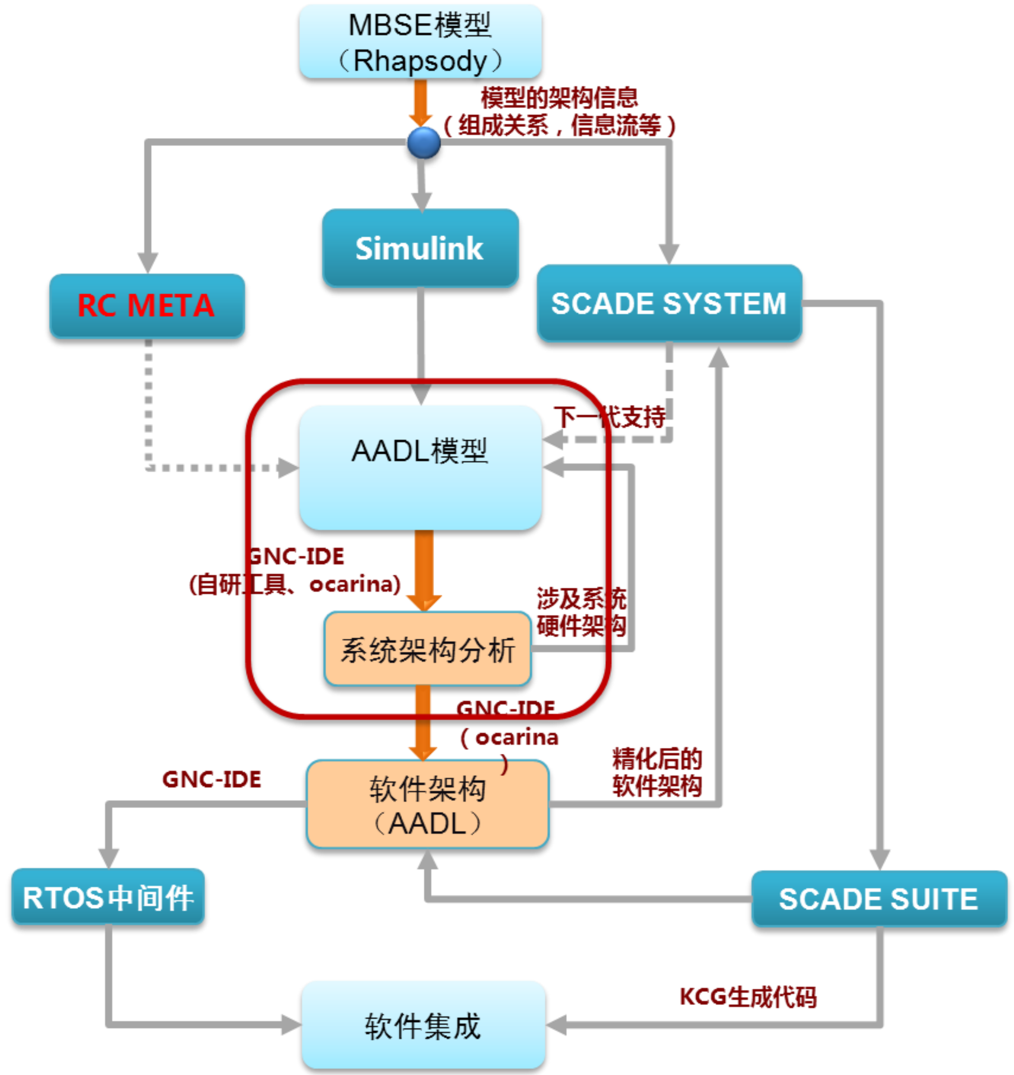
需求分析：采用SysML建模工具进行需求捕获和分析

MBSE——设计综合



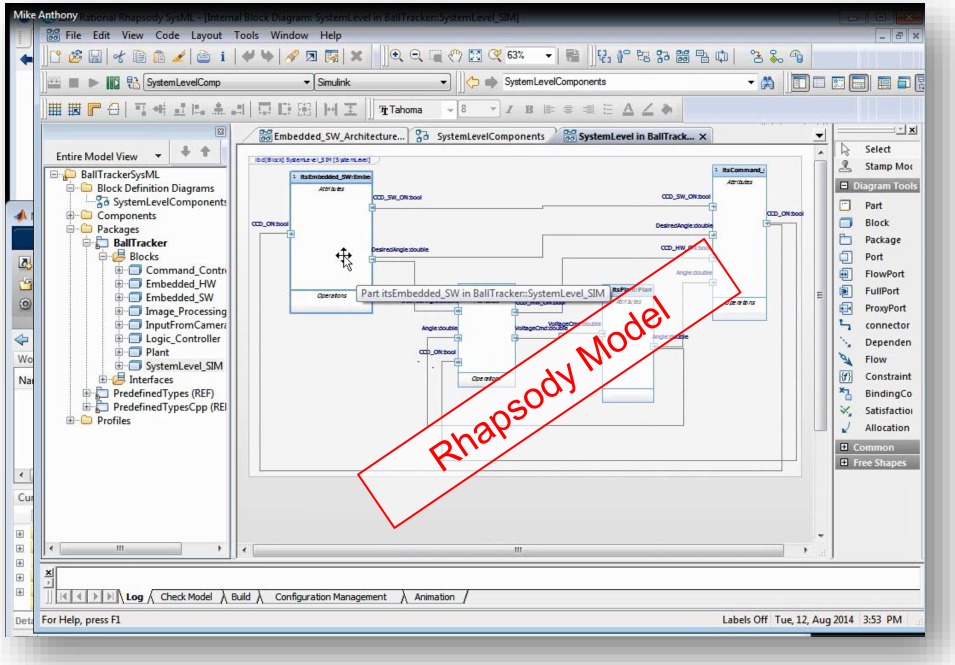
功能分析：利用SysML语言或Simulink分析系统的功能性需求（内部逻辑关系）

MBSE——模型之间的传递关系

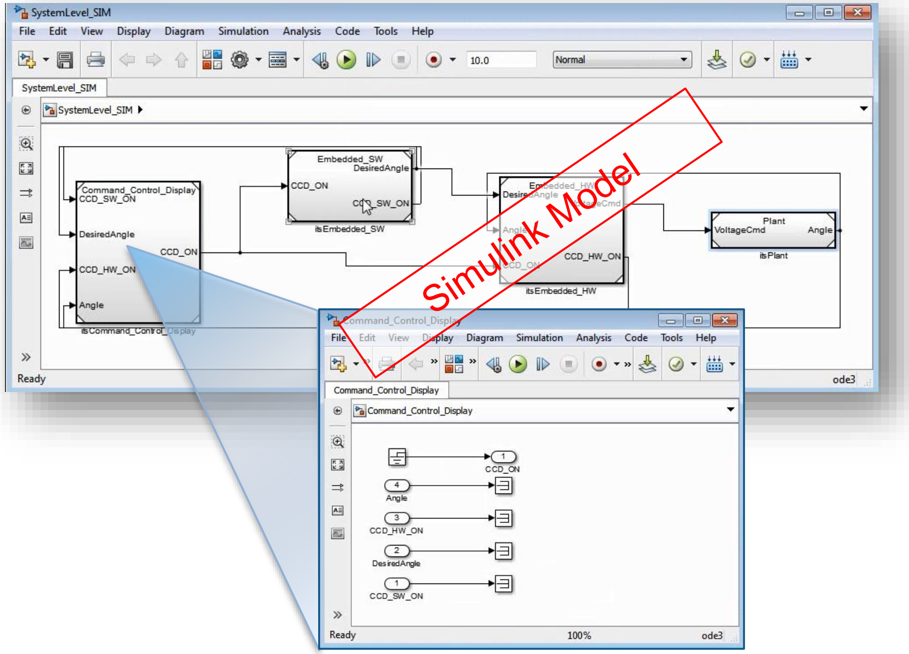


不同模型间的转换关系

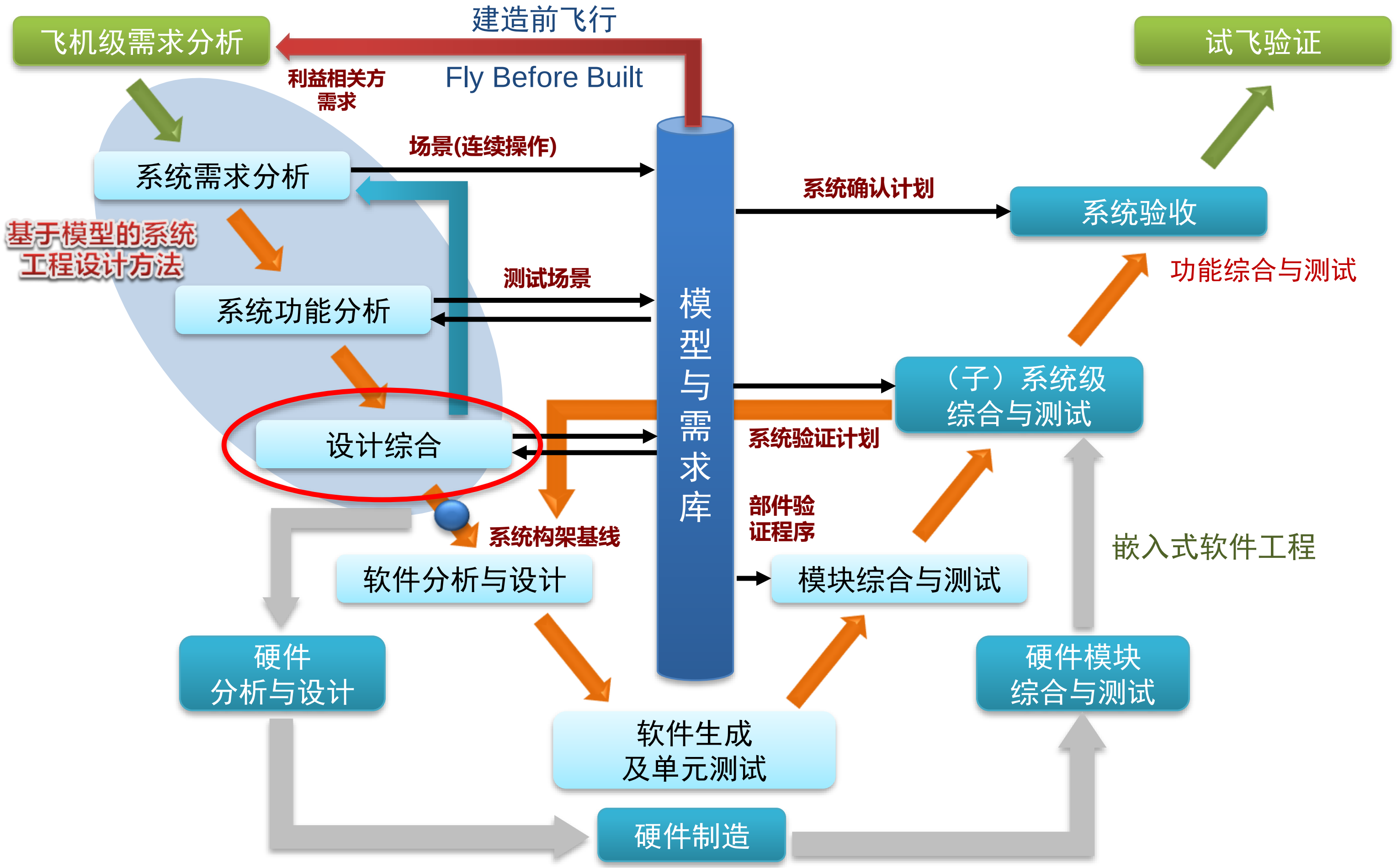
Rhapsody模型与Simulink模型间的转换



Automatic import

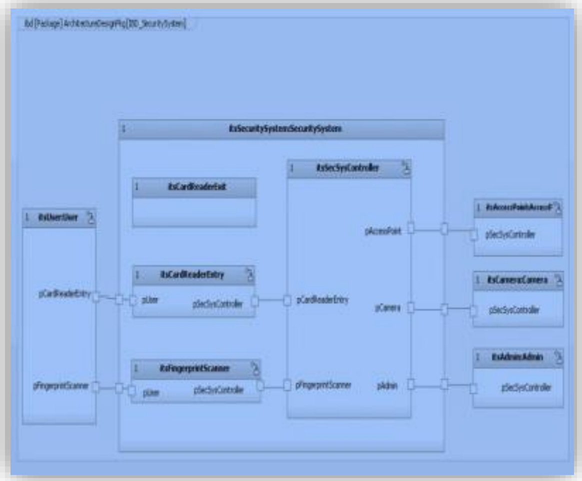


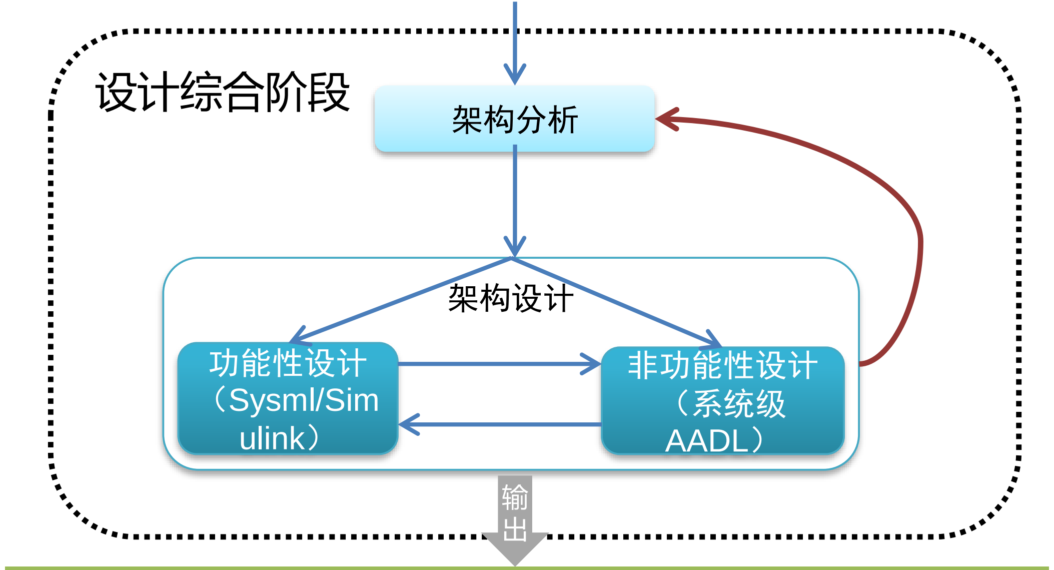
MBSE——设计综合

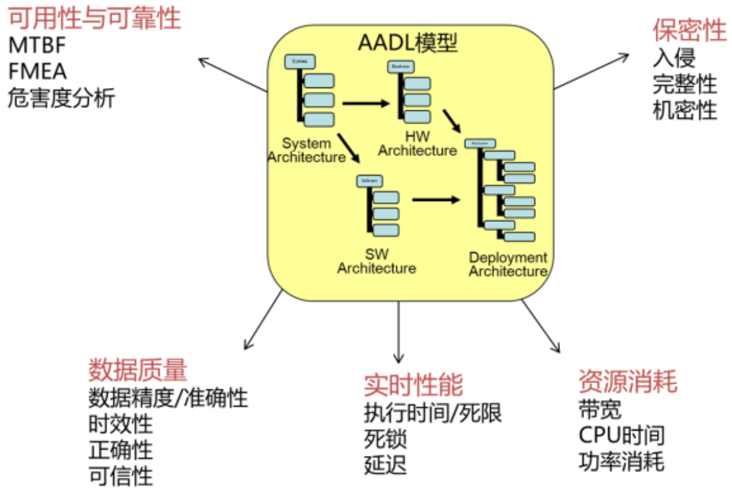


建造前飞行
Fly Before Built
飞机级需求分析
利益相关方需求
试飞验证
系统需求分析
场景(连续操作)
系统确认计划
系统验收
基于模型的系统工程设计方法
功能综合与测试
测试场景
系统功能分析
模型与需求库
（子）系统级综合与测试
设计综合
系统验证计划
部件验证程序
系统构架基线
嵌入式软件工程
软件分析与设计
模块综合与测试
硬件分析与设计
硬件模块综合与测试
软件生成及单元测试
硬件制造

