MATLAB EXPO

모델 기반 설계를 활용한 Legacy C,C++ 코드의 통합과 검증

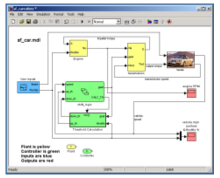
김학범 차장, 매스웍스코리아

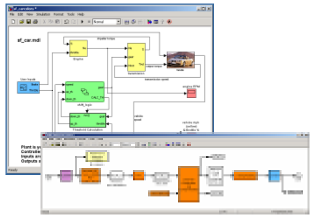
MathWorks®

Model-Based Design

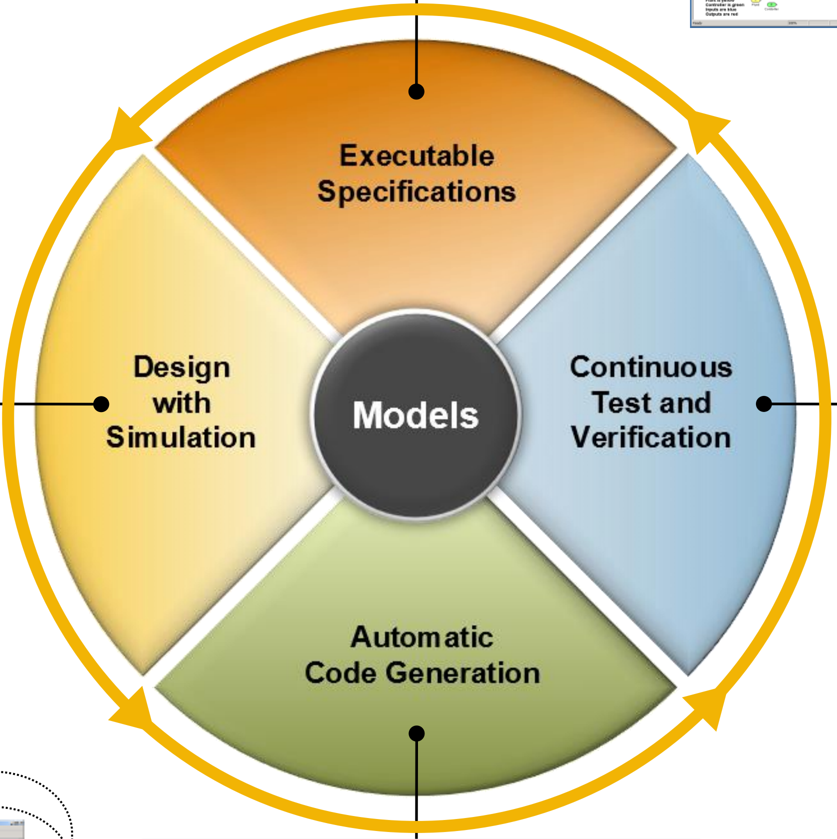
Development Process

Executable models
-Unambiguous
-Only "one truth"

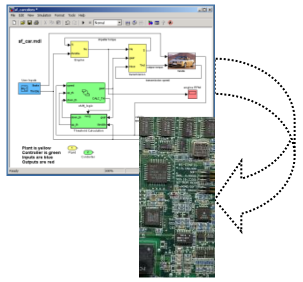


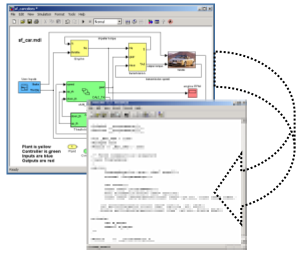


Simulation
-Reduces physical prototypes
-Systematic "what-if" analysis


Executable Specifications
Design with Simulation
Models
Continuous Test and Verification
Automatic Code Generation

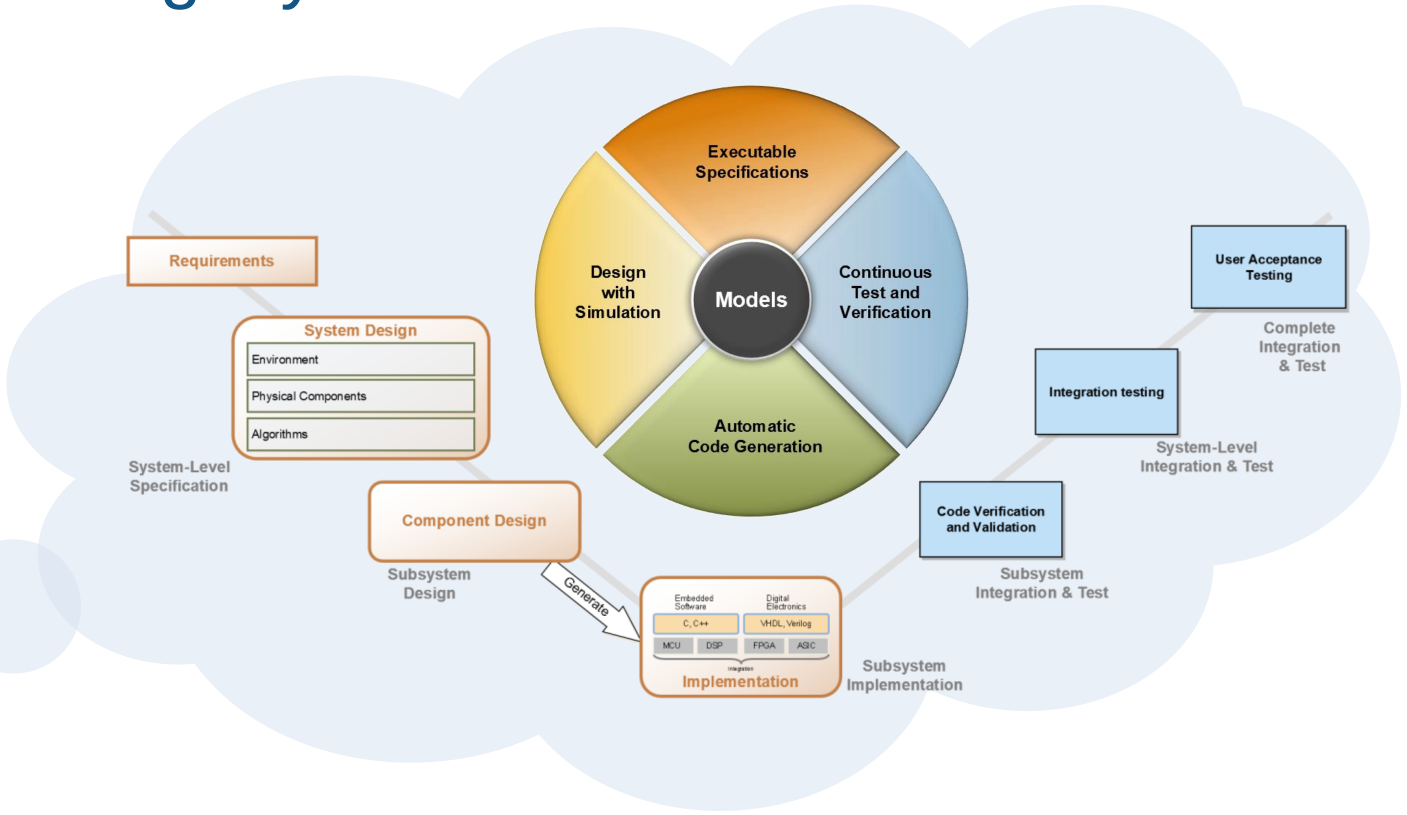
Analyze and Test the Design
-Create reusable tests
-Detects errors earlier
-Formal methods





Automatic code generation
-Minimizes coding errors

How to get started MBD with Legacy Code?



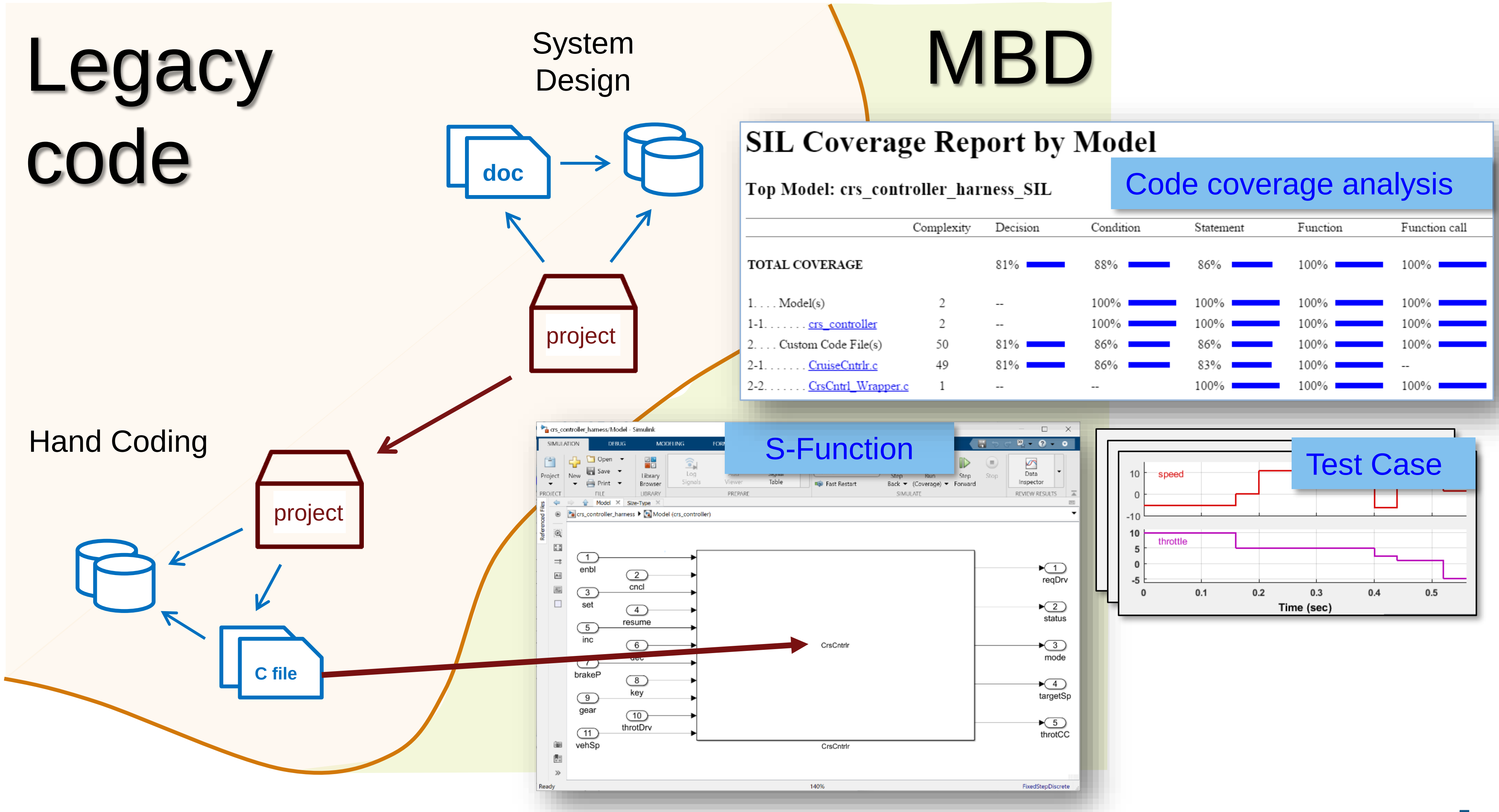
Agenda

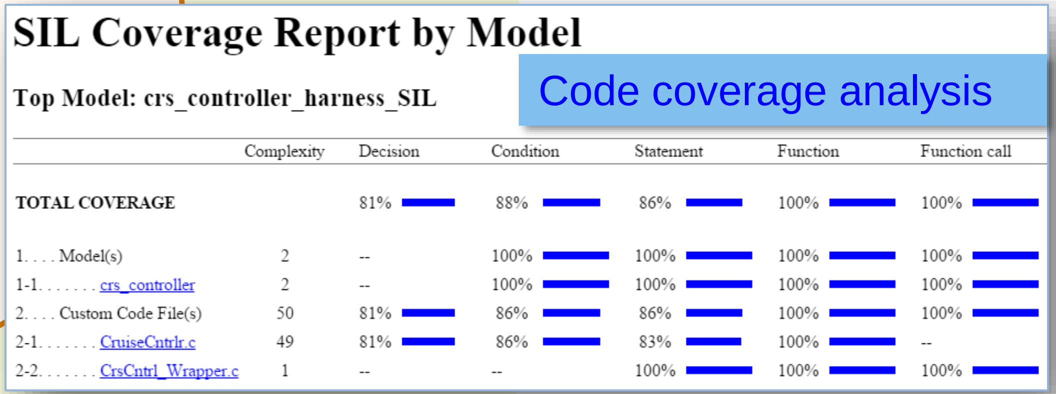
- How to get started MBD with Legacy Code?
- Legacy Code Integration using Simulink
- Legacy Code Verification
- Key Takeaways

Agenda

- **How to get started MBD with Legacy Code?**
- Legacy Code Integration using Simulink
- Legacy Code Verification
- Key Takeaways