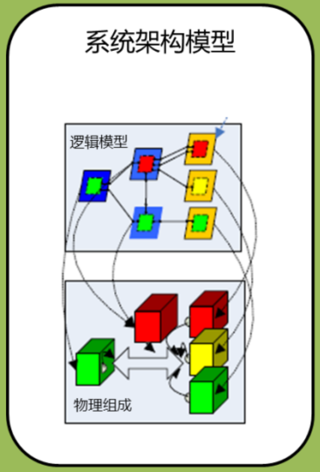
MBSE——功能性与非功能性设计

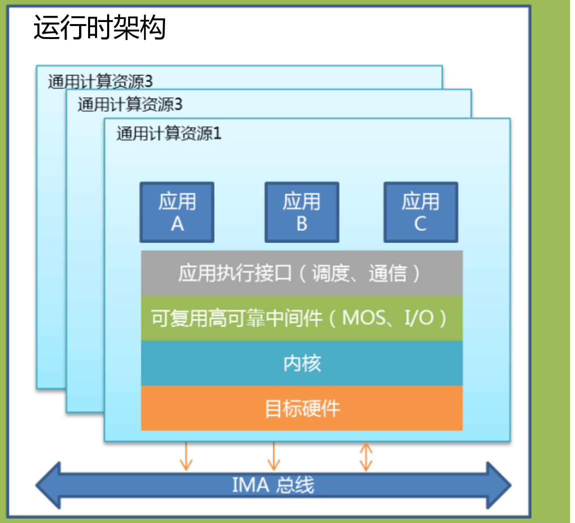




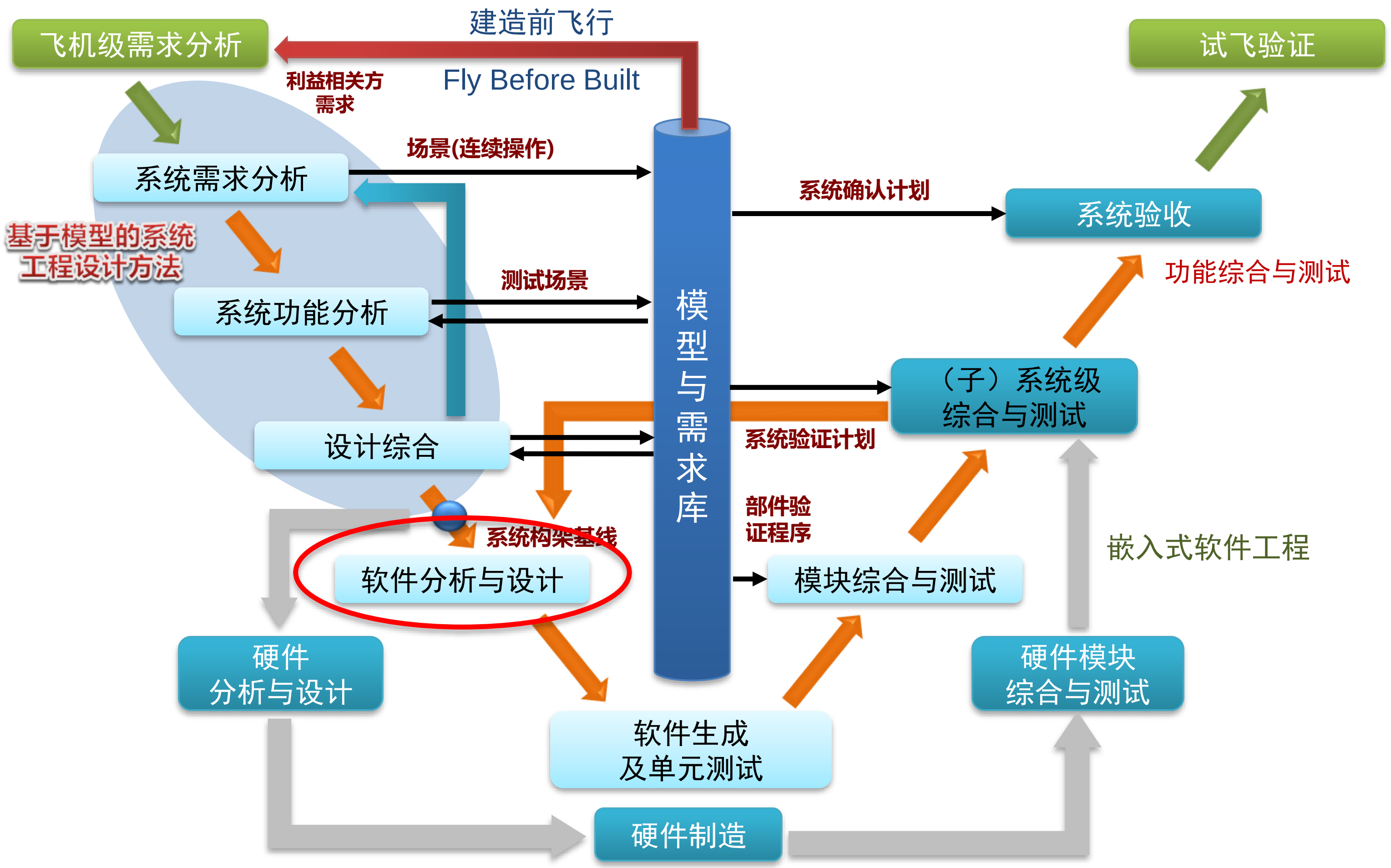


阶段产品

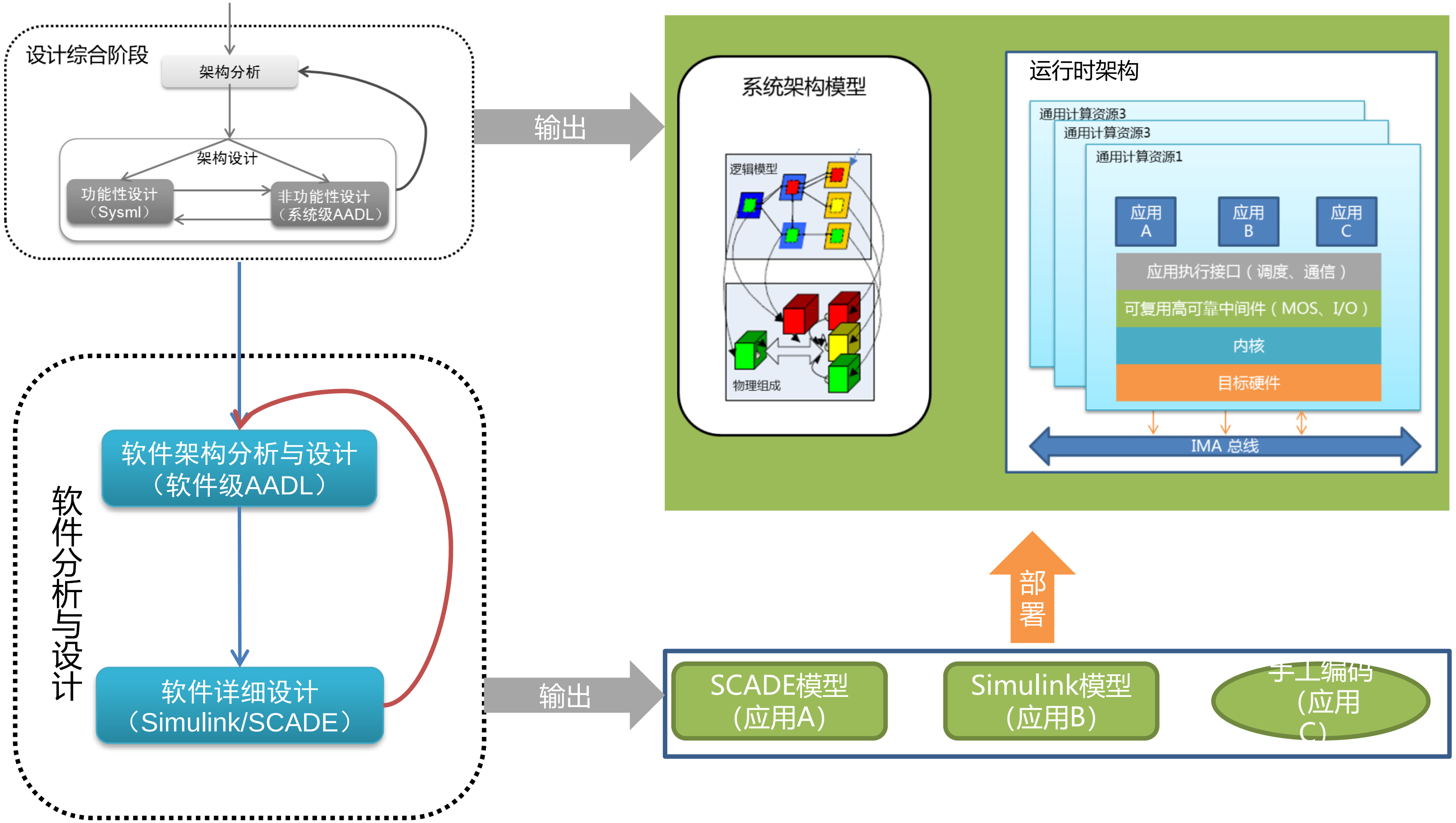




MBSE——软件分析与设计



MBSE——软件分析与设计



设计综合阶段

架构分析

架构设计

功能性设计（Sysml）

非功能性设计（系统级AADL）

输出

系统架构模型

运行时架构

通用计算资源3

通用计算资源3

通用计算资源1

应用 A

应用 B

应用 C

应用执行接口（调度、通信）

可复用高可靠中间件（MOS、I/O）

内核

目标硬件

IMA 总线

软件分析与设计

软件架构分析与设计（软件级AADL）

软件详细设计（Simulink/SCADE）

输出

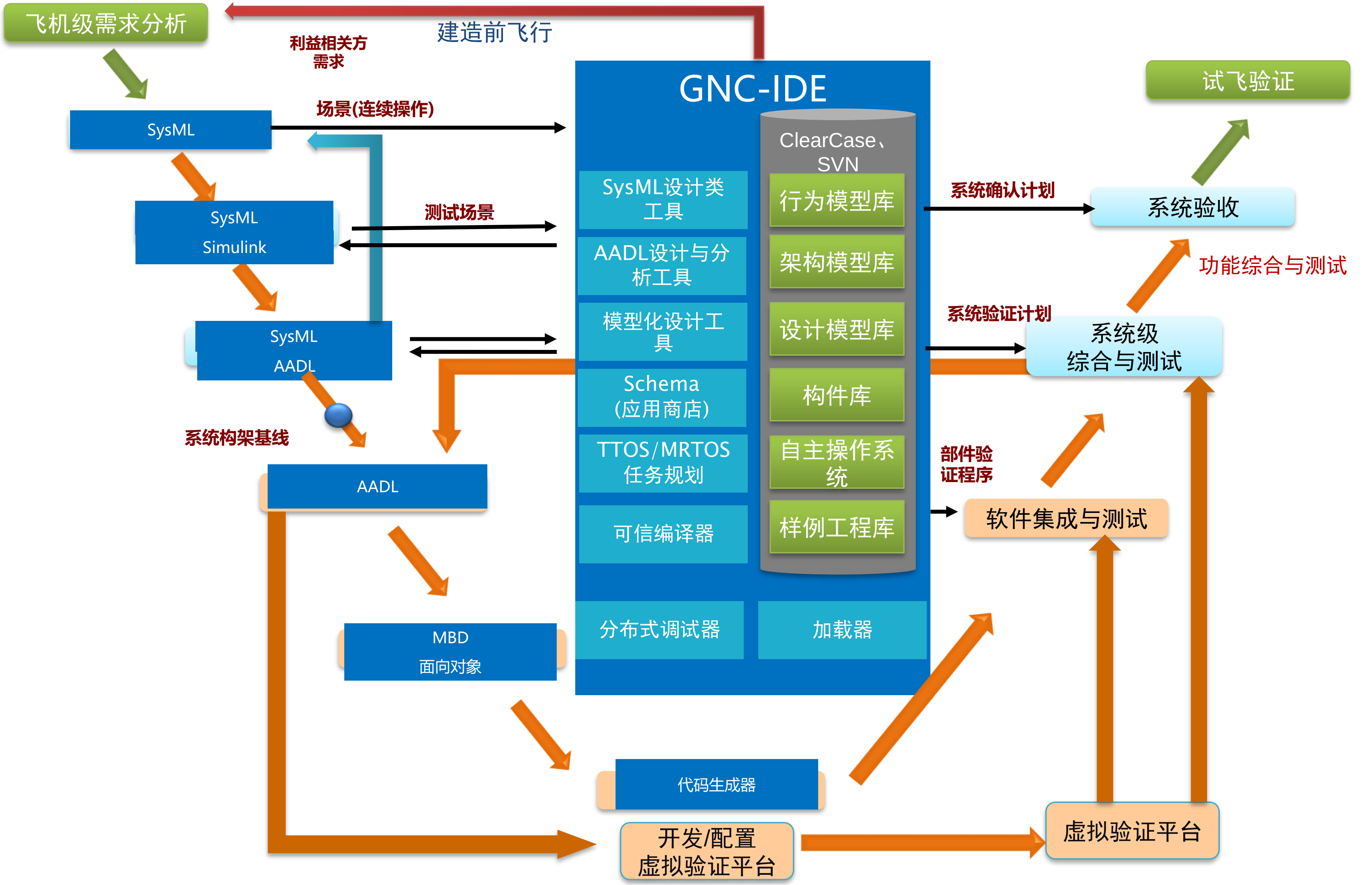
部署

SCADE模型（应用A）

Simulink模型（应用B）

手工编码（应用C）

MBSE工具链



飞机级需求分析
利益相关方需求
建造前飞行
场景(连续操作)
SysML
SysML
Simulink
测试场景
SysML
AADL
系统构架基线
AADL
MBD
面向对象
代码生成器
开发/配置
虚拟验证平台
GNC-IDE
ClearCase、SVN
SysML设计类工具
AADL设计与分析工具
模型化设计工具
Schema
(应用商店)
TTOS/MRTOS
任务规划
可信编译器
行为模型库
架构模型库
设计模型库
构件库
自主操作系统
样例工程库
分布式调试器
加载器
试飞验证
系统确认计划
系统验收
功能综合与测试
系统验证计划
系统级
综合与测试
部件验证程序
软件集成与测试
虚拟验证平台

目录

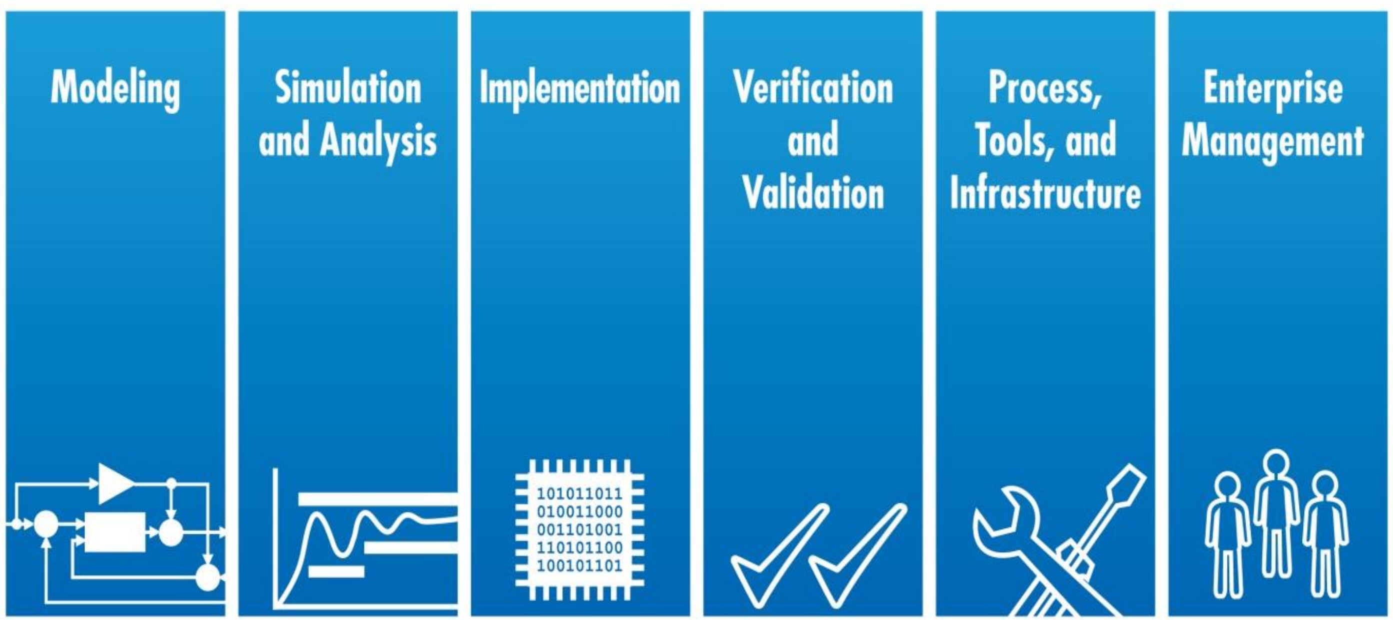
1 技术应用背景

2 MBSE流程

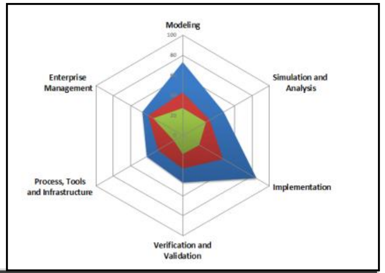
3 工具建设

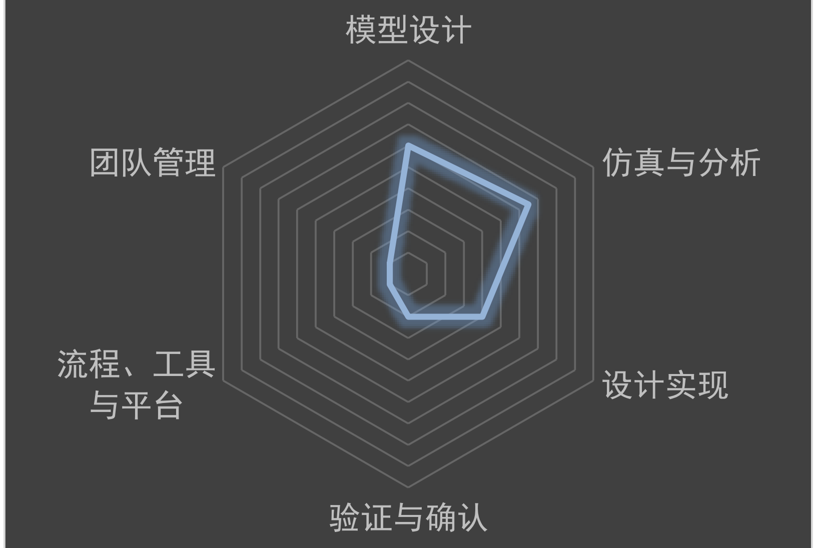
4 应用实例

基于模型设计的企业成熟度标准



模型设计 | 仿真分析 | 代码生成 | 验证确认 | 流程工具 | 组织管理





当前存在的问题分析

模型应用成熟度相关因素

Simulink只用于仿真

工程师使用Simulink仅用于算法仿真，并没有将Simulink平台作为全流程的开发工具，关键是仍没有突破自动代码生成的技术和心理障碍，模型验证更是无从谈起

模型无法在开发中传递

Simulink模型仿真完成后，工程师又忙于琐碎的文档和编码工作，模型无法向下传递，还没有成为知识传递与积累的载体，模型也没有实现标准化的设计

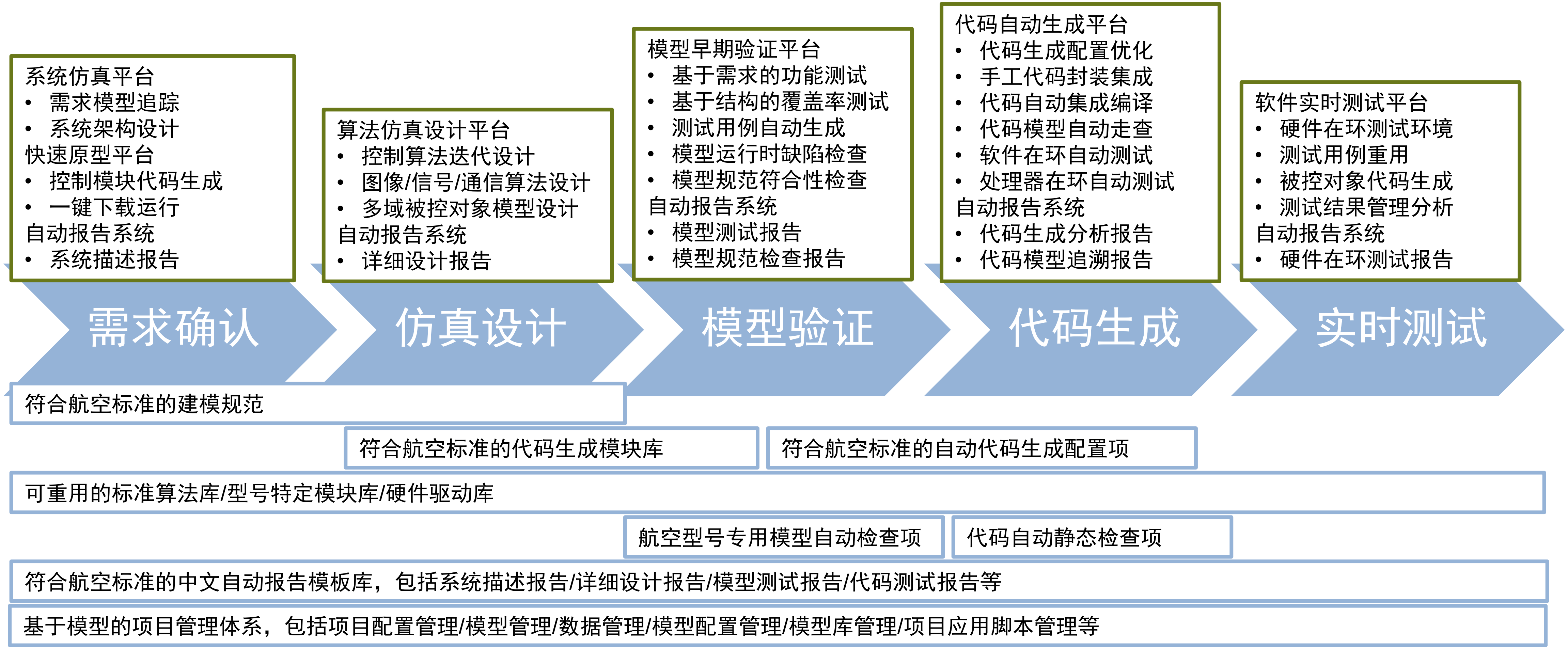
开发流程无法自动化

Simulink提供了建模/仿真/验证/代码生成等一些列工具，但这些工具都相对独立，并没有与特定的流程进行集成，所以导致工程师在使用时没有目标，或者就不知何时使用

缺少模型库和架构参考

在模型设计过程中，不同的功能有不同的设计方法，选择多样化，但是，对于嵌入式系统，什么样的模块是安全的，什么样的架构是安全的，设计人员也不清楚

模型开发环境构建思路与实施



需求确认

系统仿真平台
- 需求模型追踪
- 系统架构设计

快速原型平台
- 控制模块代码生成
- 一键下载运行

自动报告系统
- 系统描述报告

仿真设计

算法仿真设计平台
- 控制算法迭代设计
- 图像/信号/通信算法设计
- 多域被控对象模型设计

自动报告系统
- 详细设计报告

模型验证

模型早期验证平台
- 基于需求的功能测试
- 基于结构的覆盖率测试
- 测试用例自动生成
- 模型运行时缺陷检查
- 模型规范符合性检查

自动报告系统
- 模型测试报告
- 模型规范检查报告

代码生成

代码自动生成平台
- 代码生成配置优化
- 手工代码封装集成
- 代码自动集成编译
- 代码模型自动走查
- 软件在环自动测试
- 处理器在环自动测试

自动报告系统
- 代码生成分析报告
- 代码模型追溯报告

实时测试

软件实时测试平台
- 硬件在环测试环境
- 测试用例重用
- 被控对象代码生成
- 测试结果管理分析

自动报告系统
- 硬件在环测试报告

符合航空标准的建模规范

符合航空标准的代码生成模块库

符合航空标准的自动代码生成配置项

可重用的标准算法库/型号特定模块库/硬件驱动库

航空型号专用模型自动检查项

代码自动静态检查项

符合航空标准的中文自动报告模板库，包括系统描述报告/详细设计报告/模型测试报告/代码测试报告等

基于模型的项目管理体系，包括项目配置管理/模型管理/数据管理/模型配置管理/模型库管理/项目应用脚本管理等

模型开发环境 —— 关键技术

黑盒转白盒

性能优化

高安全代码生成机制研究

- 高安全可靠代码生成原理解析
- 代码生成配置解析
- 模型检查项解析
- 可复用手工编码反向生成机制分析
- 基于EC的代码生成对比分析

高安全系统模型设计规范制定

- NASA/Honeywell/MATHWORKS等航空航天企业模型设计和代码生成规范研究
- 汽车行业成熟模型设计标准研究
- 飞控所现有模型设计经验总结
- 模型规范的自动化检查与评判

符合DO-178B/C的基于模型项目管理体系研究

- MBD符合DO-178流程制定
- 符合DO-178的项目模板设计
- 基于模型的DO文档梳理
- 功能自动化脚本开发

高安全代码自动生成配置项研究

- 以安全子集库生成代码为基准制定高安全代码生成配置参数
- 制定高安全代码生成的技术策略和实现过程，确保模型和代码的一致性
- 通过配置项的设计保证生成代码符合MISA C标准

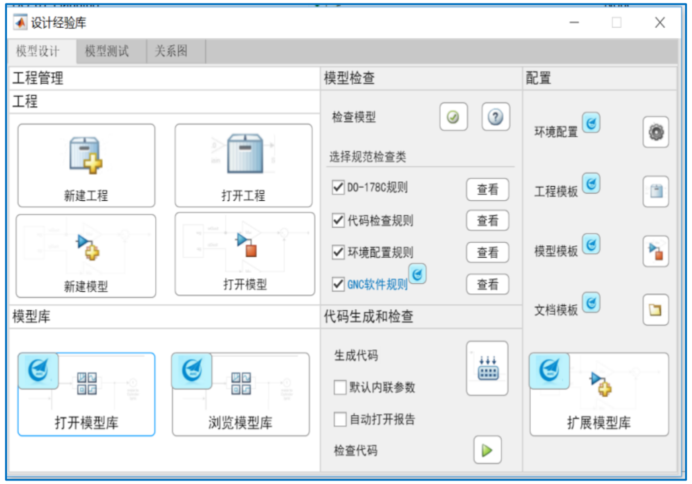
高安全代码模型库测试

- 模型库单一模块单元级测试
- 子系统级别代码生成对比分析
- 系统级别代码生成对比分析

模型开发环境 —— 核心内容

系统主体框架开发及代码生成

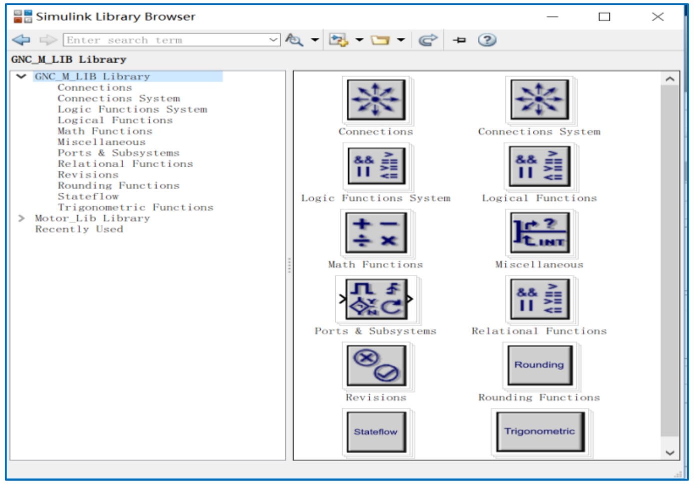
- ➢ 完成MDE流程设计与系统功能划分
- ➢ 完成主体框架设计与软件界面开发
- ➢ 实现了功能应用的独立开发与集成



☑ DO-178C规则
☑ 代码检查规则
☑ 环境配置规则
☑ GNC软件规则
□ 默认内联参数
□ 自动打开报告

高安全代码生成模型库开发

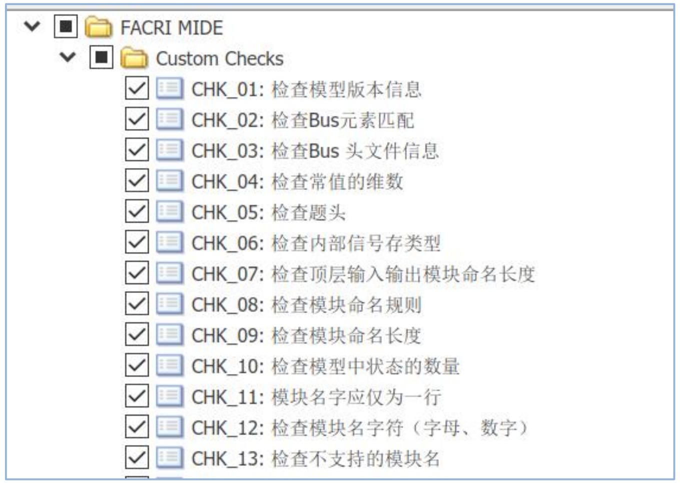
- ➢ 160多个库模块
- ➢ 自建部分伺服软件二级模块库



GNC_M_LIB Library
- GNC_M_LIB Library
  - Connections
  - Connections System
  - Logic Functions System
  - Logical Functions
  - Math Functions
  - Miscellaneous
  - Ports & Subsystems
  - Relational Functions
  - Revisions
  - Rounding Functions
  - Stateflow
  - Trigonometric Functions
- Motor_Lib Library
- Recently Used

模型设计与代码生成规范建立及自动化检查项开发

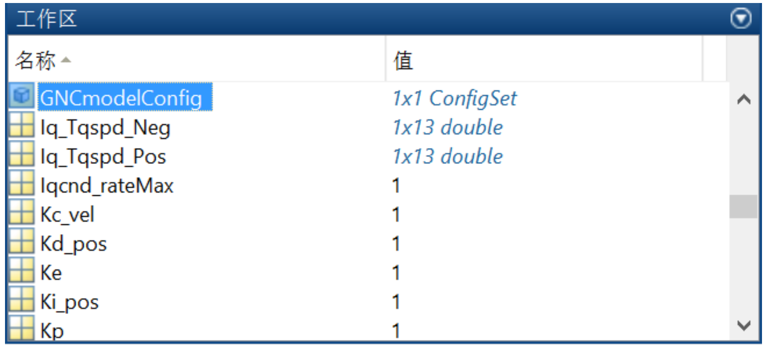
- ➢ 参考Honeywell/NASA/Mathworks建模规范，构建 FACRI自有建模规范



- ■ FACRI MIDE
  - ■ Custom Checks
    - ☑ CHK_01: 检查模型版本信息
    - ☑ CHK_02: 检查Bus元素匹配
    - ☑ CHK_03: 检查Bus 头文件信息
    - ☑ CHK_04: 检查常值的维数
    - ☑ CHK_05: 检查题头
    - ☑ CHK_06: 检查内部信号存类型
    - ☑ CHK_07: 检查顶层输入输出模块命名长度
    - ☑ CHK_08: 检查模块命名规则
    - ☑ CHK_09: 检查模块命名长度
    - ☑ CHK_10: 检查模型中状态的数量
    - ☑ CHK_11: 模块名字应仅为一行
    - ☑ CHK_12: 检查模块名字符（字母、数字）
    - ☑ CHK_13: 检查不支持的模块名

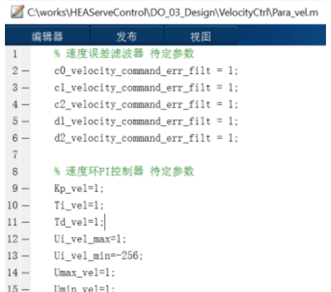
系统主体框架开发及代码生成

- ➢ 完成MDE流程设计与系统功能划分
- ➢ 完成主体框架设计与软件界面开发
- ➢ 实现了功能应用的独立开发与集成



工作区

名称 ▲	值
GNCmodelConfig	1x1 ConfigSet
Iq_Tqspd_Neg	1x13 double
Iq_Tqspd_Pos	1x13 double
Iqcnd_rateMax	1
Kc_vel	1
Kd_pos	1
Ke	1
Ki_pos	1
Kp	1