Verify Legacy Code using Simulink



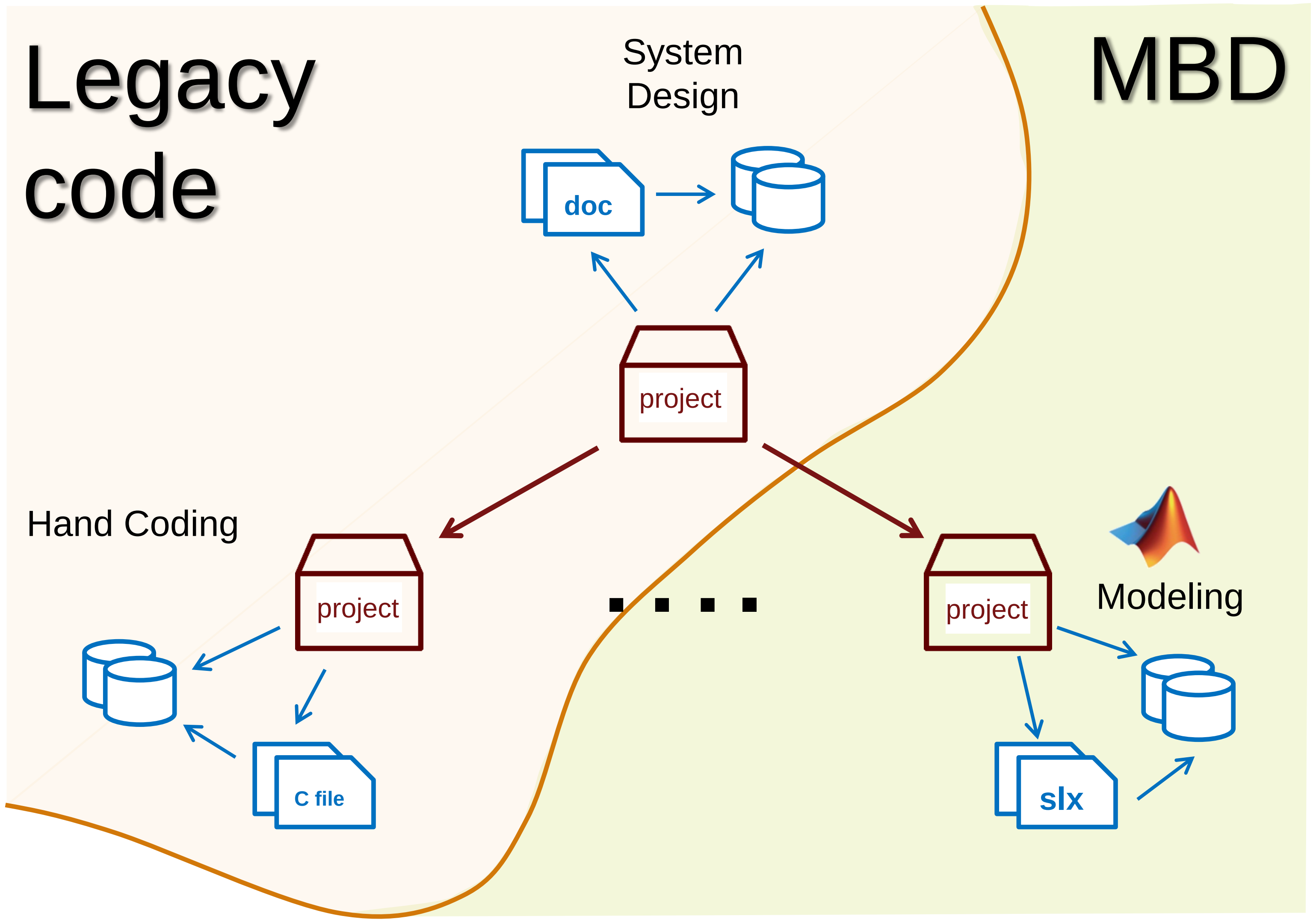


SIL Coverage Report by Model

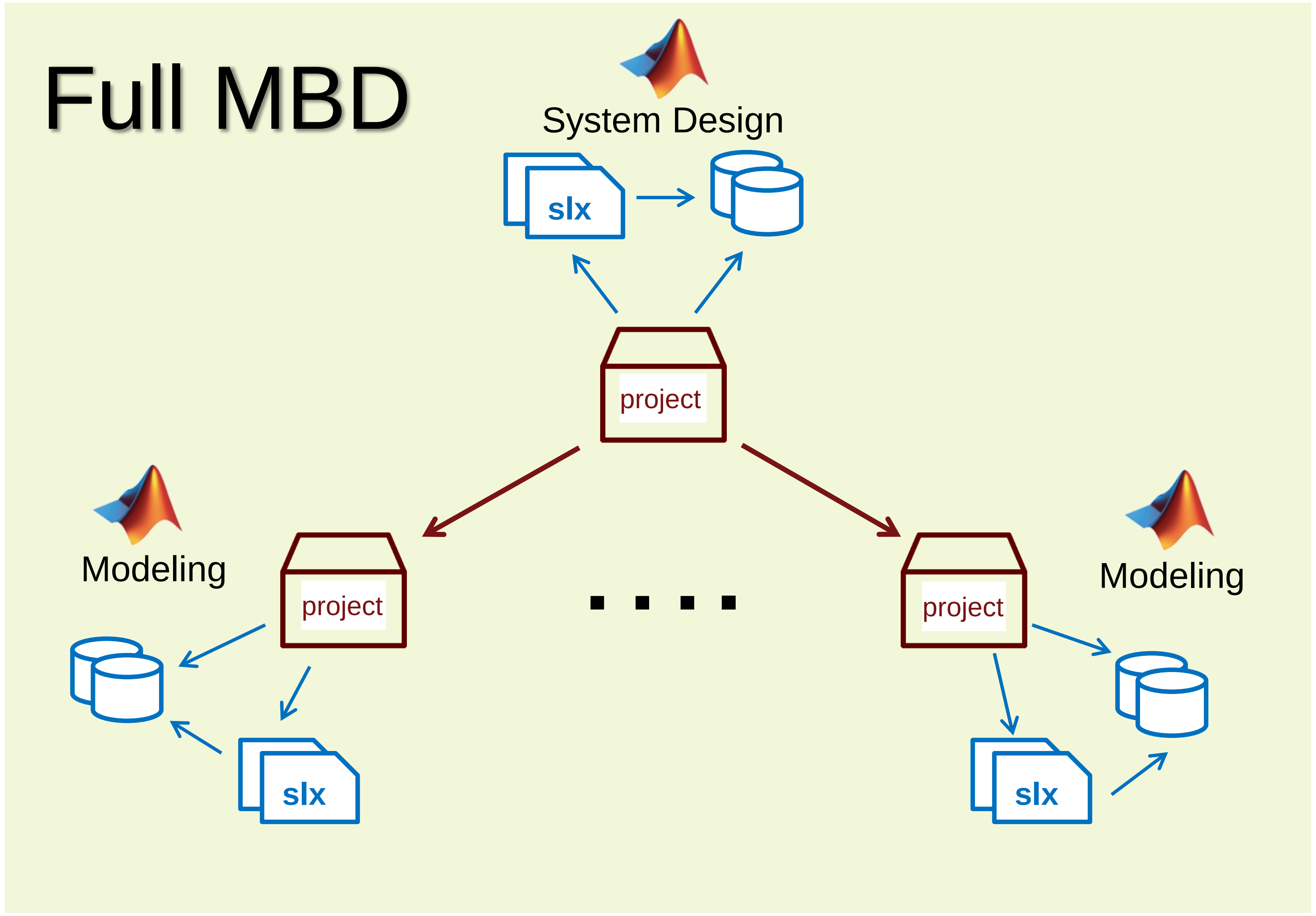
Top Model: crs_controller_harness_SIL

	Complexity	Decision	Condition	Statement	Function	Function call
TOTAL COVERAGE		81%	88%	86%	100%	100%
1. . . . Model(s)	2	--	100%	100%	100%	100%
1-1. . . . . . . crs_controller	2	--	100%	100%	100%	100%
2. . . . Custom Code File(s)	50	81%	86%	86%	100%	100%
2-1. . . . . . . CruiseCntrlr.c	49	81%	86%	83%	100%	--
2-2. . . . . . . CrsCntrl_Wrapper.c	1	--	--	100%	100%	100%

Experiment with a Small Piece of the Project



Adopt Full MBD to Project



Model-Based Design With Legacy C/C++ Code?

Legacy Code → MBD with Legacy Code → Full MBD

Legacy Code Verification using Simulink

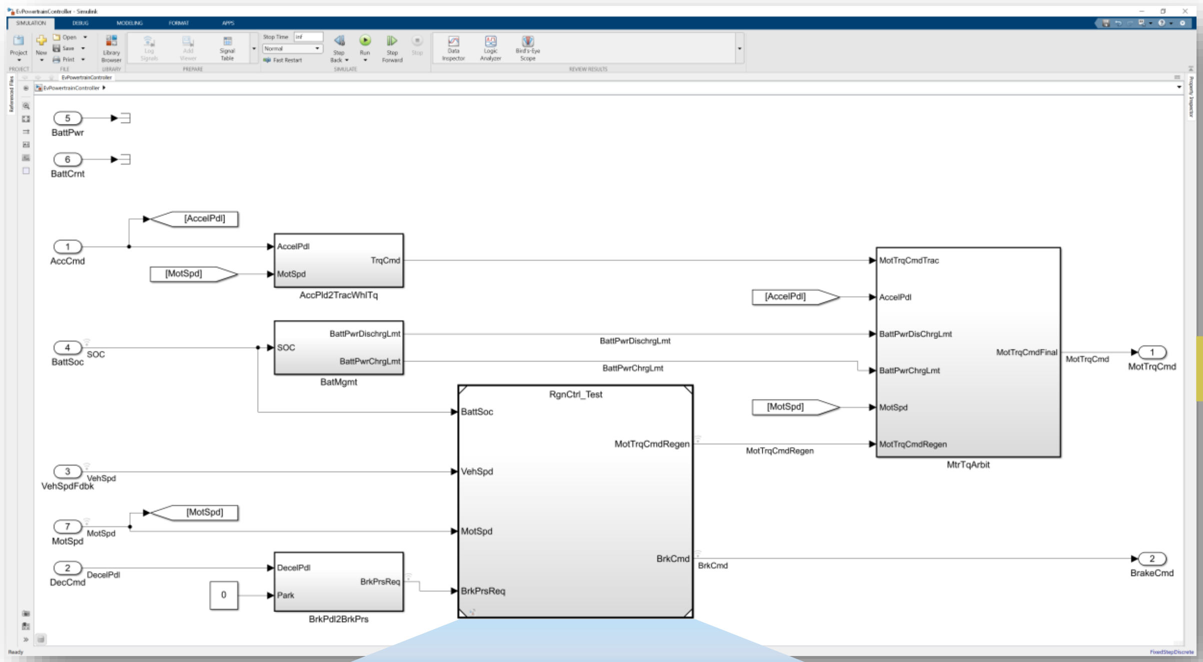
MBD with C/C++ code

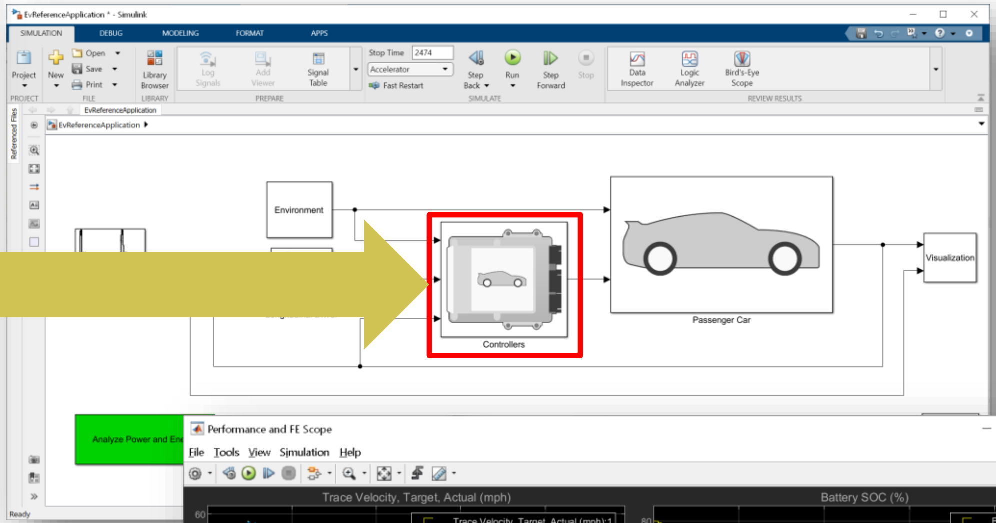
Agenda

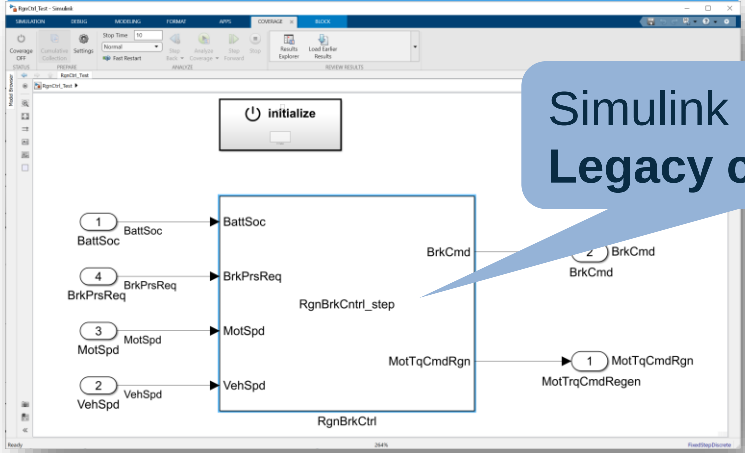
- How to get started MBD with Legacy Code?
- **Legacy Code Integration using Simulink**
- Legacy Code Verification
- Key Takeaways