C:\works\HEAServeControl\DO_03_Design\VelocityCtrl\Para_vel.m

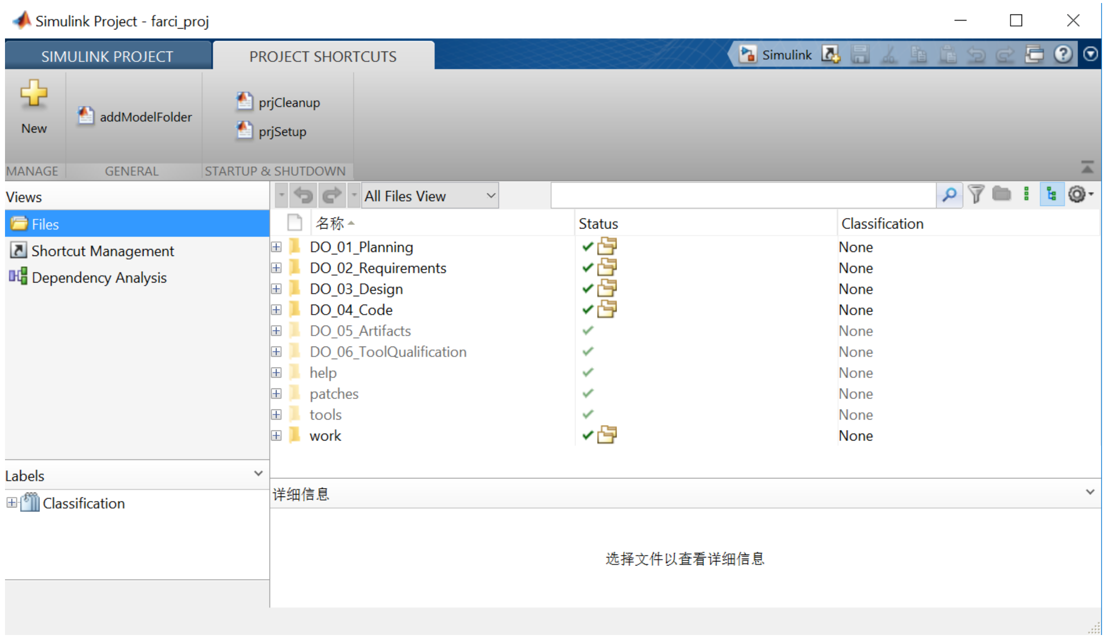
```
% 速度误差滤波器 待定参数
c0_velocity_command_err_filt = 1;
c1_velocity_command_err_filt = 1;
c2_velocity_command_err_filt = 1;
d1_velocity_command_err_filt = 1;
d2_velocity_command_err_filt = 1;

% 速度环PI控制器 待定参数
Kp_vel=1;
Ti_vel=1;
Td_vel=1;
Ui_vel_max=1;
Ui_vel_min=-256;
Umax_vel=1;
Umin_vel=1;

Kc_vel = 1;
```

独立的配置和参数文件

单模型开发->工程开发，集成SVN/GIT

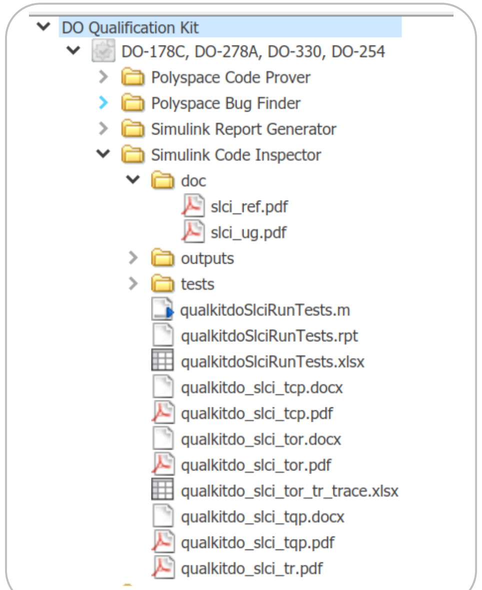


Simulink Project - farci_proj

名称 ▲	Status	Classification
DO_01_Planning	✓	None
DO_02_Requirements	✓	None
DO_03_Design	✓	None
DO_04_Code	✓	None
DO_05_Artifacts	✓	None
DO_06_ToolQualification	✓	None
help	✓	None
patches	✓	None
tools	✓	None
work	✓	None

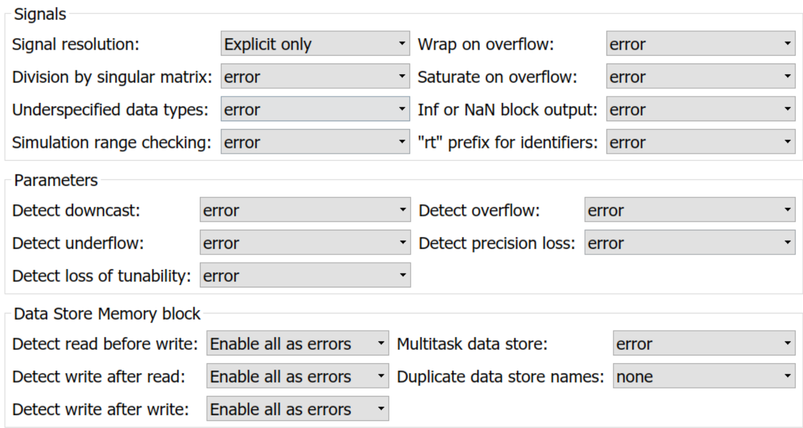
选择文件以查看详细信息

用可认证工具Simulink Code Inspector(TQL-4/TQL-5)作为代码检查工具



- DO Qualification Kit
  - DO-178C, DO-278A, DO-330, DO-254
    - Polyspace Code Prover
    - Polyspace Bug Finder
    - Simulink Report Generator
    - Simulink Code Inspector
      - doc
        - slci_ref.pdf
        - slci_ug.pdf
      - outputs
      - tests
      - qualkitdoSlciRunTests.m
      - qualkitdoSlciRunTests.rpt
      - qualkitdoSlciRunTests.xlsx
      - qualkitdo_slci_tcp.docx
      - qualkitdo_slci_tcp.pdf
      - qualkitdo_slci_tor.docx
      - qualkitdo_slci_tor.pdf
      - qualkitdo_slci_tor_tr_trace.xlsx
      - qualkitdo_slci_tqp.docx
      - qualkitdo_slci_tqp.pdf
      - qualkitdo_slci_tr.pdf

更严格的建模环境



Signals

Signal resolution:	Explicit only	Wrap on overflow:	error
Division by singular matrix:	error	Saturate on overflow:	error
Underspecified data types:	error	Inf or NaN block output:	error
Simulation range checking:	error	"rt" prefix for identifiers:	error

Parameters

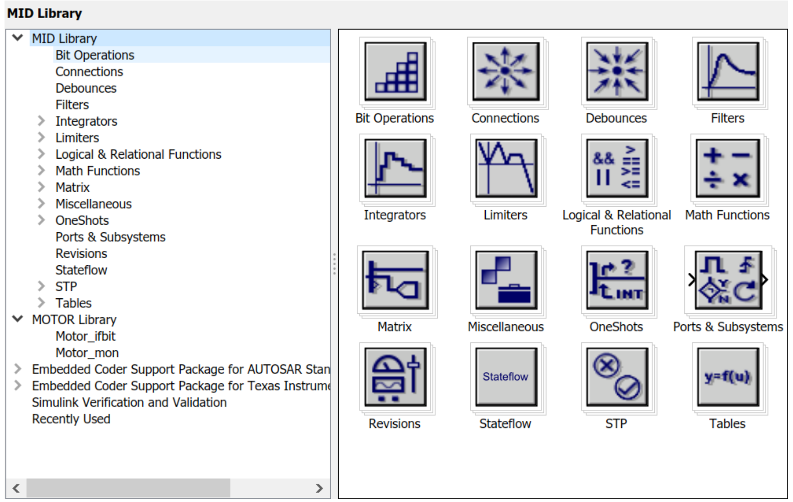
Detect downcast:	error	Detect overflow:	error
Detect underflow:	error	Detect precision loss:	error
Detect loss of tunability:	error		

Data Store Memory block

Detect read before write:	Enable all as errors	Multitask data store:	error
Detect write after read:	Enable all as errors	Duplicate data store names:	none
Detect write after write:	Enable all as errors		

高安全代码生成模型库

- 160多个库模块
- 自建部分伺服软件二级模块库



MID Library

- MID Library
  - Bit Operations
  - Connections
  - Debounces
  - Filters
  - Integrators
  - Limiters
  - Logical & Relational Functions
  - Math Functions
  - Matrix
  - Miscellaneous
  - OneShots
  - Ports & Subsystems
  - Revisions
  - Stateflow
  - STP
  - Tables
- MOTOR Library
  - Motor_ifbit
  - Motor_mon
- Embedded Coder Support Package for AUTOSAR Stan
- Embedded Coder Support Package for Texas Instrume
- Simulink Verification and Validation
- Recently Used

对模块配置参数加以限制，如查表模块

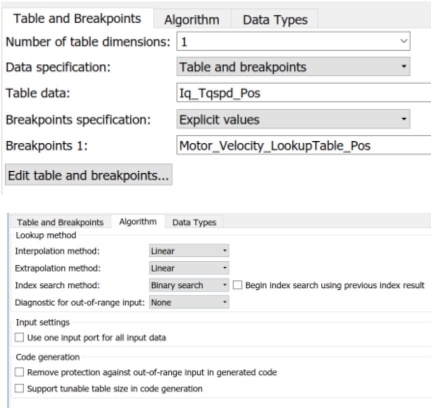


Table and Breakpoints | Algorithm | Data Types

Number of table dimensions: 1

Data specification: Table and breakpoints

Table data: Iq_Tqspd_Pos

Breakpoints specification: Explicit values

Breakpoints 1: Motor_Velocity_LookupTable_Pos

Edit table and breakpoints...

Table and Breakpoints | Algorithm | Data Types

Lookup method

Interpolation method: Linear

Extrapolation method: Linear

Index search method: Binary search □ Begin index search using previous index result

Diagnostic for out-of-range input: None

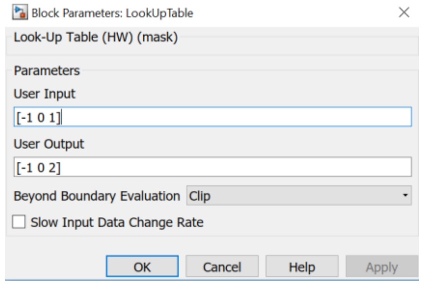
Input settings

□ Use one input port for all input data

Code generation

□ Remove protection against out-of-range input in generated code

□ Support tunable table size in code generation



Block Parameters: LookUpTable

Look-Up Table (HW) (mask)

Parameters

User Input

[-1 0 1]

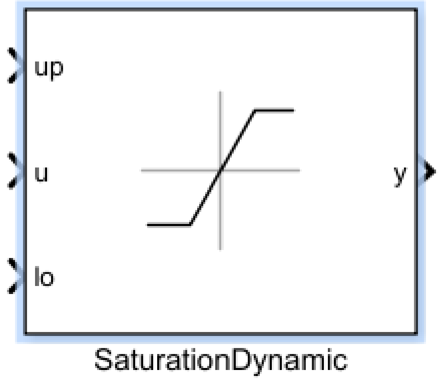
User Output

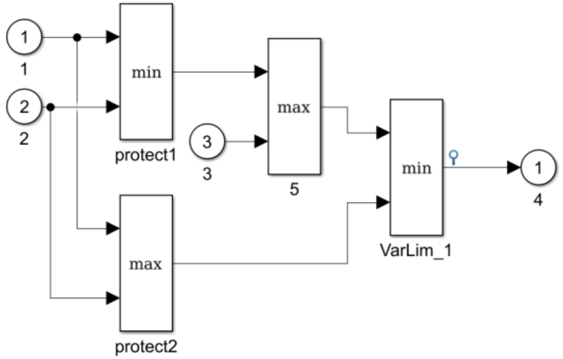
[-1 0 2]