Model-Based Design with Legacy Code

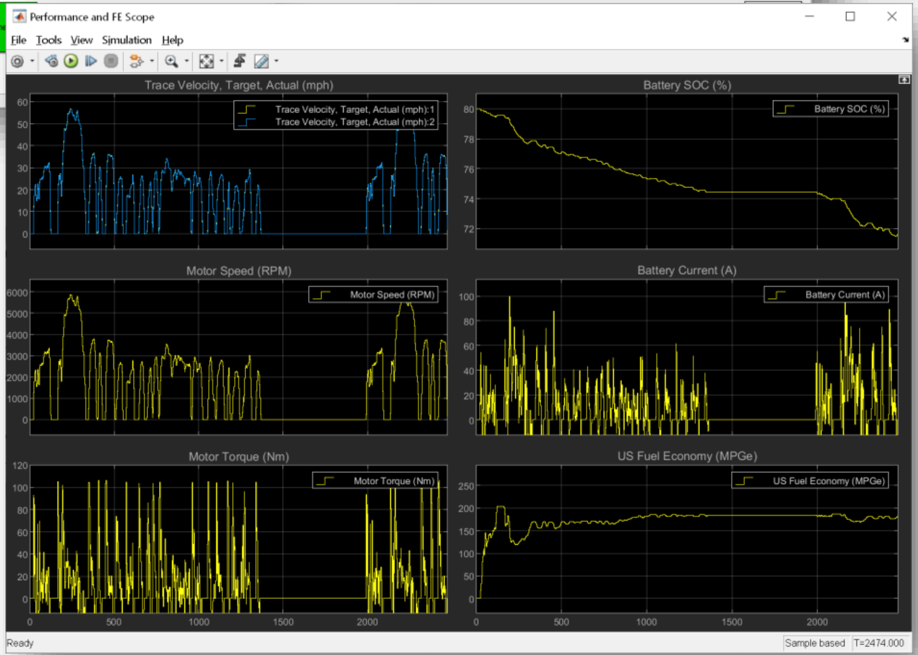
Example: EV Vehicle Simulation for VCU



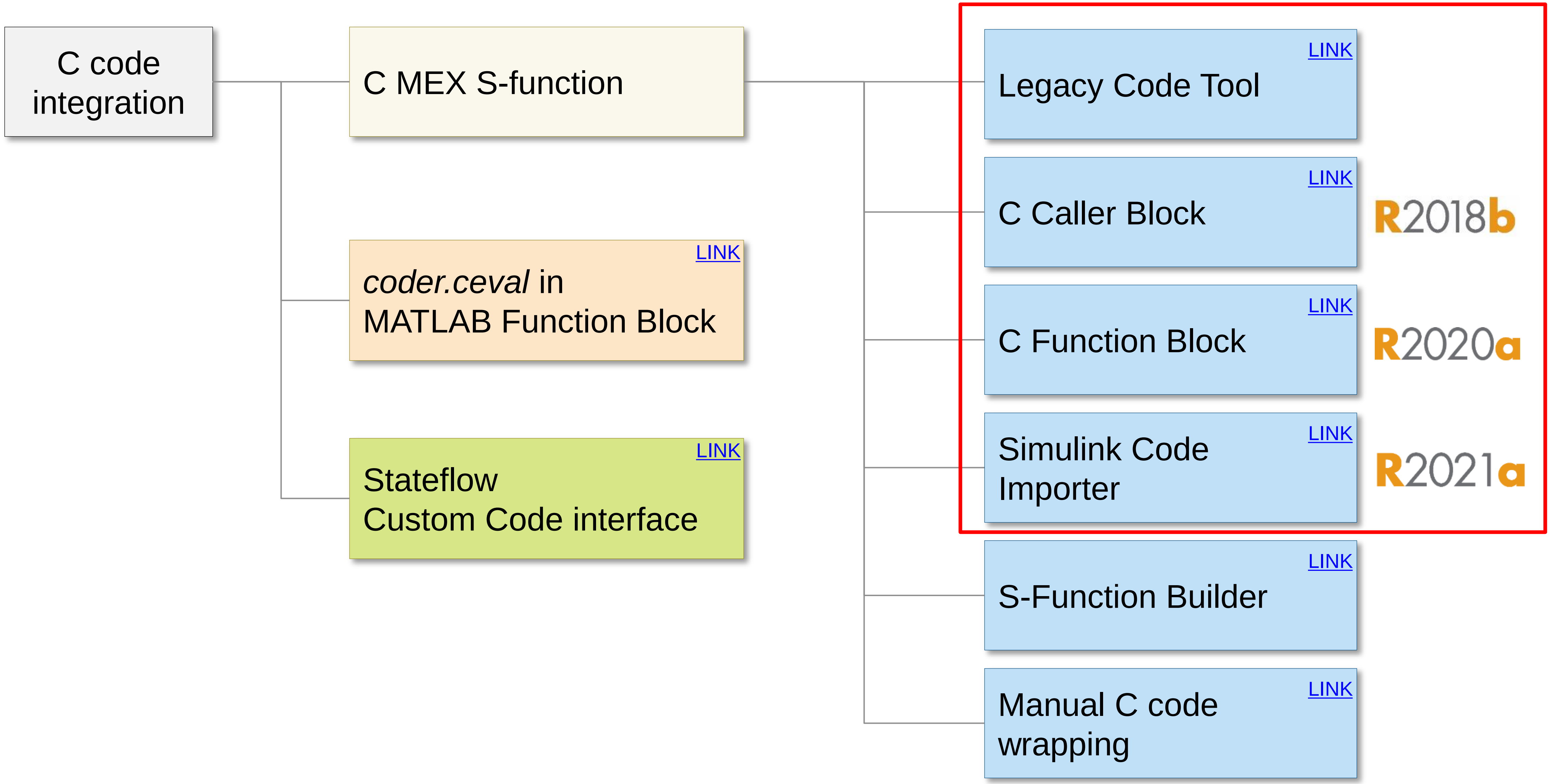




Simulink block calling **Legacy codes**

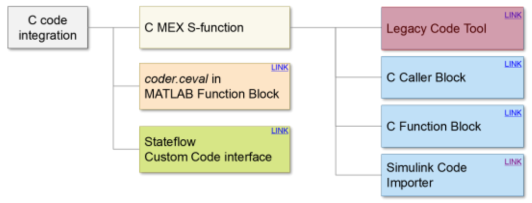


Legacy Code Integration Methods



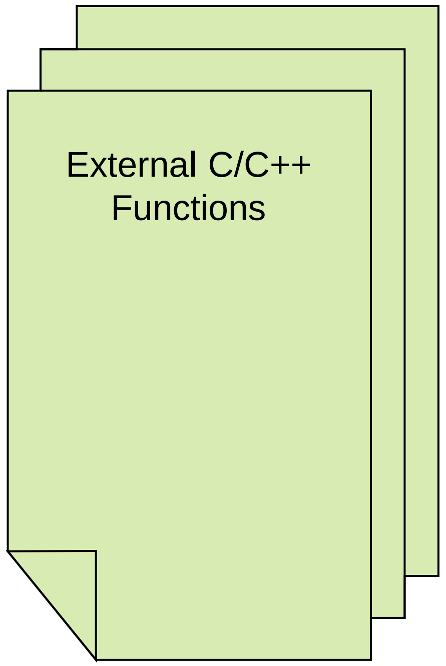
Legacy Code Integration Methods

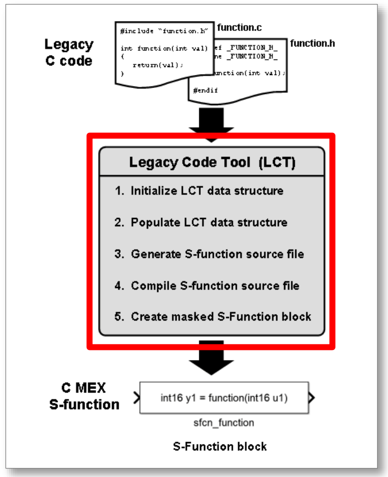
Type	Features	Manual Building Process	Block / UI Icon
Legacy Code Tool	▪ Full flexibility, Generate and build S-function with easy-to-use MATLAB API ▪ Creation of device driver blocks for HW input and output	Need to build when changing codes	system S-Function
C Caller Block	▪ Easy to call a function in legacy code ▪ Calls a single function in one block ▪ Good for Unit test of C code	No manual build process	<FunctionName> C Caller
C Function Block	▪ Advantage of C Caller Block + Easy to add C code in a Simulink block ▪ Call multiple functions in one block ▪ Unit and integration test of C code		C C Function
Simulink Code Importer	▪ Easy to access (UI in Toolstrip) ▪ Create a block library for C function ▪ Unit and integration test of C code with Simulink Test		Code Importer



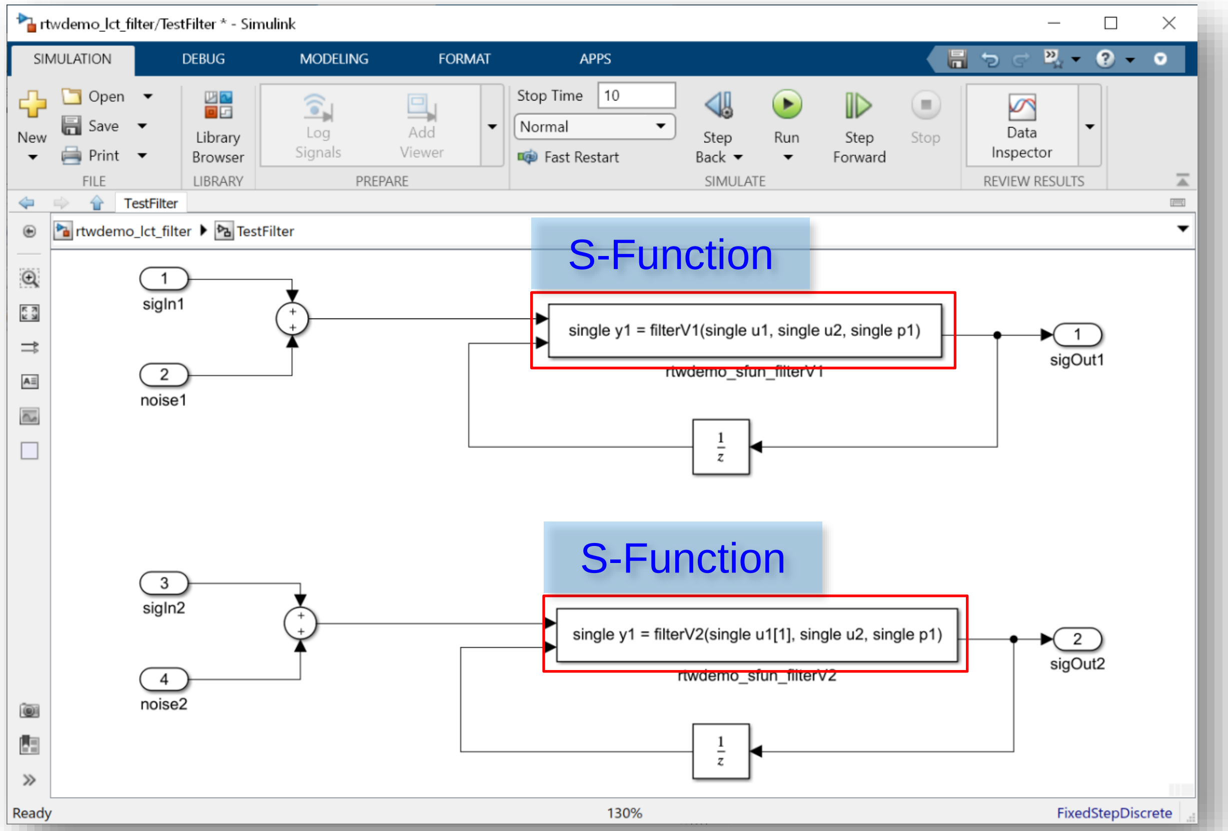
Legacy Code Tool

- Integrate existing C/C++ functions, such as device drivers, lookup tables, and general functions and interfaces, into Simulink

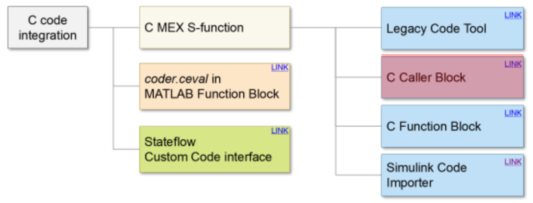



Legacy C code
function.c
function.h
Legacy Code Tool (LCT)
1. Initialize LCT data structure
2. Populate LCT data structure
3. Generate S-function source file
4. Compile S-function source file
5. Create masked S-Function block
C MEX S-function
int16 y1 = function(int16 u1)
sfcn_function
S-Function block

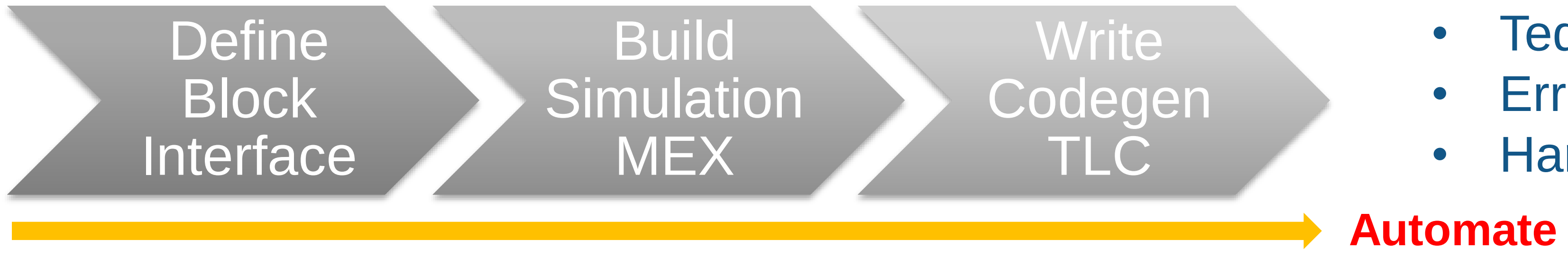
Generating S-function



C Caller Block



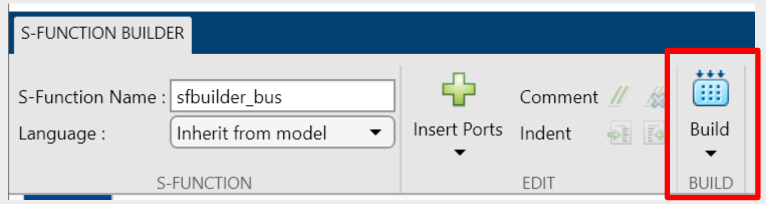
- Key feature
  - Automate the process



- Tedious
- Error prone
- Hard to maintain

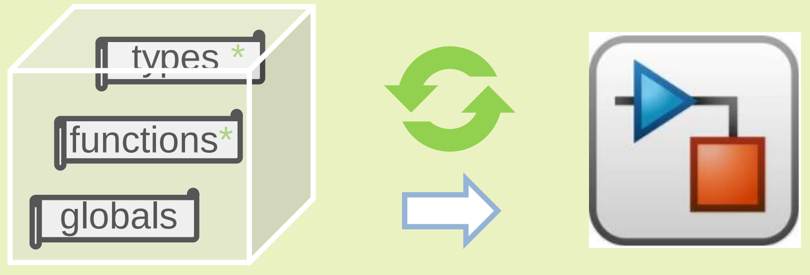
  - Synchronize with custom code changes with C Caller

Legacy Code Tool
→ Must build after modifying codes

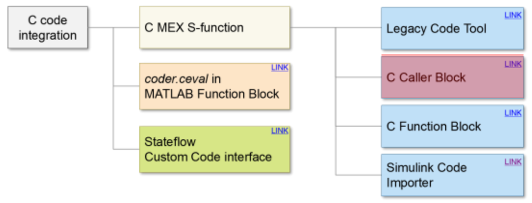


```
legacy_code('compile', def);
```

C Caller: Automatically synchronized



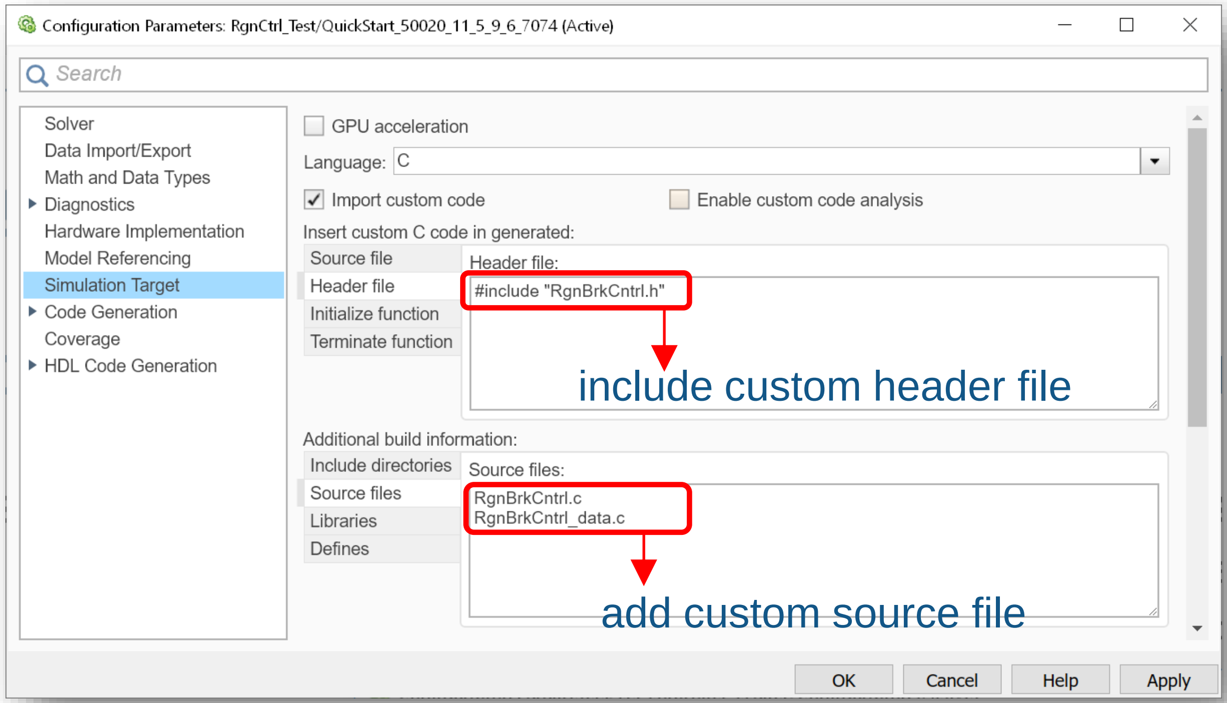
C/C++ Code



C Caller Block

Specify Custom Code in the Configuration Parameters

- Custom code is specified on the Configuration Parameters
  - **Header file section:** Any code that needs to be inserted into the header file
  - **Source files section:** List of source files that needs to be compiled