Beyond Boundary Evaluation Clip

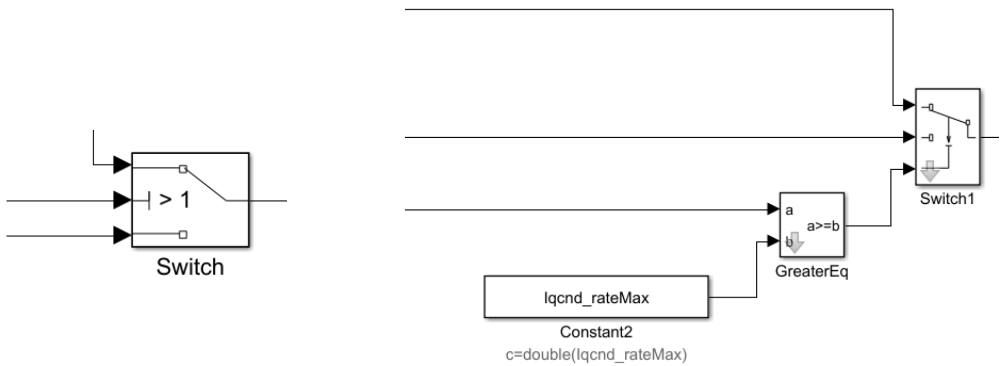
□ Slow Input Data Change Rate

用简单模块组合白盒实现复杂功能，如限幅模块





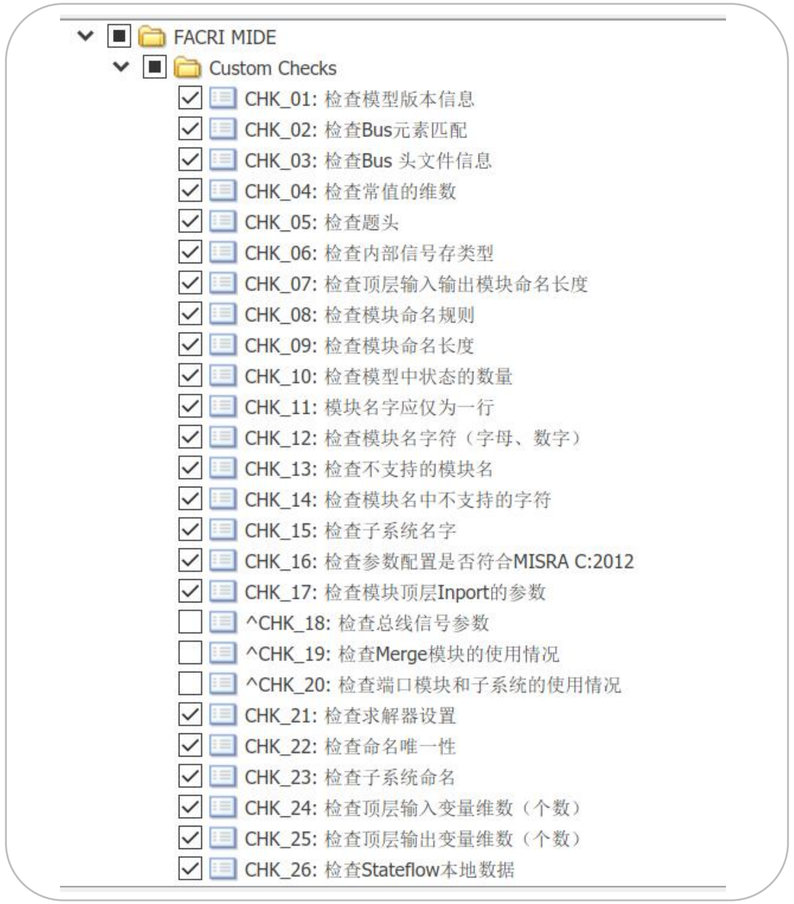
对模块使用加以限制，如Switch模块



建模规范及自动化检查项

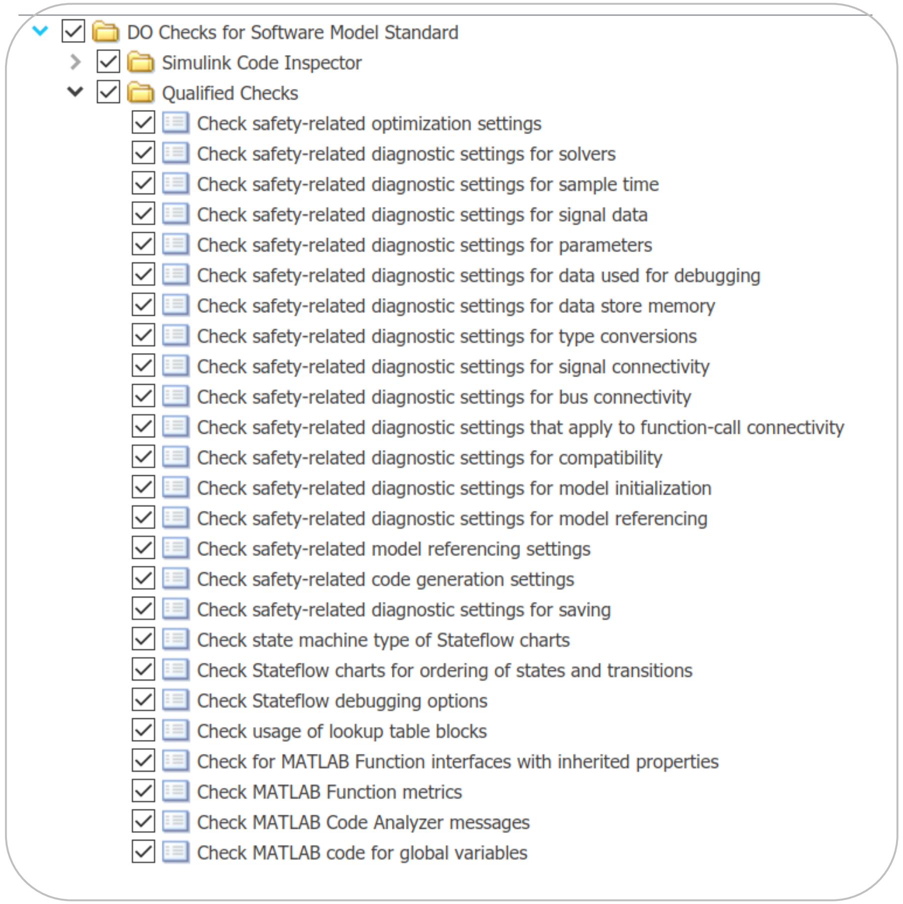
- 参考Honeywell/NASA/Mathworks建模规范，构建 FACRI自有建模规范

自定义规范检查50项



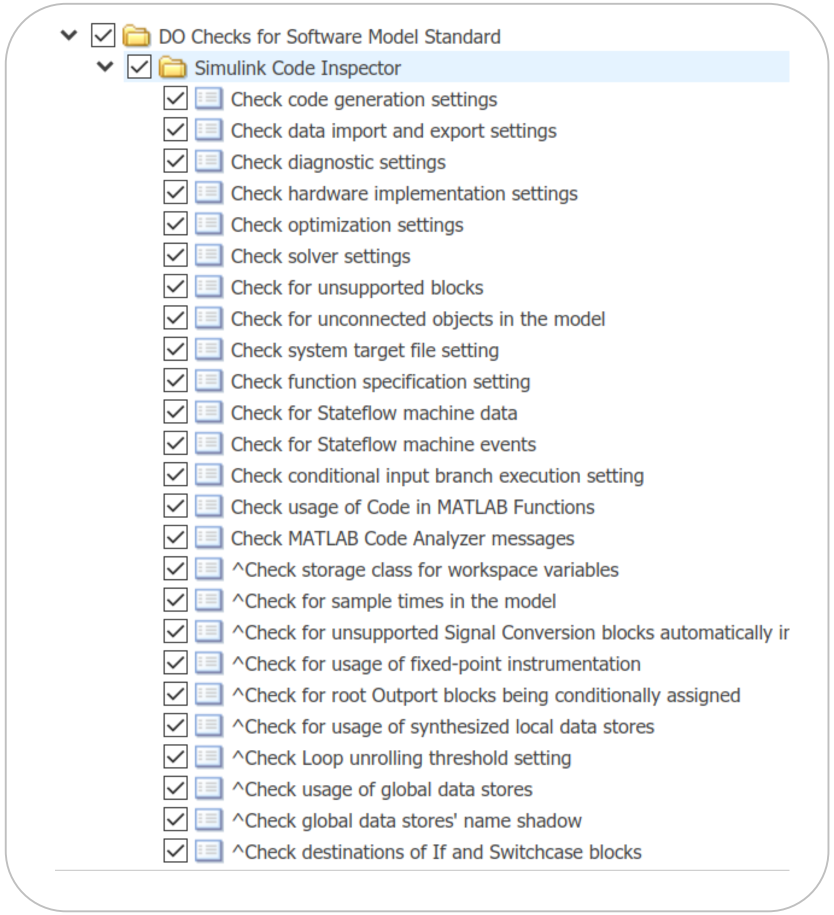
- ■ FACRI MIDE
  - ■ Custom Checks
    - ☑ CHK_01: 检查模型版本信息
    - ☑ CHK_02: 检查Bus元素匹配
    - ☑ CHK_03: 检查Bus 头文件信息
    - ☑ CHK_04: 检查常值的维数
    - ☑ CHK_05: 检查题头
    - ☑ CHK_06: 检查内部信号存类型
    - ☑ CHK_07: 检查顶层输入输出模块命名长度
    - ☑ CHK_08: 检查模块命名规则
    - ☑ CHK_09: 检查模块命名长度
    - ☑ CHK_10: 检查模型中状态的数量
    - ☑ CHK_11: 模块名字应仅为一行
    - ☑ CHK_12: 检查模块名字符（字母、数字）
    - ☑ CHK_13: 检查不支持的模块名
    - ☑ CHK_14: 检查模块名中不支持的字符
    - ☑ CHK_15: 检查子系统名字
    - ☑ CHK_16: 检查参数配置是否符合MISRA C:2012
    - ☑ CHK_17: 检查模块顶层Inport的参数
    - □ ^CHK_18: 检查总线信号参数
    - □ ^CHK_19: 检查Merge模块的使用情况
    - □ ^CHK_20: 检查端口模块和子系统的使用情况
    - ☑ CHK_21: 检查求解器设置
    - ☑ CHK_22: 检查命名唯一性
    - ☑ CHK_23: 检查子系统命名
    - ☑ CHK_24: 检查顶层输入变量维数（个数）
    - ☑ CHK_25: 检查顶层输出变量维数（个数）
    - ☑ CHK_26: 检查Stateflow本地数据

高安全相关检查33项



- ☑ DO Checks for Software Model Standard
  - ☑ Simulink Code Inspector
  - ☑ Qualified Checks
    - ☑ Check safety-related optimization settings
    - ☑ Check safety-related diagnostic settings for solvers
    - ☑ Check safety-related diagnostic settings for sample time
    - ☑ Check safety-related diagnostic settings for signal data
    - ☑ Check safety-related diagnostic settings for parameters
    - ☑ Check safety-related diagnostic settings for data used for debugging
    - ☑ Check safety-related diagnostic settings for data store memory
    - ☑ Check safety-related diagnostic settings for type conversions
    - ☑ Check safety-related diagnostic settings for signal connectivity
    - ☑ Check safety-related diagnostic settings for bus connectivity
    - ☑ Check safety-related diagnostic settings that apply to function-call connectivity
    - ☑ Check safety-related diagnostic settings for compatibility
    - ☑ Check safety-related diagnostic settings for model initialization
    - ☑ Check safety-related diagnostic settings for model referencing
    - ☑ Check safety-related model referencing settings
    - ☑ Check safety-related code generation settings
    - ☑ Check safety-related diagnostic settings for saving
    - ☑ Check state machine type of Stateflow charts
    - ☑ Check Stateflow charts for ordering of states and transitions
    - ☑ Check Stateflow debugging options
    - ☑ Check usage of lookup table blocks
    - ☑ Check for MATLAB Function interfaces with inherited properties
    - ☑ Check MATLAB Function metrics
    - ☑ Check MATLAB Code Analyzer messages
    - ☑ Check MATLAB code for global variables

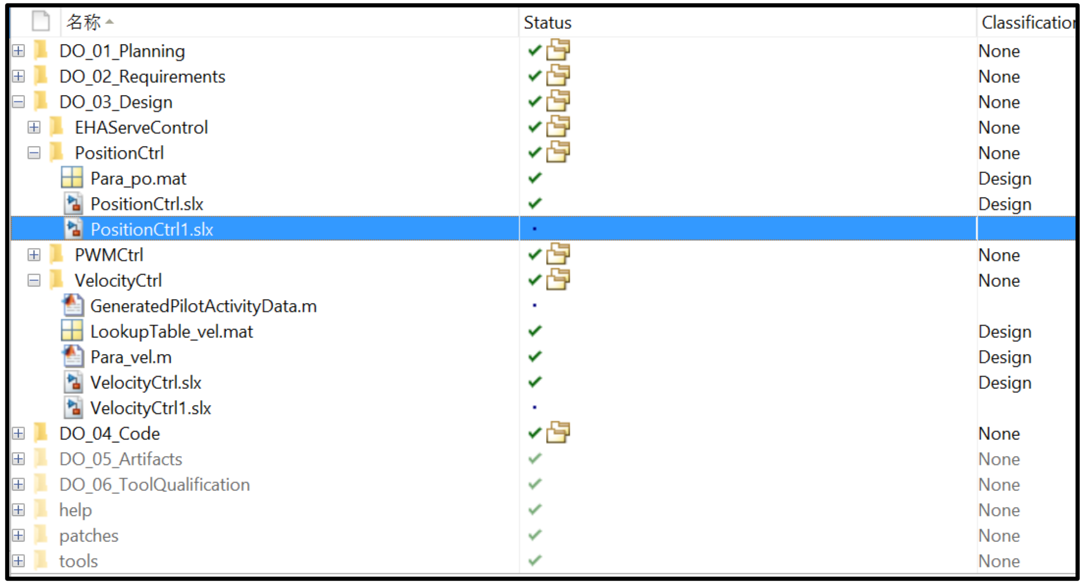
代码模型一致性检查54项



- ☑ DO Checks for Software Model Standard
  - ☑ Simulink Code Inspector
    - ☑ Check code generation settings
    - ☑ Check data import and export settings
    - ☑ Check diagnostic settings
    - ☑ Check hardware implementation settings
    - ☑ Check optimization settings
    - ☑ Check solver settings
    - ☑ Check for unsupported blocks
    - ☑ Check for unconnected objects in the model
    - ☑ Check system target file setting
    - ☑ Check function specification setting
    - ☑ Check for Stateflow machine data
    - ☑ Check for Stateflow machine events
    - ☑ Check conditional input branch execution setting
    - ☑ Check usage of Code in MATLAB Functions
    - ☑ Check MATLAB Code Analyzer messages
    - ☑ ^Check storage class for workspace variables
    - ☑ ^Check for sample times in the model
    - ☑ ^Check for unsupported Signal Conversion blocks automatically ir
    - ☑ ^Check for usage of fixed-point instrumentation
    - ☑ ^Check for root Outport blocks being conditionally assigned
    - ☑ ^Check for usage of synthesized local data stores
    - ☑ ^Check Loop unrolling threshold setting
    - ☑ ^Check usage of global data stores
    - ☑ ^Check global data stores' name shadow
    - ☑ ^Check destinations of If and Switchcase blocks

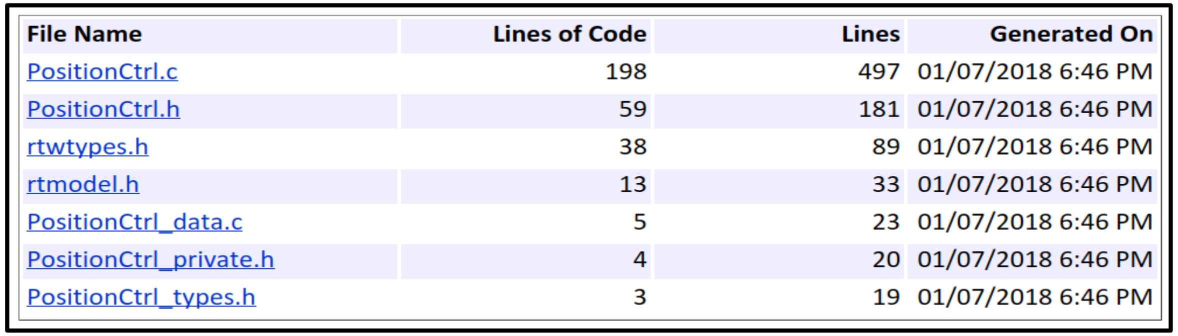
工程应用1——伺服控制算法建模

以伺服控制系统为例，进行高可靠系统的建模



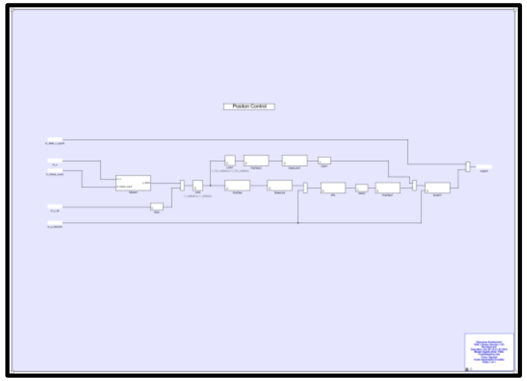
名称	Status	Classificatio
DO_01_Planning	✓	None
DO_02_Requirements	✓	None
DO_03_Design	✓	None
EHAServeControl	✓	None
PositionCtrl	✓	None
Para_po.mat	✓	Design
PositionCtrl.slx	✓	Design
PositionCtrl1.slx	•	
PWMCtrl	✓	None
VelocityCtrl	✓	None
GeneratedPilotActivityData.m	•	
LookupTable_vel.mat	✓	Design
Para_vel.m	✓	Design
VelocityCtrl.slx	✓	Design
VelocityCtrl1.slx	•	
DO_04_Code	✓	None
DO_05_Artifacts	✓	None
DO_06_ToolQualification	✓	None
help	✓	None
patches	✓	None
tools	✓	None

位置环生成C代码，通过代码模型一致性检查

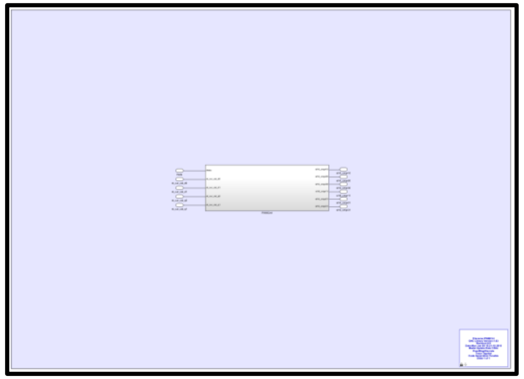


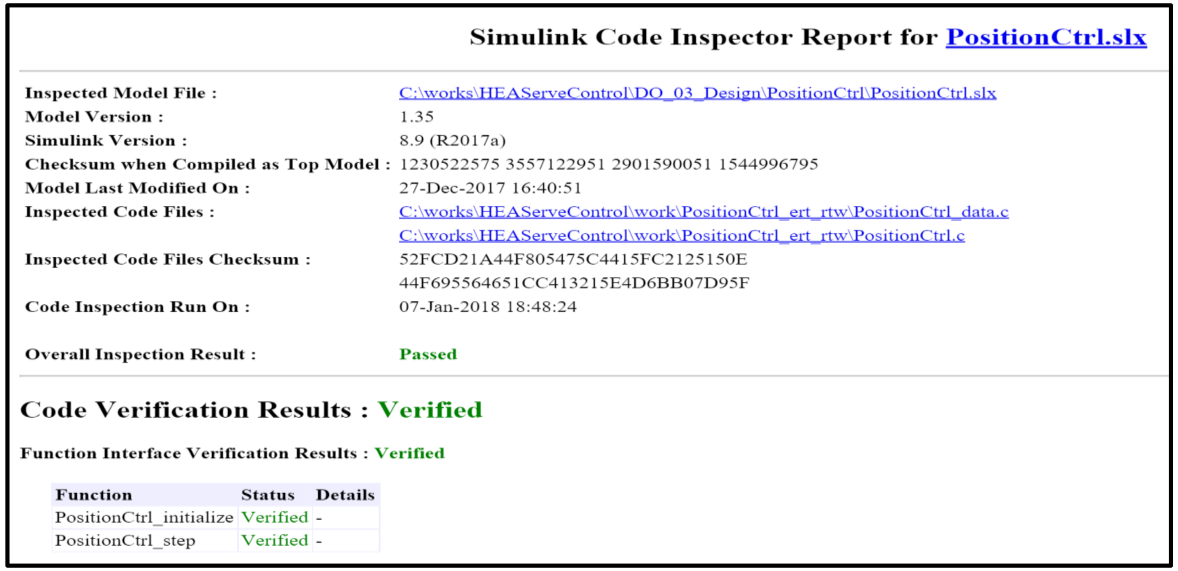
File Name	Lines of Code	Lines	Generated On
PositionCtrl.c	198	497	01/07/2018 6:46 PM
PositionCtrl.h	59	181	01/07/2018 6:46 PM
rtwtypes.h	38	89	01/07/2018 6:46 PM
rtmodel.h	13	33	01/07/2018 6:46 PM
PositionCtrl_data.c	5	23	01/07/2018 6:46 PM
PositionCtrl_private.h	4	20	01/07/2018 6:46 PM
PositionCtrl_types.h	3	19	01/07/2018 6:46 PM