Configuration Parameters: RgnCtrl_Test/QuickStart_50020_11_5_9_6_7074 (Active)

Search

☐ GPU acceleration

Language: C

☑ Import custom code ☐ Enable custom code analysis

Insert custom C code in generated:

Header file:

```
#include "RgnBrkCntrl.h"
```

include custom header file

Additional build information:

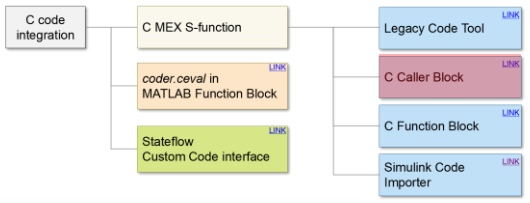
Source files:

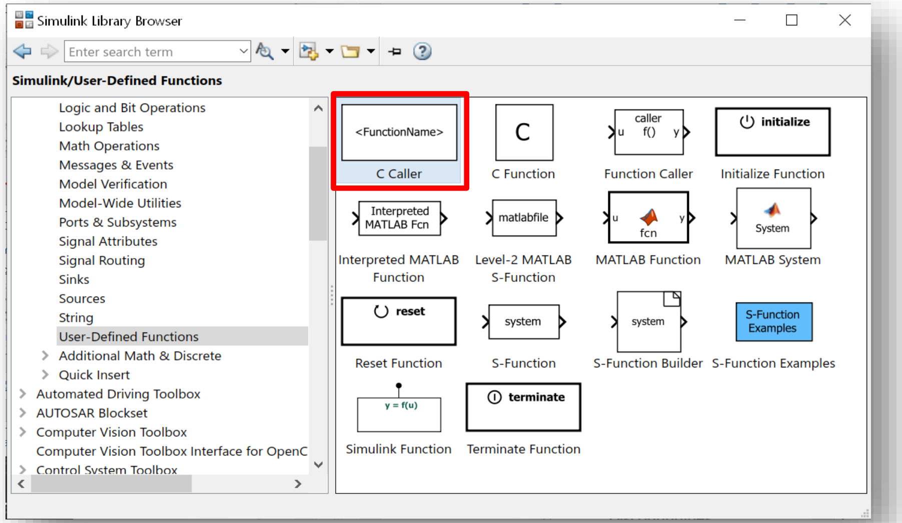
```
RgnBrkCntrl.c
RgnBrkCntrl_data.c
```

add custom source file

C Caller Block

Select the function that you want to call



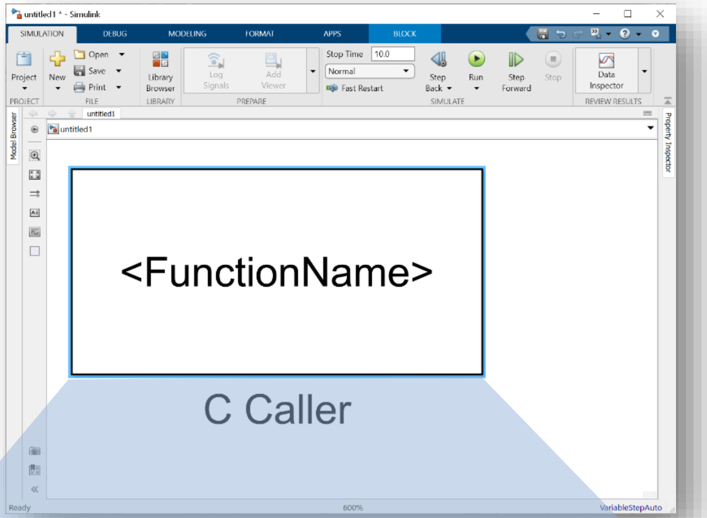


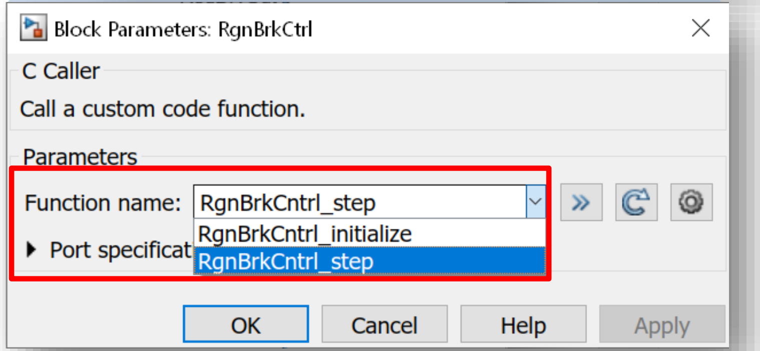
Simulink Library Browser

Simulink/User-Defined Functions

- Logic and Bit Operations
- Lookup Tables
- Math Operations
- Messages & Events
- Model Verification
- Model-Wide Utilities
- Ports & Subsystems
- Signal Attributes
- Signal Routing
- Sinks
- Sources
- String
- User-Defined Functions
- Additional Math & Discrete
- Quick Insert
- Automated Driving Toolbox
- AUTOSAR Blockset
- Computer Vision Toolbox
- Computer Vision Toolbox Interface for OpenC
- Control System Toolbox

C Caller, C Function, Function Caller, Initialize Function, Interpreted MATLAB Function, Level-2 MATLAB S-Function, MATLAB Function, MATLAB System, Reset Function, S-Function, S-Function Builder, S-Function Examples, Simulink Function, Terminate Function





Block Parameters: RgnBrkCtrl

C Caller

Call a custom code function.

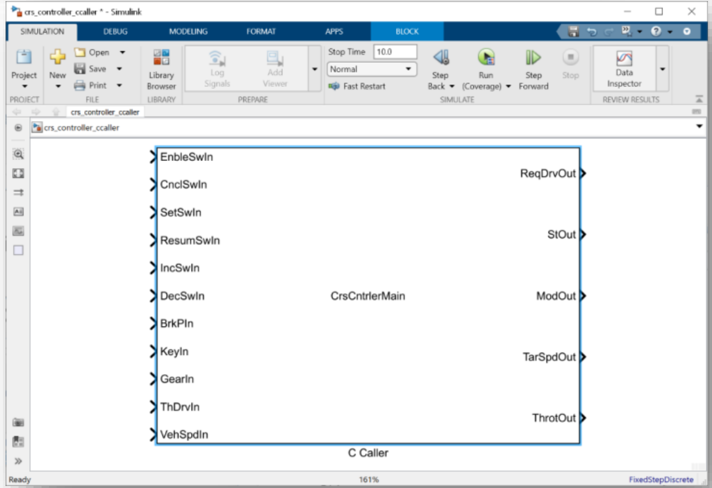
Parameters

Function name: RgnBrkCntrl_step

- RgnBrkCntrl_initialize
- RgnBrkCntrl_step

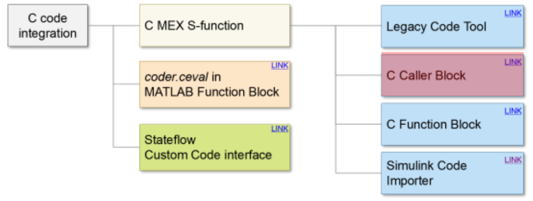
Port specificat

OK | Cancel | Help | Apply

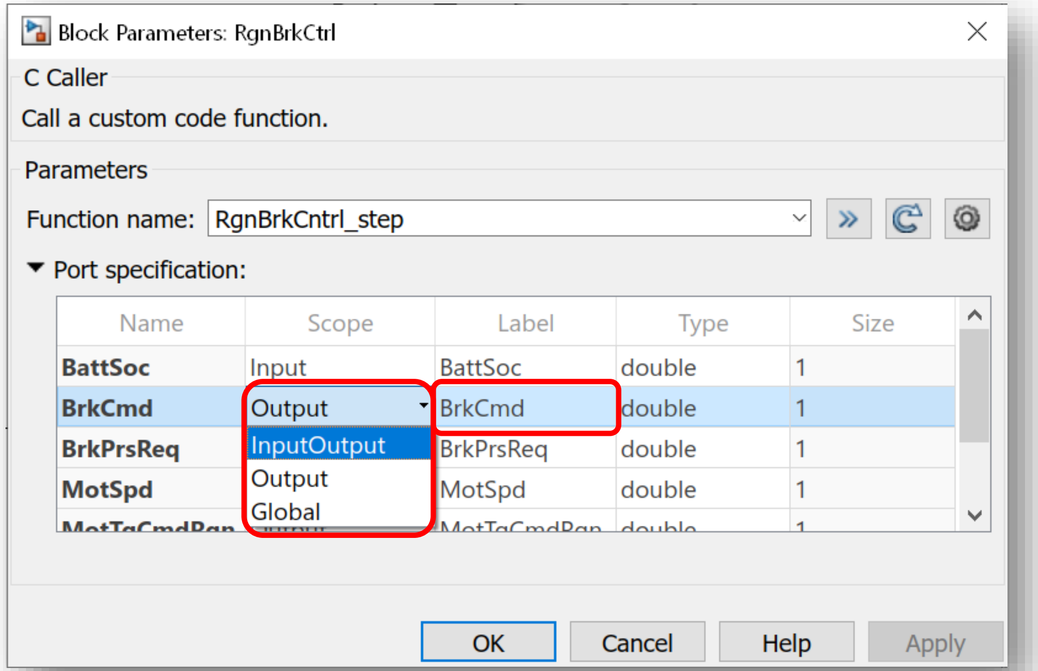


C Caller Block

Customize the function that you want to call



- Mapping inputs, outputs or parameters to C Caller Block



Block Parameters: RgnBrkCtrl

C Caller

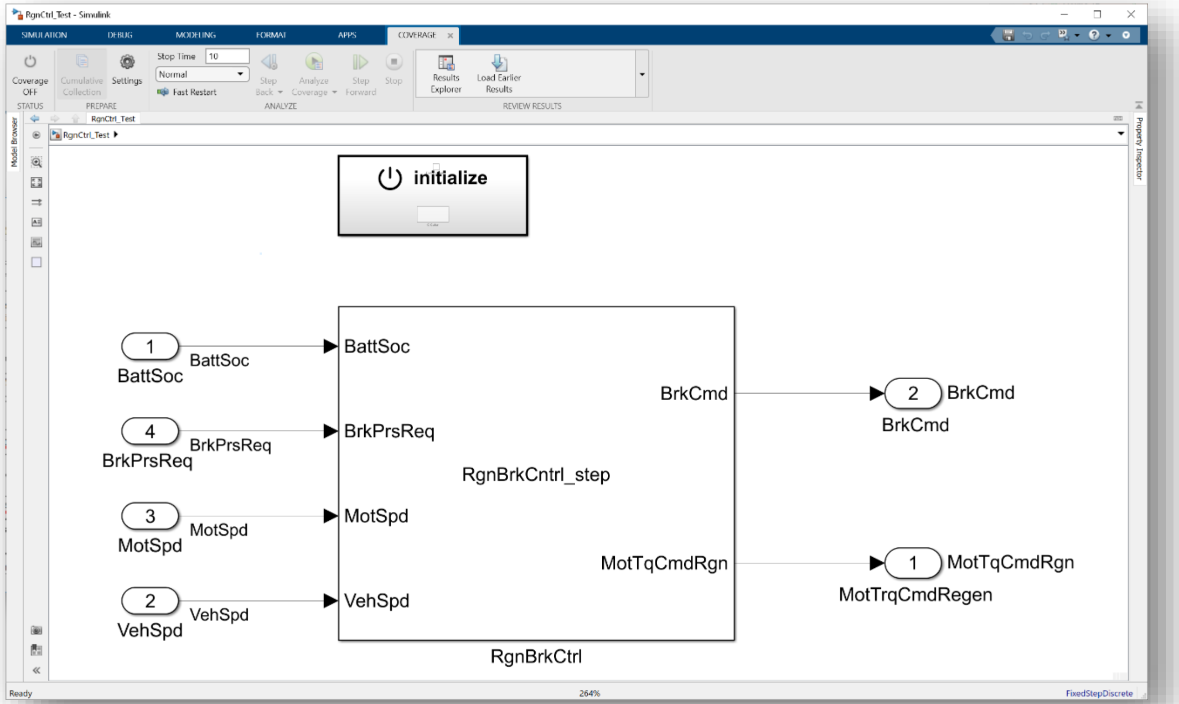
Call a custom code function.

Parameters

Function name: RgnBrkCntrl_step

Port specification:

Name	Scope	Label	Type	Size
BattSoc	Input	BattSoc	double	1
BrkCmd	Output	BrkCmd	double	1
BrkPrsReq		BrkPrsReq	double	1
MotSpd		MotSpd	double	1
MotTqCmdRgn		MotTqCmdRgn	double	1

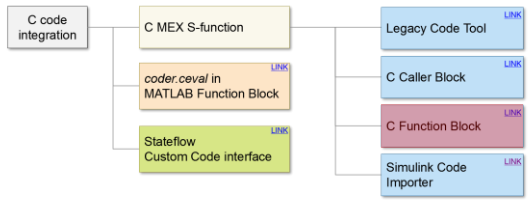


1) Change argument scope to “Output”

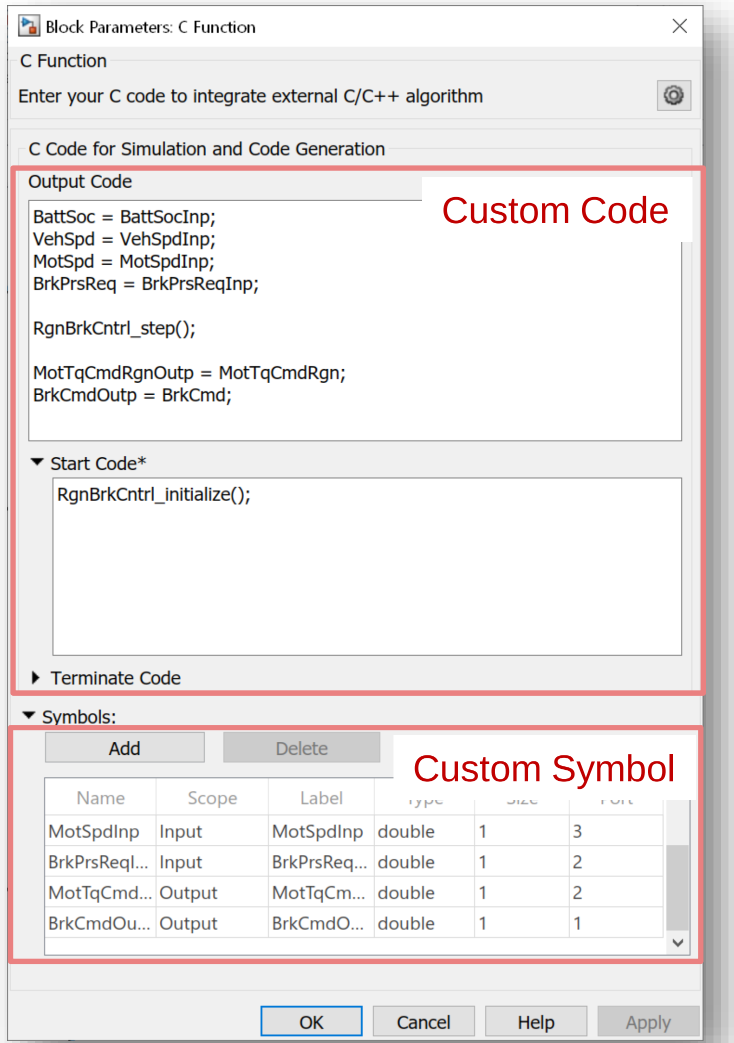
2) (Optional) Override with a better port name

3) Complete the test model with connecting signal ports

C Function Block



- Motivation
  - Make it really easy to add custom code to Simulink
  - Make the simpler uses of S Function Builder easy
  - Replace Legacy Code Block
- Behavior
  - Code is parsed and managed in Simulink
  - Better customizing and diagnostics
  - Supports Start/Terminate



Block Parameters: C Function

C Function

Enter your C code to integrate external C/C++ algorithm

C Code for Simulation and Code Generation

Output Code

```
BattSoc = BattSocInp;
VehSpd = VehSpdInp;
MotSpd = MotSpdInp;
BrkPrsReq = BrkPrsReqInp;

RgnBrkCntrl_step();

MotTqCmdRgnOutp = MotTqCmdRgn;
BrkCmdOutp = BrkCmd;
```

Custom Code

Start Code*

```
RgnBrkCntrl_initialize();
```

Terminate Code

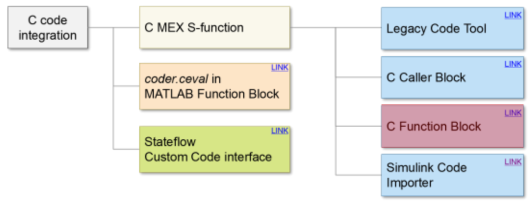
Symbols:

Add Delete

Custom Symbol

Name	Scope	Label	Type	Size	Port
MotSpdInp	Input	MotSpdInp	double	1	3
BrkPrsReqI...	Input	BrkPrsReq...	double	1	2
MotTqCmd...	Output	MotTqCm...	double	1	2
BrkCmdOu...	Output	BrkCmdO...	double	1	1

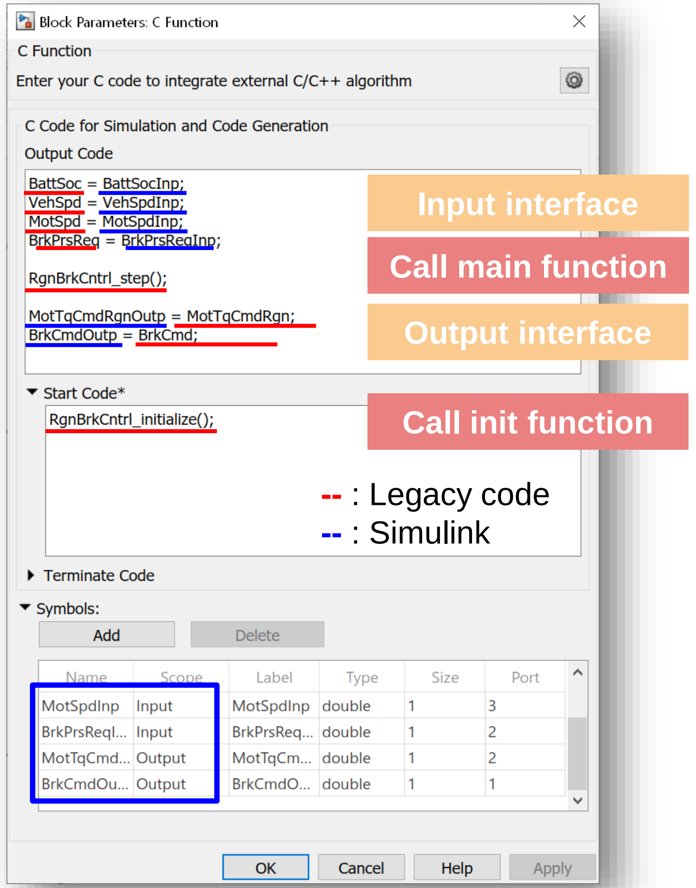
OK Cancel Help Apply



C Function Block

Customize the code and variables that you want to call

- Customize code in Output Code, Start Code, Terminate Code
- Mapping inputs, outputs or parameters to C Function Block



Block Parameters: C Function

C Function

Enter your C code to integrate external C/C++ algorithm

C Code for Simulation and Code Generation

Output Code

```
BattSoc = BattSocInp;
VehSpd = VehSpdInp;
MotSpd = MotSpdInp;
BrkPrsReq = BrkPrsReqInp;

RgnBrkCntrl_step();

MotTqCmdRgnOutp = MotTqCmdRgn;
BrkCmdOutp = BrkCmd;
```

Input interface

Call main function

Output interface

Start Code*

```
RgnBrkCntrl_initialize();
```

Call init function

-- : Legacy code
-- : Simulink

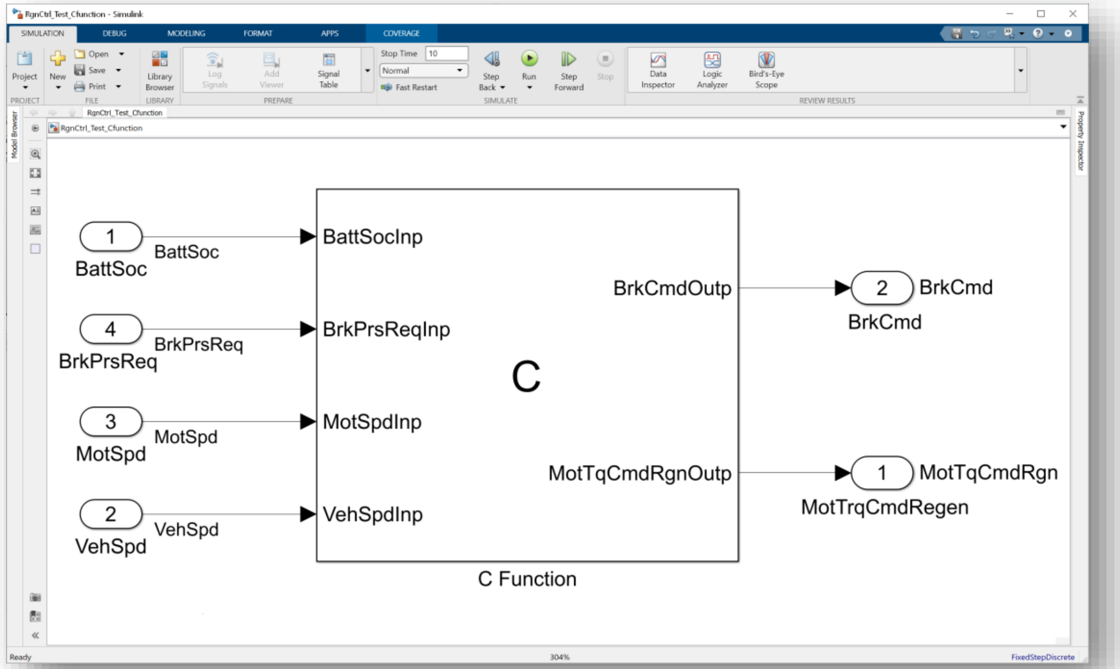
Terminate Code

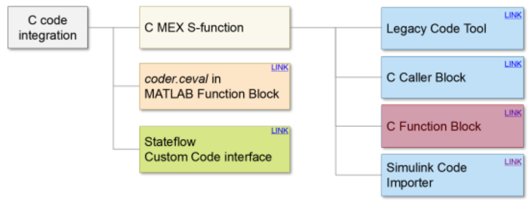
Symbols:

Add Delete

Name	Scope	Label	Type	Size	Port
MotSpdInp	Input	MotSpdInp	double	1	3
BrkPrsReqI...	Input	BrkPrsReq...	double	1	2
MotTqCmd...	Output	MotTqCm...	double	1	2
BrkCmdOu...	Output	BrkCmdO...	double	1	1

OK Cancel Help Apply

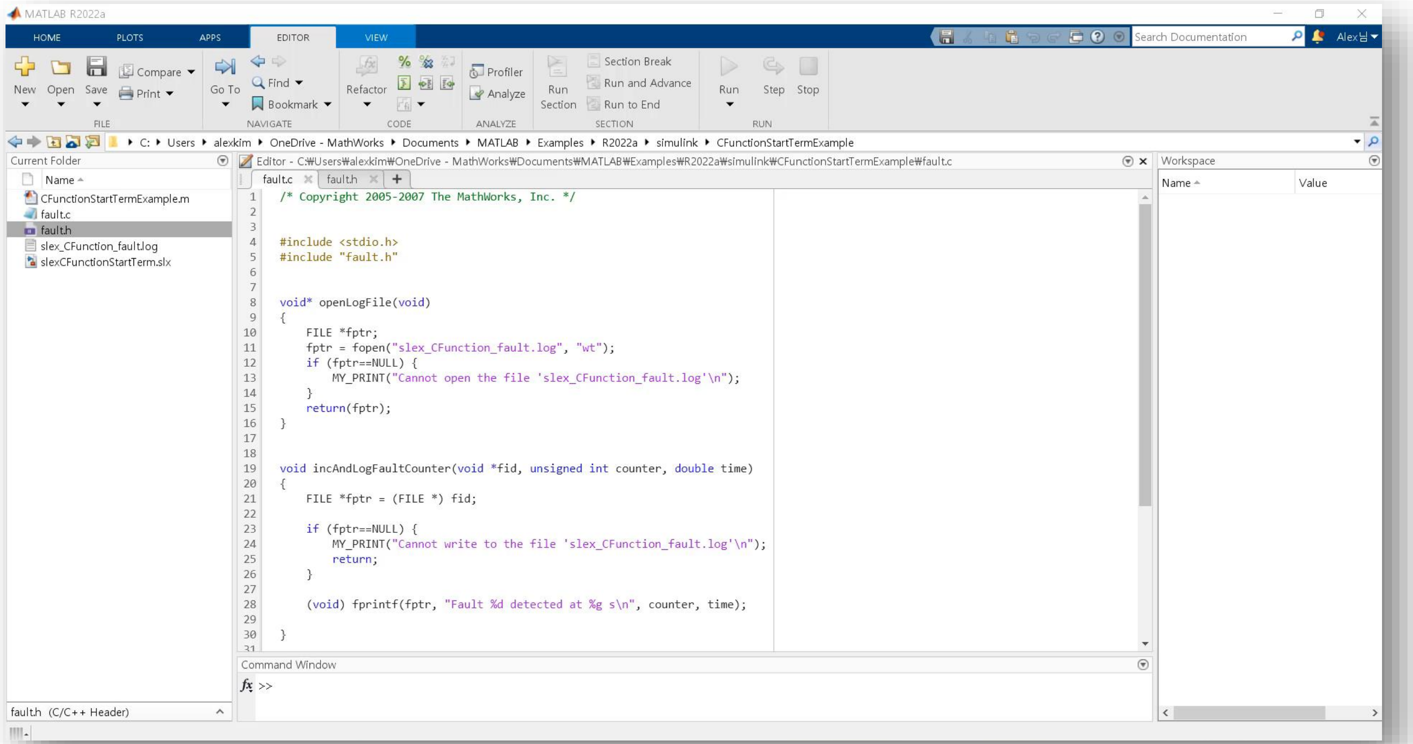




C Function Block

Support Persistent Scope and specify different code

- Write C code directly in your models
  - Call multiple external functions
- Support Persistent scope
- Specify different code for code generation and simulation using the flag *MATLAB_MEX_FILE*
- Interface directly with C++ classes in R2022a

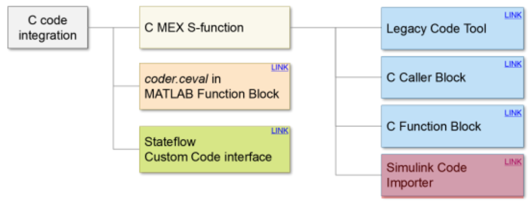


Editor - C:\Users\alexkim\OneDrive - MathWorks\Documents\MATLAB\Examples\R2022a\simulink\CFunctionStartTermExample\fault.c

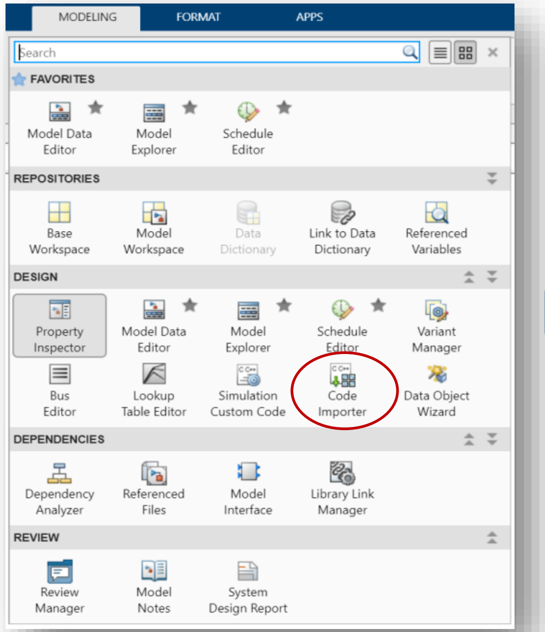
```c
/* Copyright 2005-2007 The MathWorks, Inc. */


#include <stdio.h>
#include "fault.h"


void* openLogFile(void)
{
    FILE *fptr;
    fptr = fopen("slex_CFunction_fault.log", "wt");
    if (fptr==NULL) {
        MY_PRINT("Cannot open the file 'slex_CFunction_fault.log'\n");
    }
    return(fptr);
}


void incAndLogFaultCounter(void *fid, unsigned int counter, double time)
{
    FILE *fptr = (FILE *) fid;

    if (fptr==NULL) {
        MY_PRINT("Cannot write to the file 'slex_CFunction_fault.log'\n");
        return;
    }

    (void) fprintf(fptr, "Fault %d detected at %g s\n", counter, time);

}
```

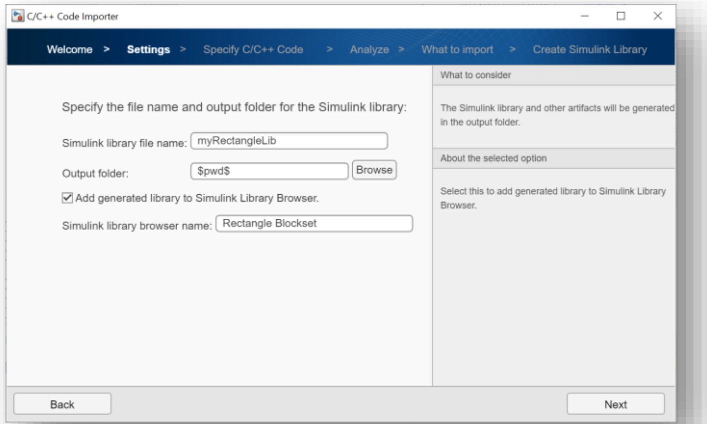
Simulink Code Importer

Import C code as reusable Simulink libraries



- Import C Code as reusable Simulink libraries
  - Block representation of C Code algorithms using C caller
- Wizard UI provides a step-by-step guidance
- Intuitive setup MATLAB APIs also available
- Integrated with Simulink Test for V&V workflow




C/C++ Code Importer

Welcome > Settings > Specify C/C++ Code > Analyze > What to import > Create Simulink Library

Specify the file name and output folder for the Simulink library:

Simulink library file name: myRectangleLib

Output folder: pwd Browse

[✓] Add generated library to Simulink Library Browser.