位置环和速度环，纯Simulink模型



PWM控制环，封装的原有C代码





Simulink Code Inspector Report for PositionCtrl.slx

Inspected Model File :	C:\works\HEAServeControl\DO_03_Design\PositionCtrl\PositionCtrl.slx
Model Version :	1.35
Simulink Version :	8.9 (R2017a)
Checksum when Compiled as Top Model :	1230522575 3557122951 2901590051 1544996795
Model Last Modified On :	27-Dec-2017 16:40:51
Inspected Code Files :	C:\works\HEAServeControl\work\PositionCtrl_ert_rtw\PositionCtrl_data.c C:\works\HEAServeControl\work\PositionCtrl_ert_rtw\PositionCtrl.c
Inspected Code Files Checksum :	52FCD21A44F805475C4415FC2125150E 44F695564651CC413215E4D6BB07D95F
Code Inspection Run On :	07-Jan-2018 18:48:24
Overall Inspection Result :	Passed

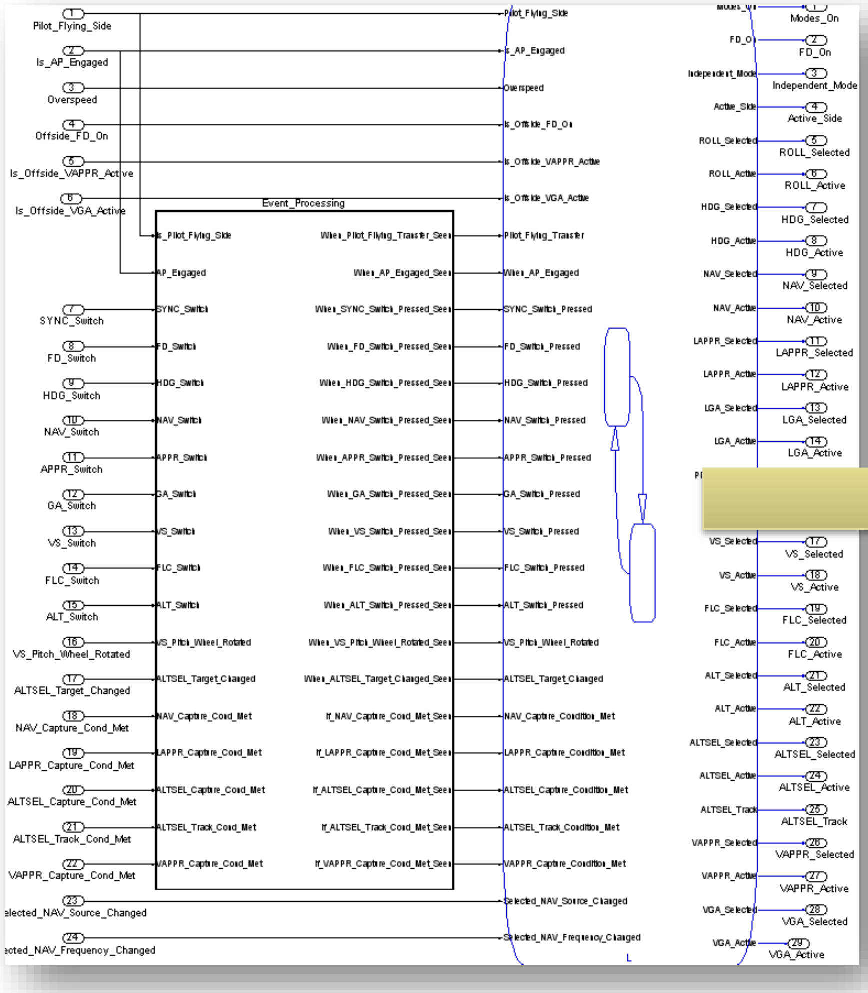
Code Verification Results : Verified

Function Interface Verification Results : Verified

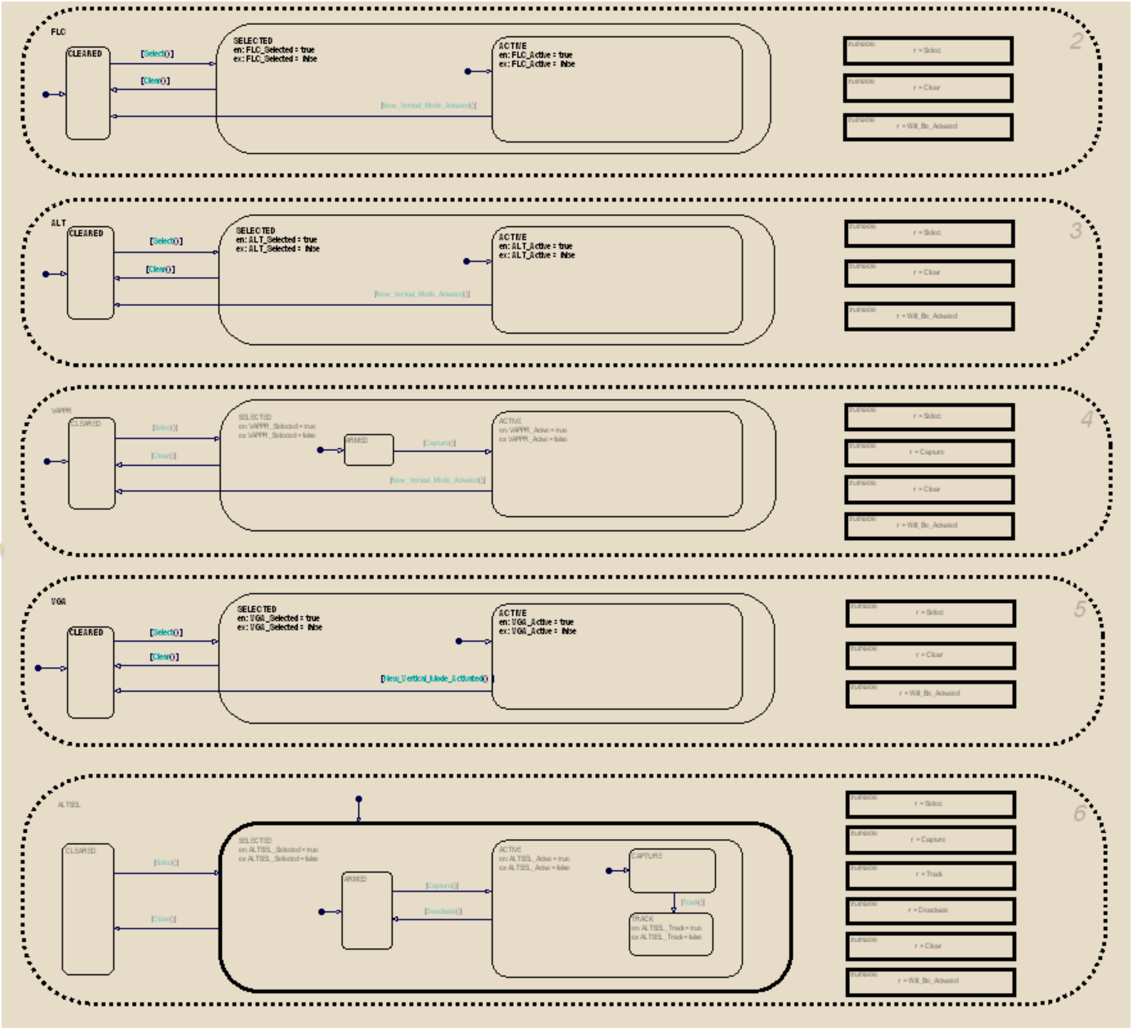
Function	Status	Details
PositionCtrl_initialize	Verified	-
PositionCtrl_step	Verified	-