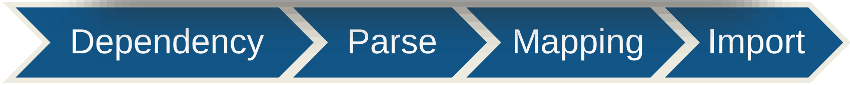
Simulink library browser name: Rectangle Blockset

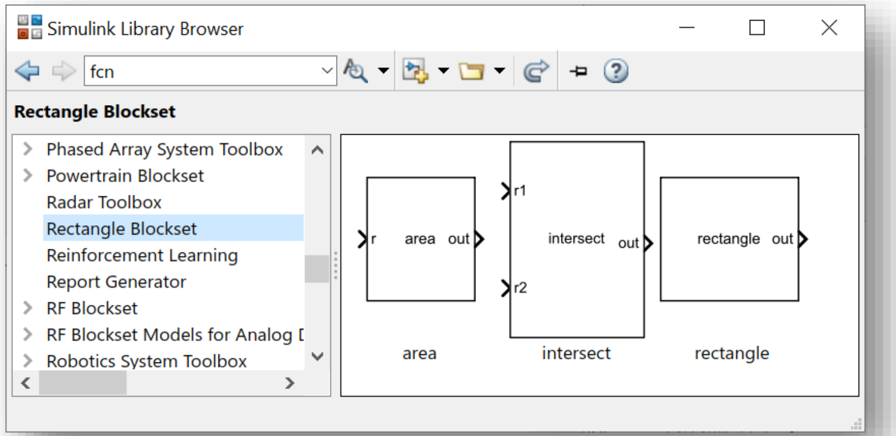
What to consider

The Simulink library and other artifacts will be generated in the output folder.

About the selected option

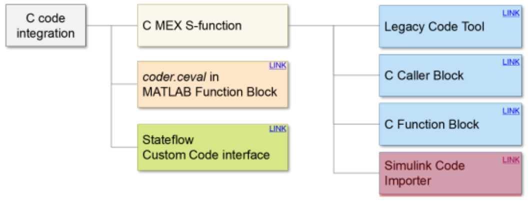
Select this to add generated library to Simulink Library Browser.

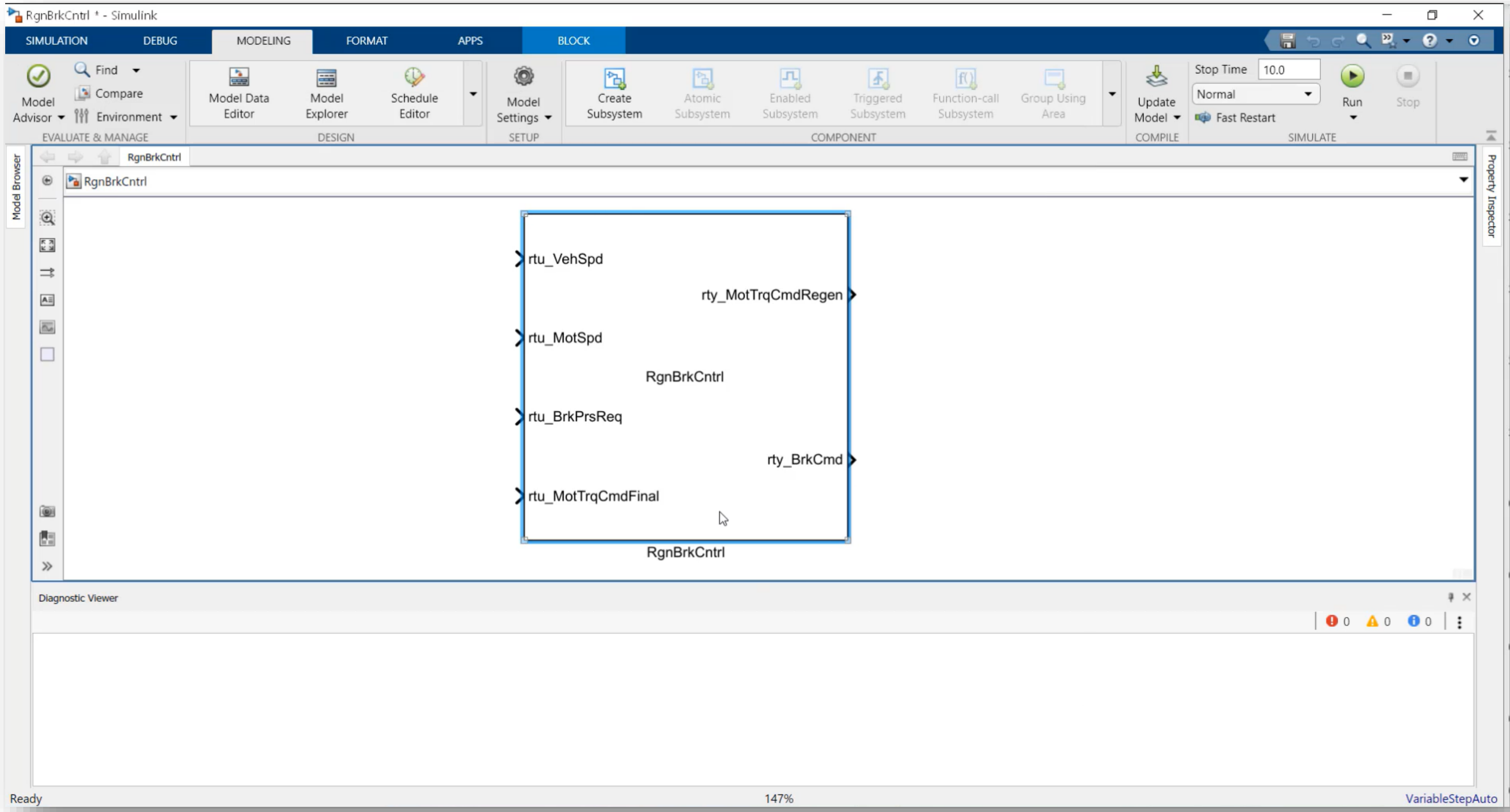




DEMO: Simulink Code Importer

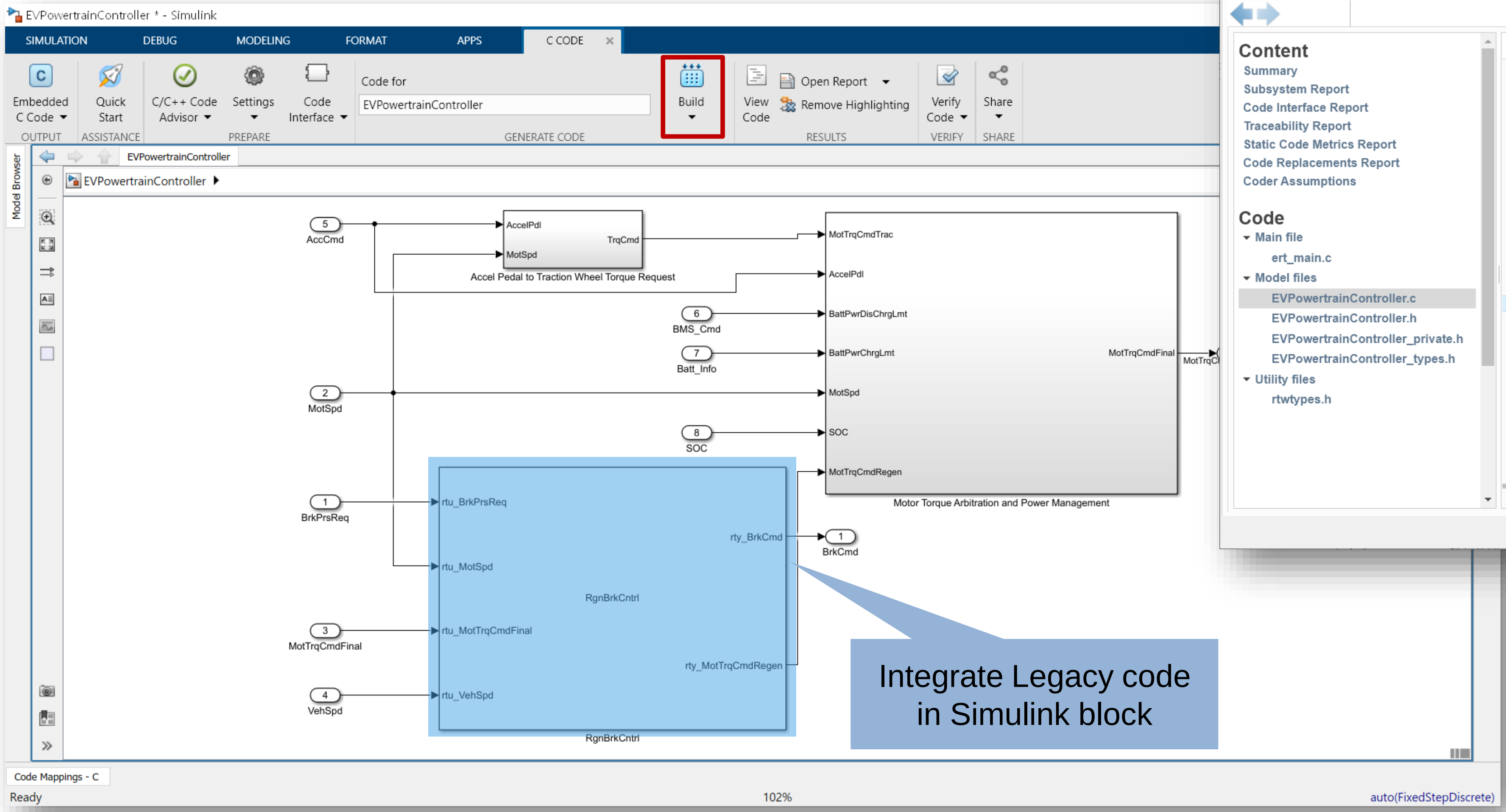
Integrating Legacy Code using Simulink Code Importer

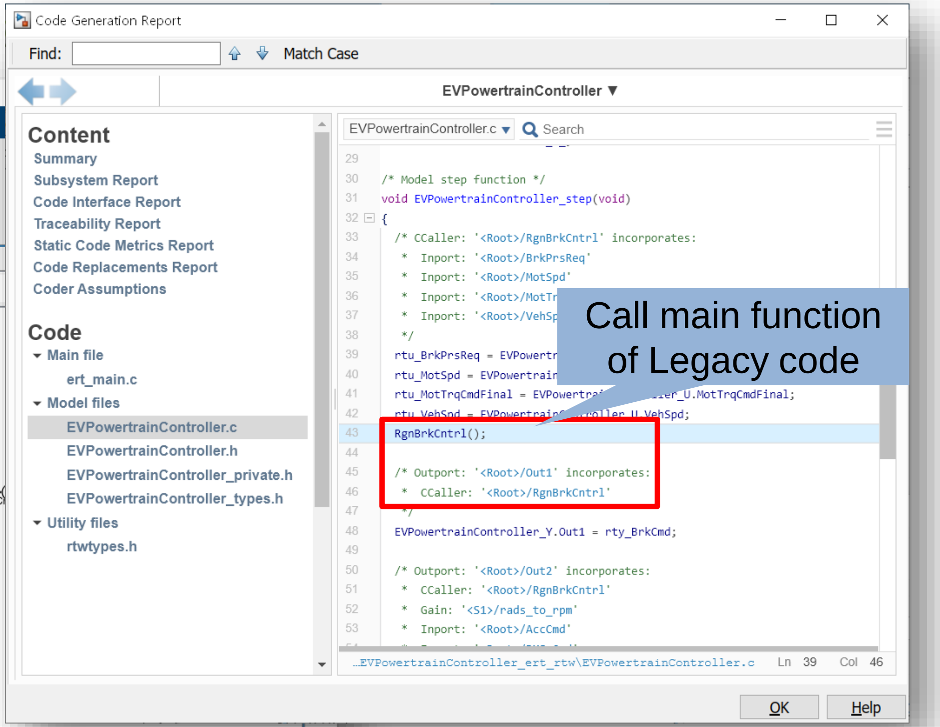




Code Generation in Integrated Model

Integrating Legacy Code in Simulink and Generating Code





```
/* Model step function */
void EVPowertrainController_step(void)
{
  /* CCaller: '<Root>/RgnBrkCntrl' incorporates:
   *  Inport: '<Root>/BrkPrsReq'
   *  Inport: '<Root>/MotSpd'
   *  Inport: '<Root>/MotTr
   *  Inport: '<Root>/VehSp
   */
  rtu_BrkPrsReq = EVPowertr
  rtu_MotSpd = EVPowertrain
  rtu_MotTrqCmdFinal = EVPowertra              ler_U.MotTrqCmdFinal;
  rtu_VehSpd = EVPowertrainController_U.VehSpd;
  RgnBrkCntrl();

  /* Outport: '<Root>/Out1' incorporates:
   *  CCaller: '<Root>/RgnBrkCntrl'
   */
  EVPowertrainController_Y.Out1 = rty_BrkCmd;

  /* Outport: '<Root>/Out2' incorporates:
   *  CCaller: '<Root>/RgnBrkCntrl'
   *  Gain: '<S1>/rads_to_rpm'
   *  Inport: '<Root>/AccCmd'
```