工程应用2 自动驾驶仪

以自动驾驶仪为例，进行复杂逻辑建模



形成复杂状态逻辑的设计方案



目录

1 技术应用背景

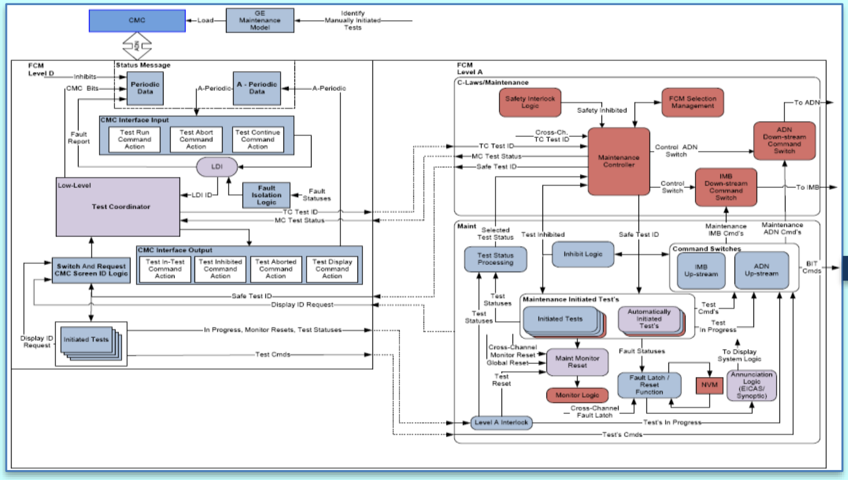
2 MBSE流程

3 工具建设

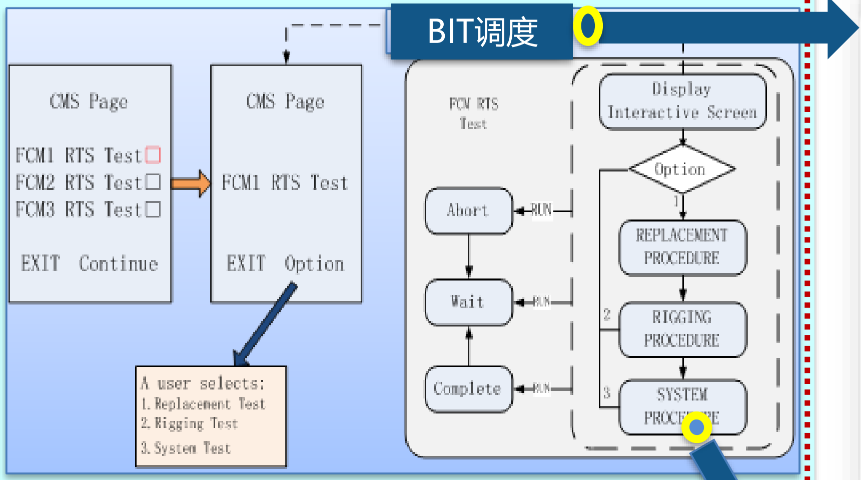
4 应用实例

基于模型设计完成C919项目的核心维护功能调度

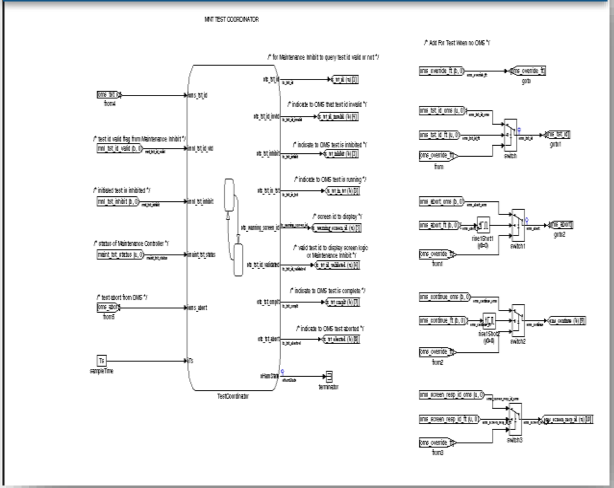
需求分析



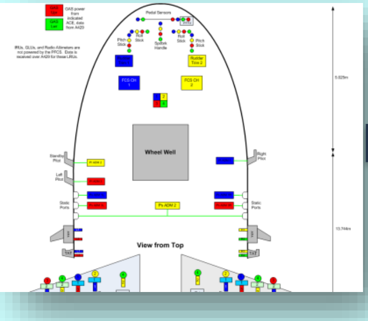
顶层设计



基于状态机的BIT调度



软、硬件处理隔离



IO处理分区

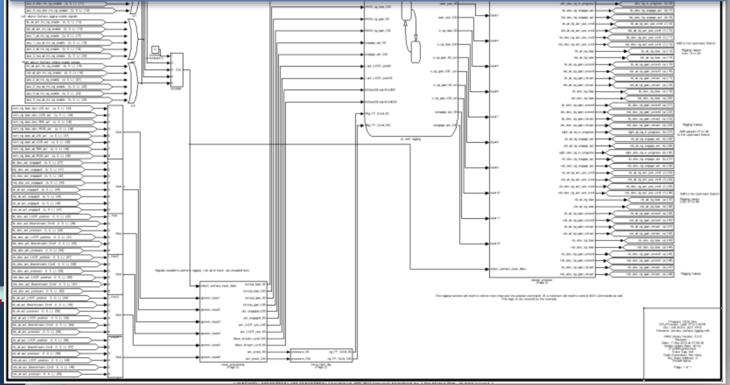
（硬件读写与时序控制，实现逻辑与I/O操作的分离）

MNT分区

实现逻辑处理

D E O S支持

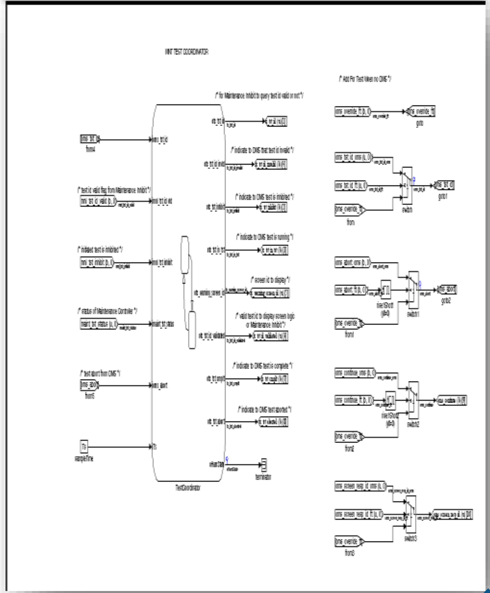
基于状态机的BIT过程

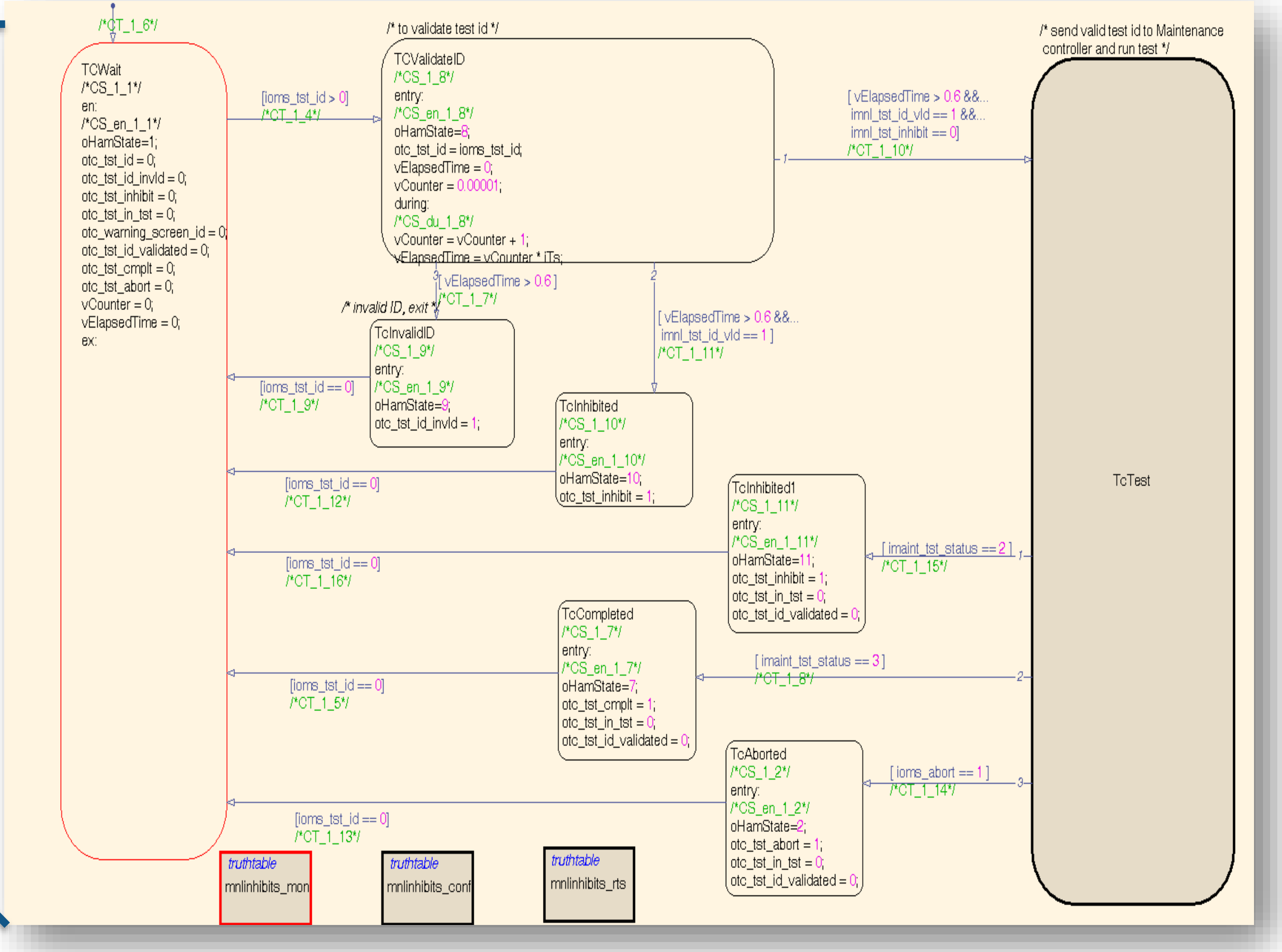


BIT调度逻辑

基于状态机方法对BIT调度逻辑进行设计

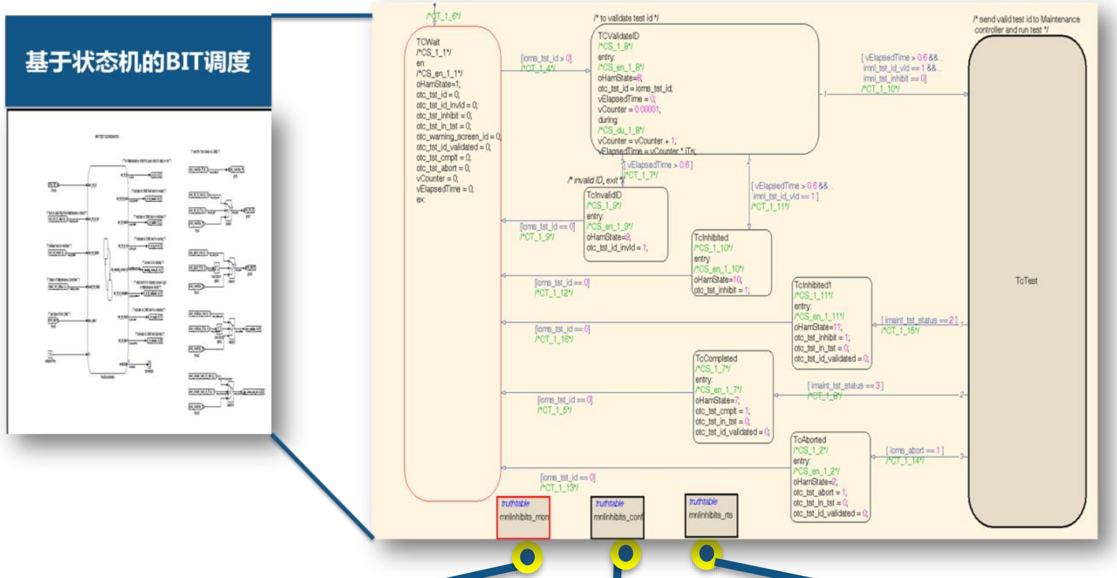
基于状态机的BIT调度



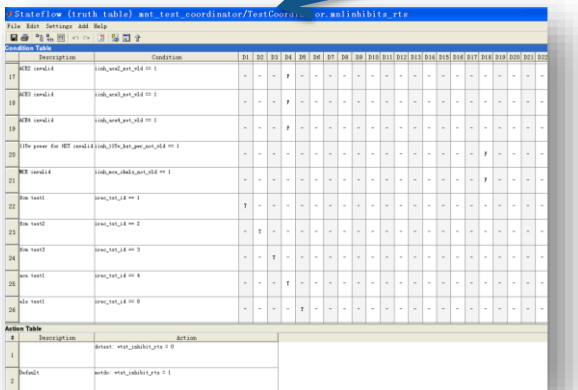


基于真值表的形式化设计方法

基于形式化方法，通过真值表描述复杂的状态跃迁条件



基于状态机的BIT调度



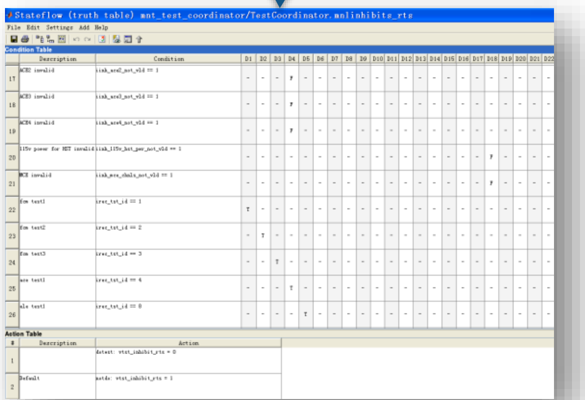
Stateflow (truth table) mnt_test_coordinator/TestCoordinator.mnlinhibits_rts

Condition Table

	Description	Condition	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22
17	ACE2 invalid	iinh_ace2_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
18	ACE3 invalid	iinh_ace3_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
19	ACE4 invalid	iinh_ace4_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
20	115v power for HST invalid	iinh_115v_hst_pwr_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
21	MCE invalid	iinh_mce_chnls_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
22	fcm test1	irec_tst_id == 1	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
23	fcm test2	irec_tst_id == 2	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
24	fcm test3	irec_tst_id == 3	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
25	ace test1	irec_tst_id == 4	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
26	ale test1	irec_tst_id == 8	-	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Action Table

#	Description	Action
1		dotest: vtst_inhibit_rts = 0
2	Default	notdo: vtst_inhibit_rts = 1



Stateflow (truth table) mnt_test_coordinator/TestCoordinator.mnlinhibits_rts

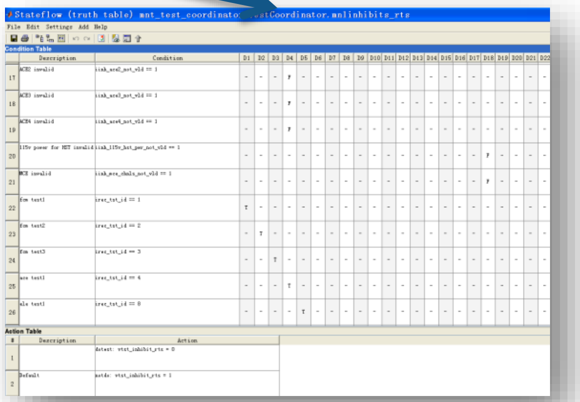
File Edit Settings Add Help

Condition Table

	Description	Condition	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22
17	ACE2 invalid	iinh_ace2_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
18	ACE3 invalid	iinh_ace3_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
19	ACE4 invalid	iinh_ace4_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
20	115v power for HST invalid	iinh_115v_hst_pwr_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
21	MCE invalid	iinh_mce_chnls_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
22	fcm test1	irec_tst_id == 1	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
23	fcm test2	irec_tst_id == 2	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
24	fcm test3	irec_tst_id == 3	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
25	ace test1	irec_tst_id == 4	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
26	ale test1	irec_tst_id == 8	-	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Action Table

#	Description	Action
1		dotest: vtst_inhibit_rts = 0
2	Default	notdo: vtst_inhibit_rts = 1



Stateflow (truth table) mnt_test_coordinator/TestCoordinator.mnlinhibits_rts

File Edit Settings Add Help

Condition Table

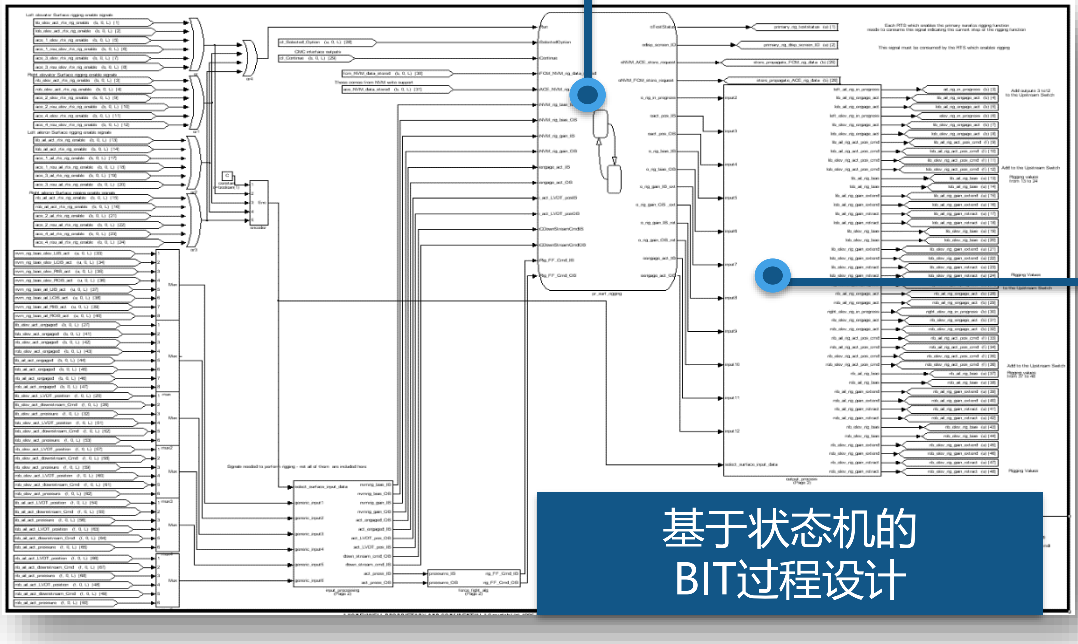
	Description	Condition	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22
17	ACE2 invalid	iinh_ace2_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
18	ACE3 invalid	iinh_ace3_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
19	ACE4 invalid	iinh_ace4_not_vld == 1	-	-	-	F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
20	115v power for HST invalid	iinh_115v_hst_pwr_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
21	MCE invalid	iinh_mce_chnls_not_vld == 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	F	-	-	-	-
22	fcm test1	irec_tst_id == 1	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
23	fcm test2	irec_tst_id == 2	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
24	fcm test3	irec_tst_id == 3	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
25	ace test1	irec_tst_id == 4	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
26	ale test1	irec_tst_id == 8	-	-	-	-	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Action Table

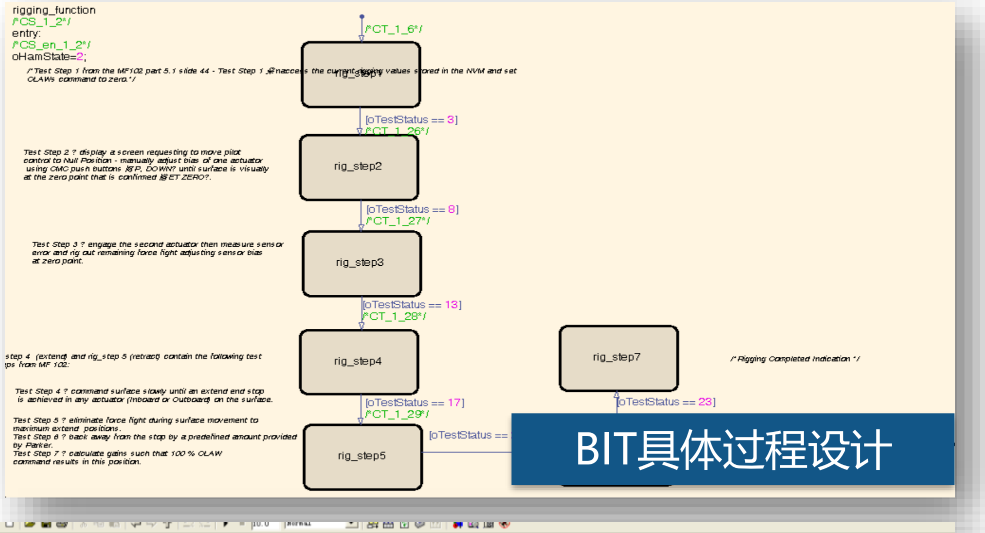
#	Description	Action
1		dotest: vtst_inhibit_rts = 0
2	Default	notdo: vtst_inhibit_rts = 1

基于状态机及Simulink模块的BIT算法设计

BIT步骤基于状态机设计
BIT算法基于Simulink模型设计



基于状态机的BIT过程设计



/*Test Step 1 from the MF102 part 5.1 slide 44 - Test Step 1 ?access the current rigging values stored in the NVM and set OLAWs command to zero.*/

Test Step 2 ? display a screen requesting to move pilot control to Null Position - manually adjust bias of one actuator using CMC push buttons ?UP, DOWN? until surface is visually at the zero point that is confirmed ?SET ZERO?.

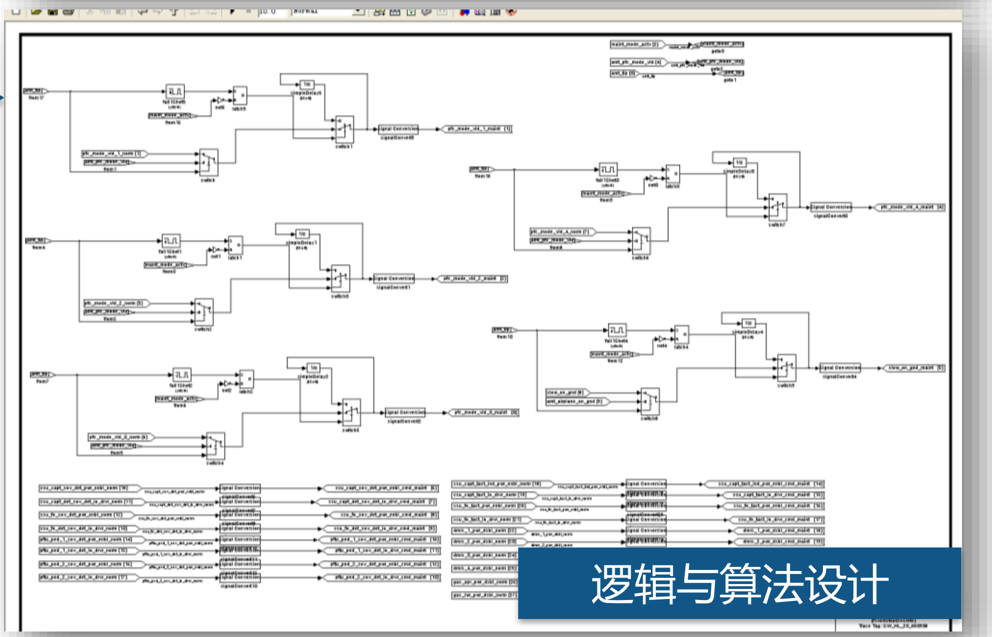
Test Step 3 ? engage the second actuator then measure sensor error and rig out remaining force fight adjusting sensor bias at zero point.

step 4 (extend) and rig_step 5 (retract) contain the following test ups from MF 102:

Test Step 4 ? command surface slowly until an extend end stop is achieved in any actuator (Inboard or Outboard) on the surface.