…EVPowertrainController_ert_rtw\EVPowertrainController.c   Ln 39   Col 46

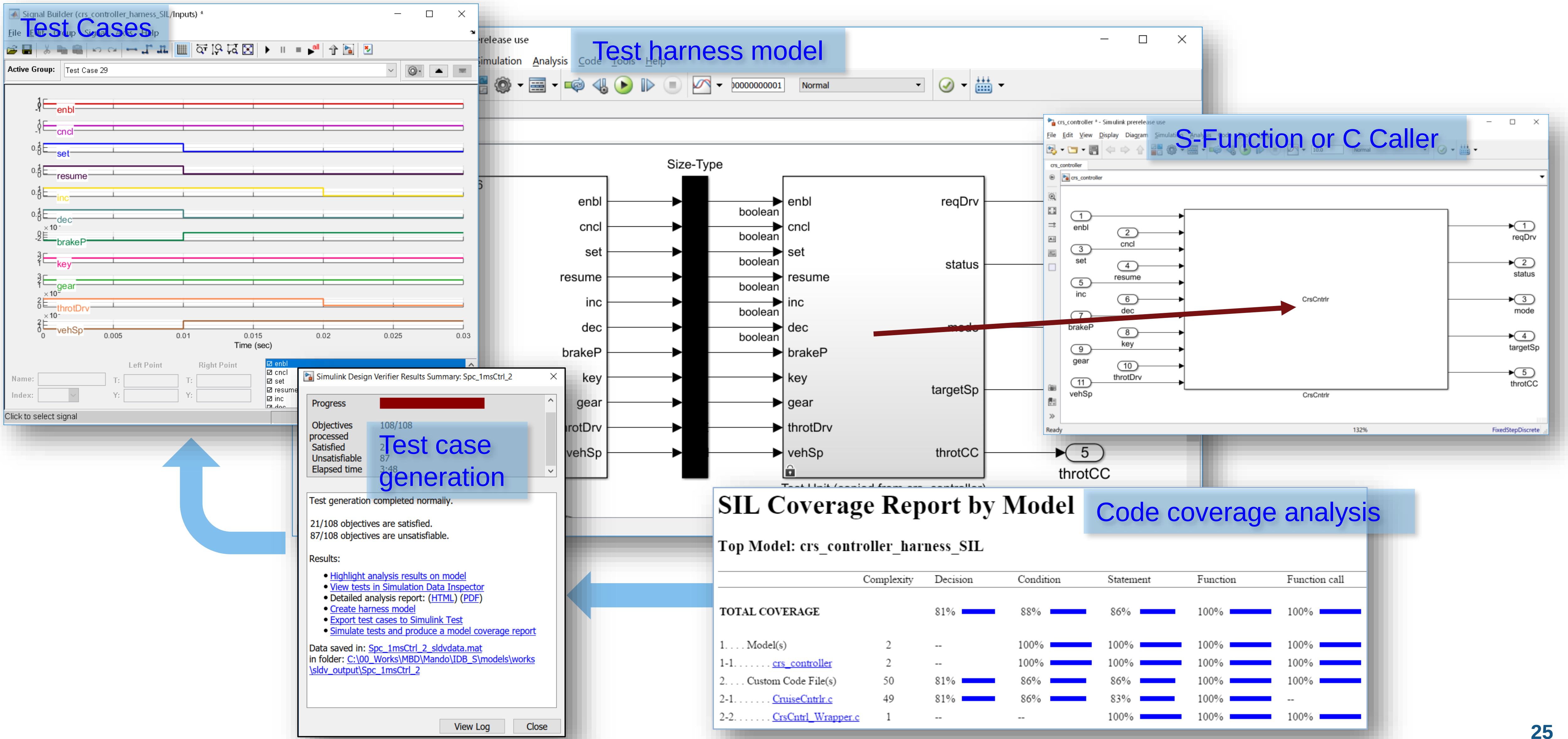
Call main function of Legacy code

Integrate Legacy code in Simulink block

Agenda

- How to get started MBD with Legacy Code?
- Legacy Code Integration using Simulink
- **Legacy Code Verification**
- Key Takeaways

Why Using Simulink for Legacy Code Testing?



Test generation completed normally.

21/108 objectives are satisfied.
87/108 objectives are unsatisfiable.

Results:

- Highlight analysis results on model
- View tests in Simulation Data Inspector
- Detailed analysis report: (HTML) (PDF)
- Create harness model
- Export test cases to Simulink Test
- Simulate tests and produce a model coverage report

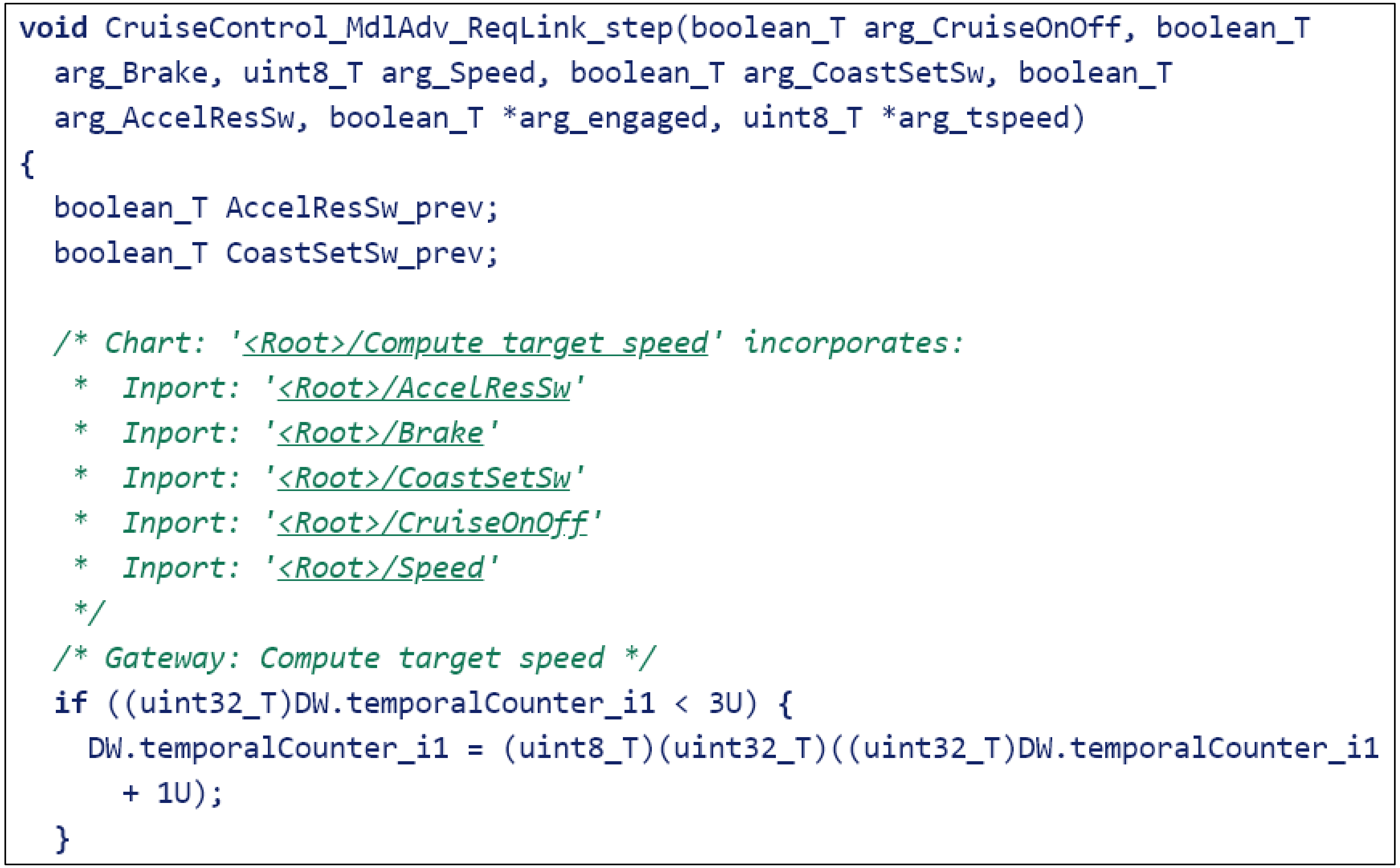
Data saved in: Spc_1msCtrl_2_sldvdata.mat
in folder: C:\00_Works\MBD\Mando\IDB_S\models\works\sldv_output\Spc_1msCtrl_2

SIL Coverage Report by Model

Top Model: crs_controller_harness_SIL

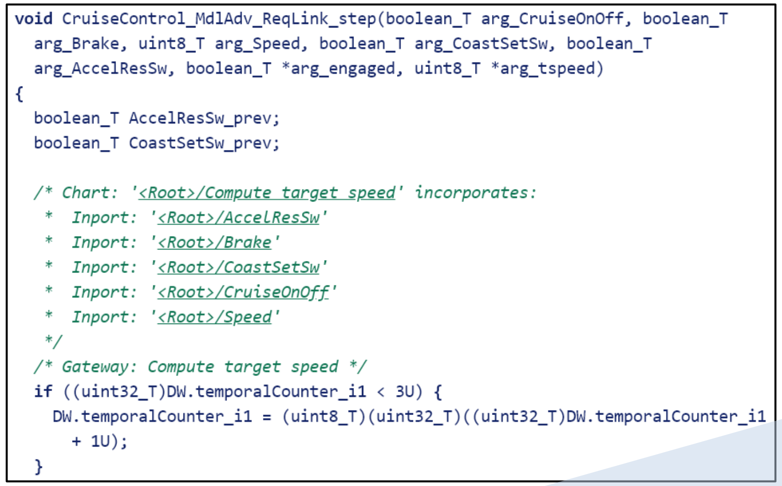
	Complexity	Decision	Condition	Statement	Function	Function call
TOTAL COVERAGE		81%	88%	86%	100%	100%
1. . . . Model(s)	2	--	100%	100%	100%	100%
1-1. . . . . . . crs_controller	2	--	100%	100%	100%	100%
2. . . . Custom Code File(s)	50	81%	86%	86%	100%	100%
2-1. . . . . . . CruiseCntrlr.c	49	81%	86%	83%	100%	--
2-2. . . . . . . CrsCntrl_Wrapper.c	1	--	--	100%	100%	100%

Legacy Code Verification Workflow

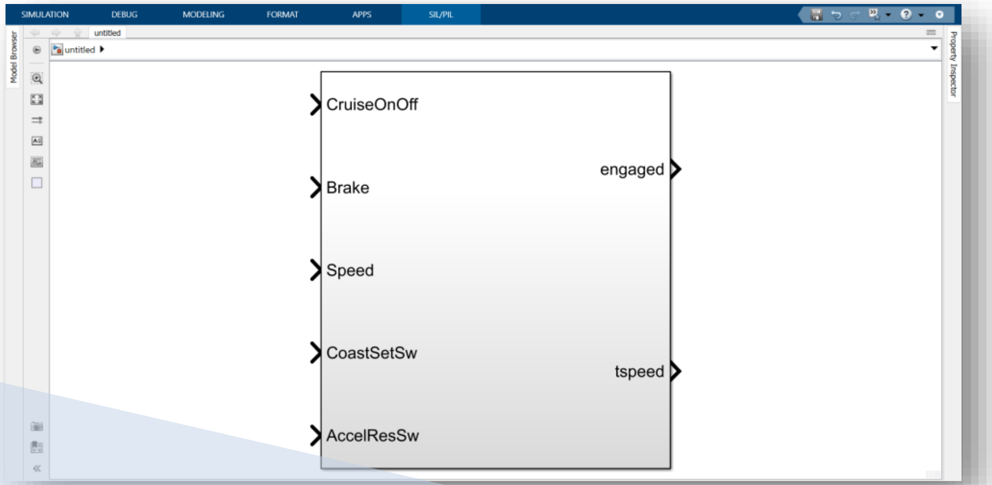


```
void CruiseControl_MdlAdv_ReqLink_step(boolean_T arg_CruiseOnOff, boolean_T
  arg_Brake, uint8_T arg_Speed, boolean_T arg_CoastSetSw, boolean_T
  arg_AccelResSw, boolean_T *arg_engaged, uint8_T *arg_tspeed)
{
  boolean_T AccelResSw_prev;
  boolean_T CoastSetSw_prev;

  /* Chart: '<Root>/Compute target speed' incorporates:
   *  Inport: '<Root>/AccelResSw'
   *  Inport: '<Root>/Brake'
   *  Inport: '<Root>/CoastSetSw'
   *  Inport: '<Root>/CruiseOnOff'
   *  Inport: '<Root>/Speed'
   */
  /* Gateway: Compute target speed */
  if ((uint32_T)DW.temporalCounter_i1 < 3U) {
    DW.temporalCounter_i1 = (uint8_T)(uint32_T)((uint32_T)DW.temporalCounter_i1
      + 1U);
  }
```

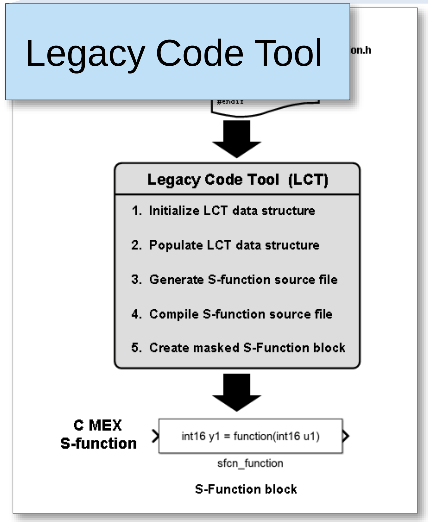
Legacy Code Verification Workflow



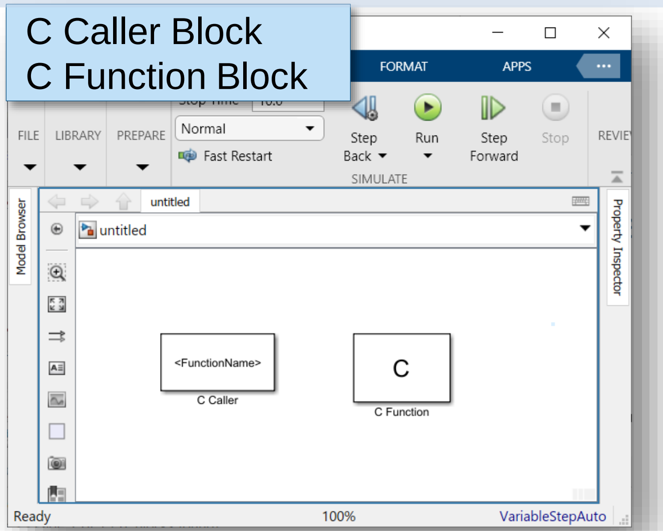
```
void CruiseControl_MdlAdv_ReqLink_step(boolean_T arg_CruiseOnOff, boolean_T
  arg_Brake, uint8_T arg_Speed, boolean_T arg_CoastSetSw, boolean_T
  arg_AccelResSw, boolean_T *arg_engaged, uint8_T *arg_tspeed)
{
  boolean_T AccelResSw_prev;
  boolean_T CoastSetSw_prev;

  /* Chart: '<Root>/Compute target speed' incorporates:
   *  Inport: '<Root>/AccelResSw'
   *  Inport: '<Root>/Brake'
   *  Inport: '<Root>/CoastSetSw'
   *  Inport: '<Root>/CruiseOnOff'
   *  Inport: '<Root>/Speed'
   */
  /* Gateway: Compute target speed */
  if ((uint32_T)DW.temporalCounter_i1 < 3U) {
    DW.temporalCounter_i1 = (uint8_T)(uint32_T)((uint32_T)DW.temporalCounter_i1
      + 1U);
  }
```