Test Step 5 ? eliminate force fight during surface movement to maximum extend positions.
Test Step 6 ? back away from the stop by a predefined amount provided by Parker.
Test Step 7 ? calculate gains such that 100 % OLAW command results in this position.

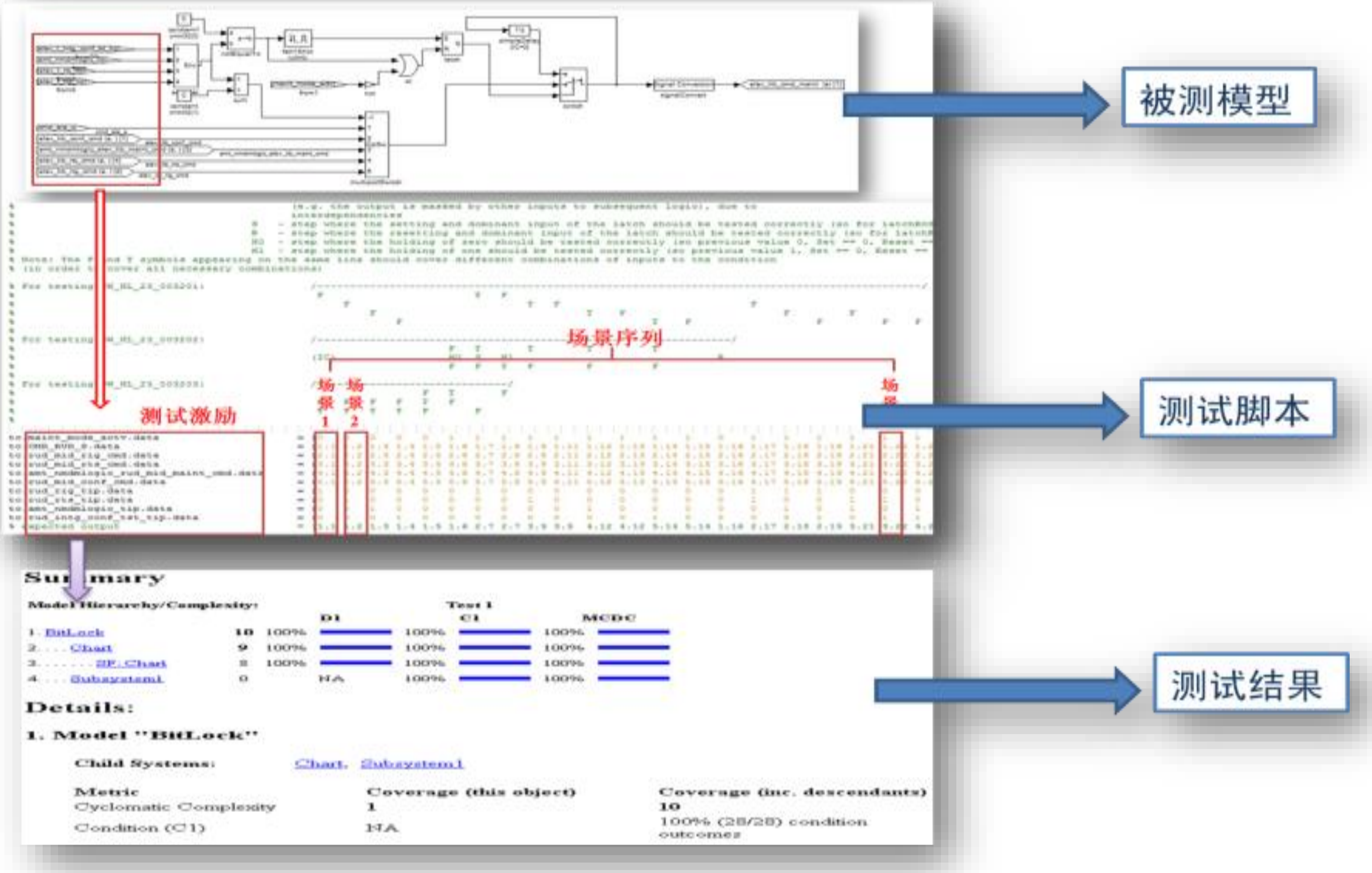
BIT具体过程设计



逻辑与算法设计

Simulink模型测试

基于脚本和场景序列的完成Simulink 模型的测试



```
%                                   (e.g. the output is masked by other inputs to subsequent logic), due to
%                                   interdependencies
%                        S   - step where the setting and dominant input of the latch should be tested correctly (ex for latch
%                        R   - step where the resetting and dominant input of the latch should be tested correctly (ex for latch
%                        H0  - step where the holding of zero should be tested correctly (ex previous value 0, Set == 0, Reset ==
%                        H1  - step where the holding of one should be tested correctly (ex previous value 1, Set == 0, Reset ==
% Note: The F and T symbols appearing on the same line should cover different combinations of inputs to the condition
% (in order to cover all necessary combinations)

% For testing M_HL_23_003201:

% For testing M_HL_23_003202:

% For testing M_HL_23_003203:

to maint_mode_actv.data
to OHB_BUS_5.data
to rud_mid_rig_cmd.data
to rud_mid_sta_cmd.data
to amt_nmdmlogic_rud_mid_maint_cmd.data
to rud_mid_conf_cmd.data
to rud_rig_tip.data
to rud_sta_tip.data
to amt_nmdmlogic_tip.data
to rud_intg_conf_tst_tip.data
% expected output
```

Summary

Model Hierarchy/Complexity:

		Test 1					
		D1		C1		MCDC	
1. BitLock	10	100%	▬	100%	▬	100%	▬
2. . . . Chart	9	100%	▬	100%	▬	100%	▬
3. . . . . . . SF: Chart	8	100%	▬	100%	▬	100%	▬
4. . . . Subsystem1	0	NA		100%	▬	100%	▬

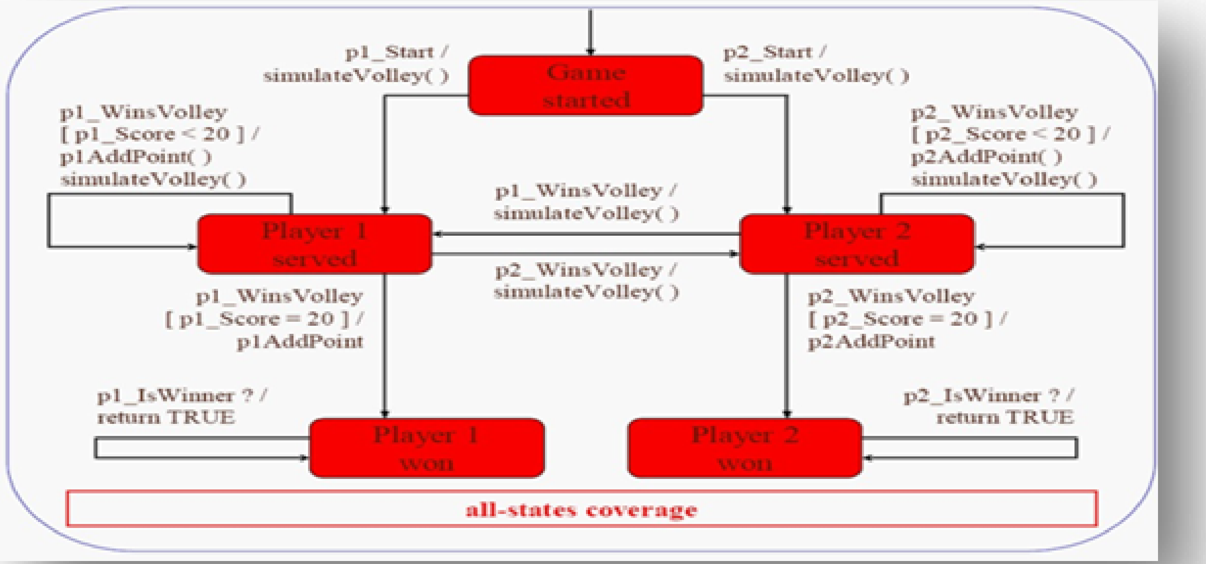
Details:

1. Model "BitLock"

Child Systems: Chart, Subsystem1

Metric	Coverage (this object)	Coverage (inc. descendants)
Cyclomatic Complexity	1	10
Condition (C1)	NA	100% (28/28) condition outcomes

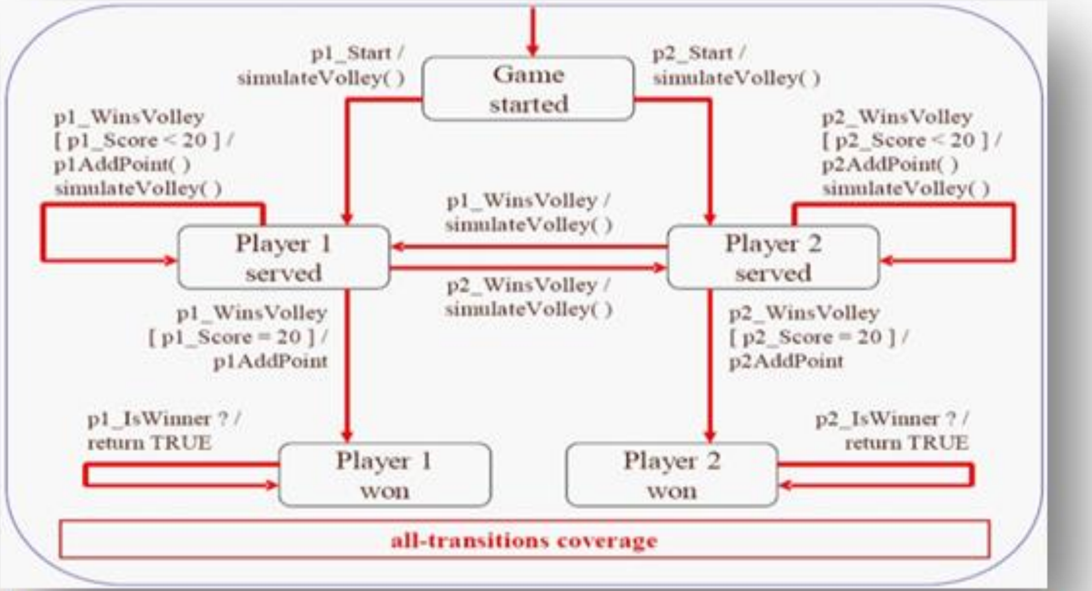
状态机模型测试



Game started
p1_Start / simulateVolley( )
p2_Start / simulateVolley( )
p1_WinsVolley [p1_Score < 20] / p1AddPoint( ) simulateVolley( )
p2_WinsVolley [p2_Score < 20] / p2AddPoint( ) simulateVolley( )
p1_WinsVolley / simulateVolley( )
Player 1 served
Player 2 served
p2_WinsVolley / simulateVolley( )
p1_WinsVolley [p1_Score = 20] / p1AddPoint
p2_WinsVolley [p2_Score = 20] / p2AddPoint
p1_IsWinner ? / return TRUE
p2_IsWinner ? / return TRUE
Player 1 won
Player 2 won
all-states coverage

解决方案第一步：

每一个状态至少被一个测试用例覆盖一次

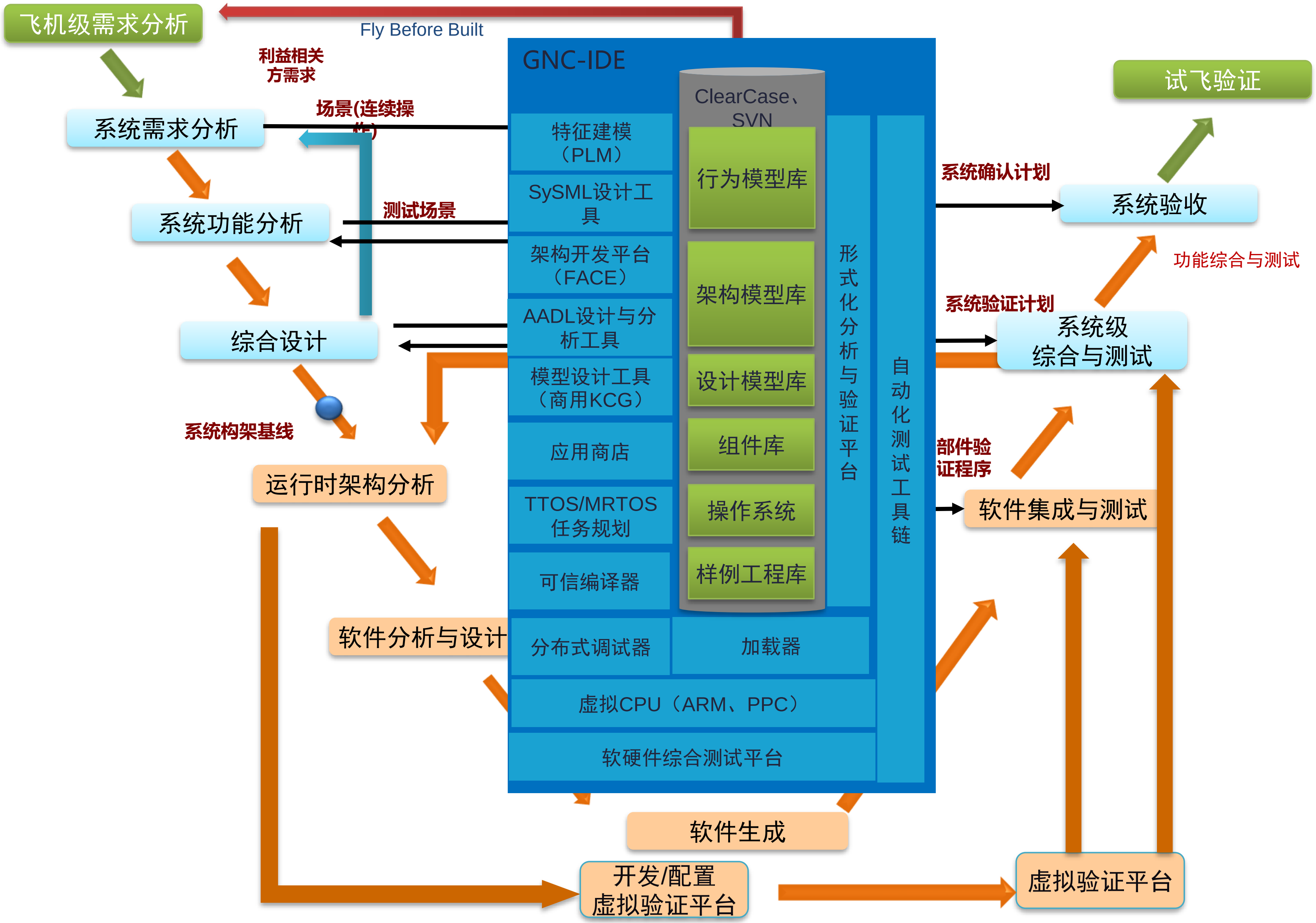


p1_Start / simulateVolley( )
Game started
p2_Start / simulateVolley( )
p1_WinsVolley [p1_Score < 20] / p1AddPoint( ) simulateVolley( )
p2_WinsVolley [p2_Score < 20] / p2AddPoint( ) simulateVolley( )
p1_WinsVolley / simulateVolley( )
Player 1 served
Player 2 served
p2_WinsVolley / simulateVolley( )
p1_WinsVolley [p1_Score = 20] / p1AddPoint
p2_WinsVolley [p2_Score = 20] / p2AddPoint
p1_IsWinner ? / return TRUE
p2_IsWinner ? / return TRUE
Player 1 won
Player 2 won
all-transitions coverage

解决方案第二步：

每一个测试转移条件至少被一个测试用例覆盖一次

打造支撑机载嵌入式实时系统的MBSE生态环境



谢 谢