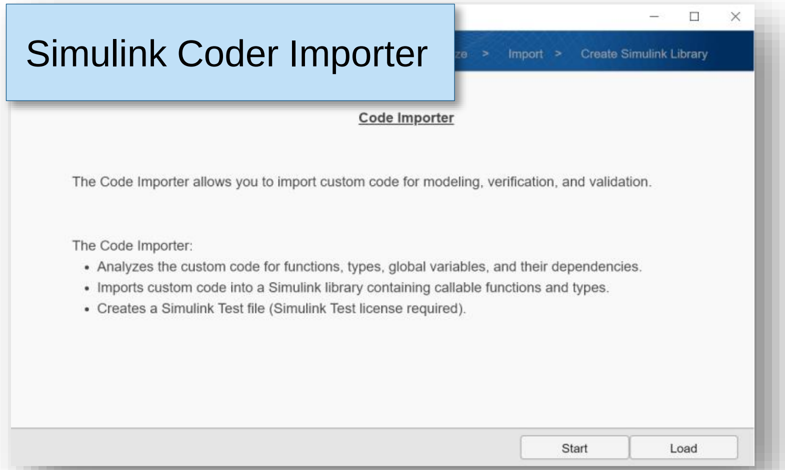
Legacy Code Tool


on.h
Legacy Code Tool (LCT)
1. Initialize LCT data structure
2. Populate LCT data structure
3. Generate S-function source file
4. Compile S-function source file
5. Create masked S-Function block
C MEX S-function
int16 y1 = function(int16 u1)
sfcn_function
S-Function block

C Caller Block
C Function Block



Simulink Coder Importer

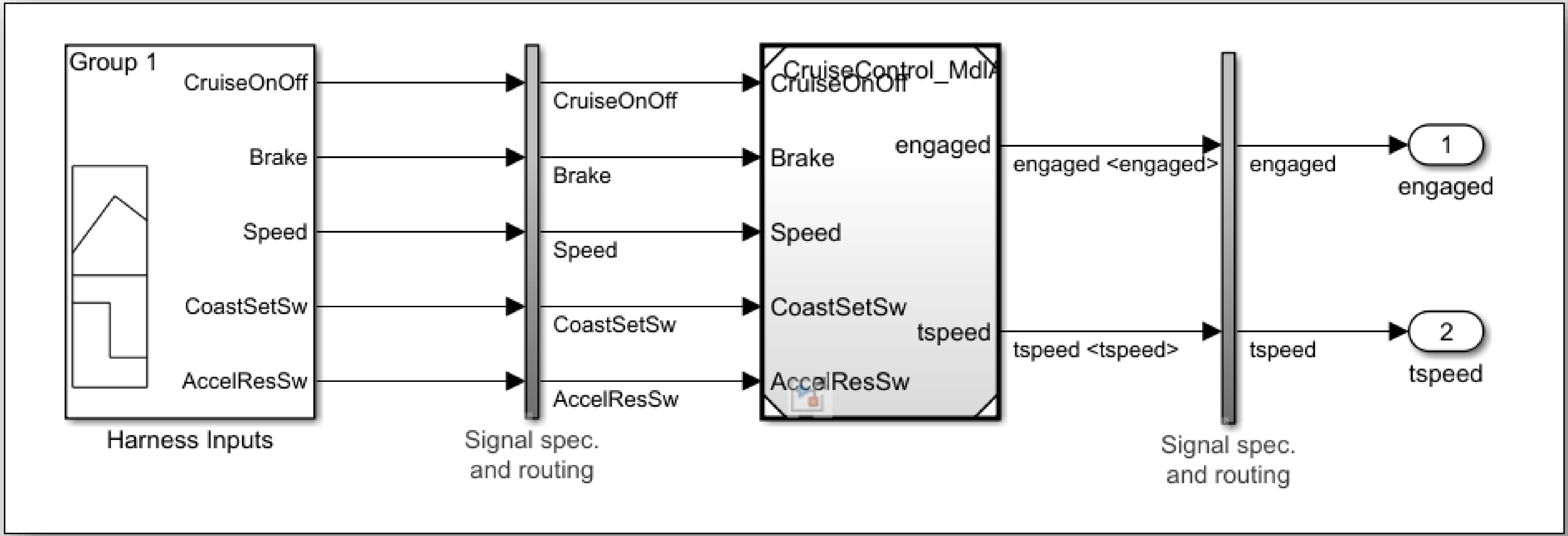


Code Importer

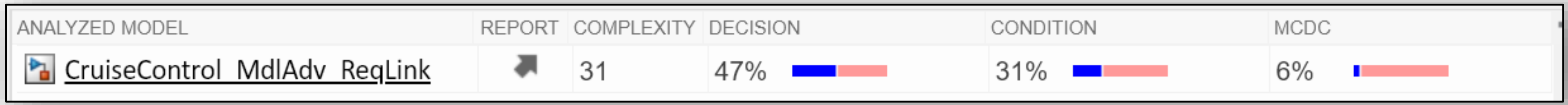
The Code Importer allows you to import custom code for modeling, verification, and validation.

The Code Importer:
- Analyzes the custom code for functions, types, global variables, and their dependencies.
- Imports custom code into a Simulink library containing callable functions and types.
- Creates a Simulink Test file (Simulink Test license required).

Legacy Code Verification Workflow

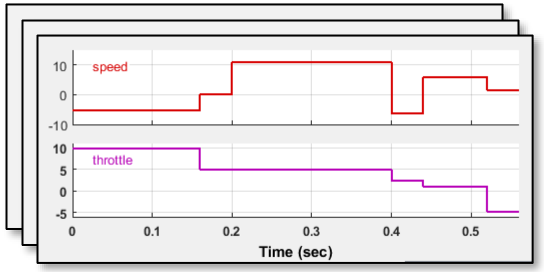


Legacy Code Verification Workflow

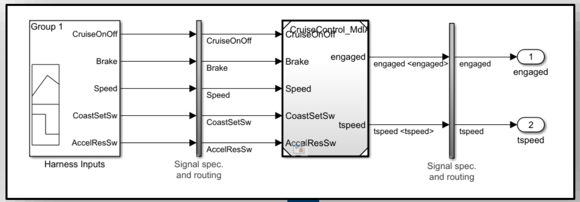


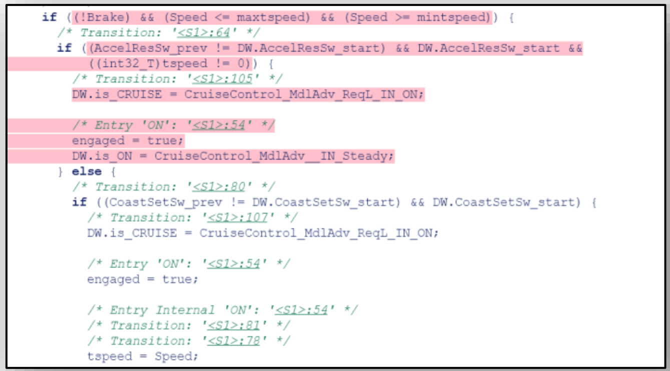
ANALYZED MODEL	REPORT	COMPLEXITY	DECISION	CONDITION	MCDC
CruiseControl_MdlAdv_ReqLink		31	47%	31%	6%

Partial Coverage



Test Cases



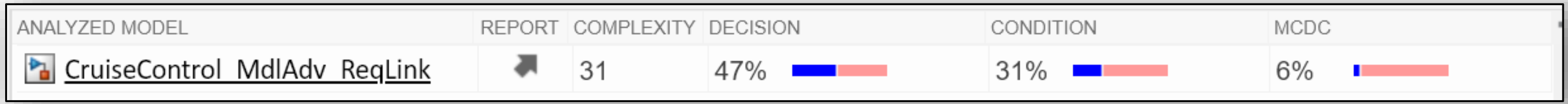


```
if ((!Brake) && (Speed <= maxtspeed) && (Speed >= mintspeed)) {
  /* Transition: '<S1>:64' */
  if ((AccelResSw_prev != DW.AccelResSw_start) && DW.AccelResSw_start &&
      ((int32_T)tspeed != 0)) {
    /* Transition: '<S1>:105' */
    DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

    /* Entry 'ON': '<S1>:54' */
    engaged = true;
    DW.is_ON = CruiseControl_MdlAdv__IN_Steady;
  } else {
    /* Transition: '<S1>:80' */
    if ((CoastSetSw_prev != DW.CoastSetSw_start) && DW.CoastSetSw_start) {
      /* Transition: '<S1>:107' */
      DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

      /* Entry 'ON': '<S1>:54' */
      engaged = true;

      /* Entry Internal 'ON': '<S1>:54' */
      /* Transition: '<S1>:81' */
      /* Transition: '<S1>:78' */
      tspeed = Speed;
```

Test Generator

Simulink Design Verifier

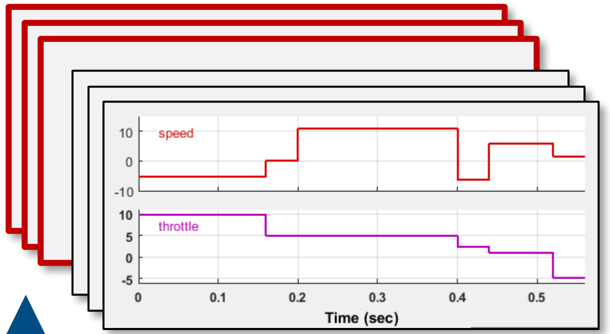
Legacy Code Verification Workflow

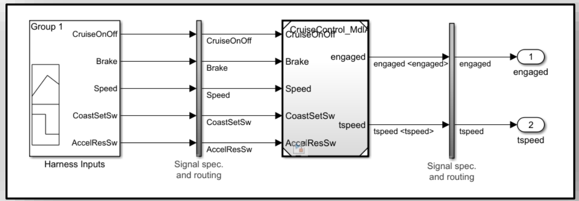


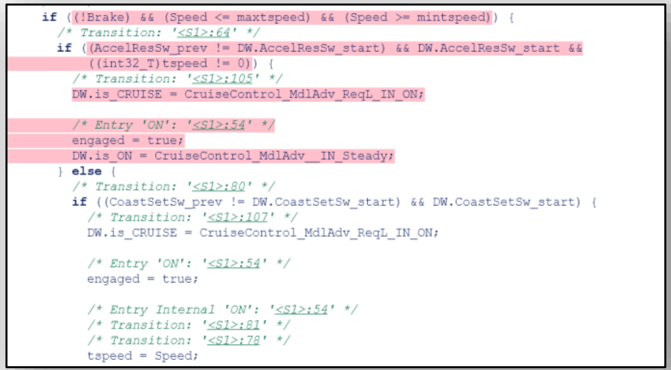
ANALYZED MODEL	REPORT	COMPLEXITY	DECISION	CONDITION	MCDC
CruiseControl_MdlAdv_ReqLink		31	47%	31%	6%

Partial Coverage

New Test Cases







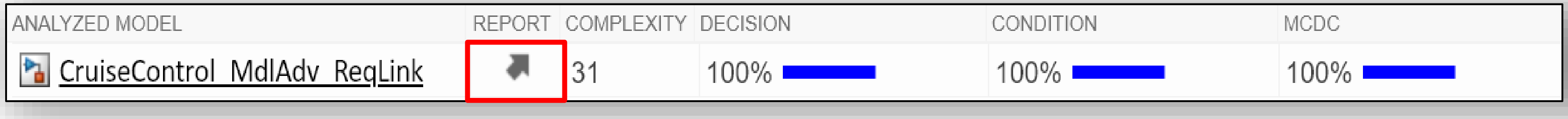
```
if ((!Brake) && (Speed <= maxtspeed) && (Speed >= mintspeed)) {
  /* Transition: '<S1>:64' */
  if ((AccelResSw_prev != DW.AccelResSw_start) && DW.AccelResSw_start &&
      ((int32_T)tspeed != 0)) {
    /* Transition: '<S1>:105' */
    DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

    /* Entry 'ON': '<S1>:54' */
    engaged = true;
    DW.is_ON = CruiseControl_MdlAdv__IN_Steady;
  } else {
    /* Transition: '<S1>:80' */
    if ((CoastSetSw_prev != DW.CoastSetSw_start) && DW.CoastSetSw_start) {
      /* Transition: '<S1>:107' */
      DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

      /* Entry 'ON': '<S1>:54' */
      engaged = true;

      /* Entry Internal 'ON': '<S1>:54' */
      /* Transition: '<S1>:81' */
      /* Transition: '<S1>:78' */
      tspeed = Speed;
```

Test Cases

Test Generator

Simulink Design Verifier

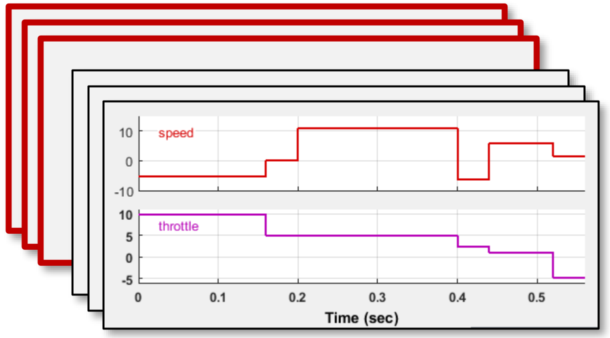
Legacy Code Verification Workflow



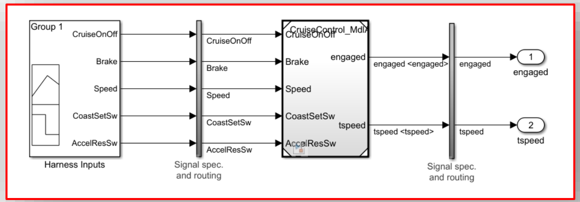
ANALYZED MODEL	REPORT	COMPLEXITY	DECISION	CONDITION	MCDC
CruiseControl_MdlAdv_ReqLink		31	100%	100%	100%

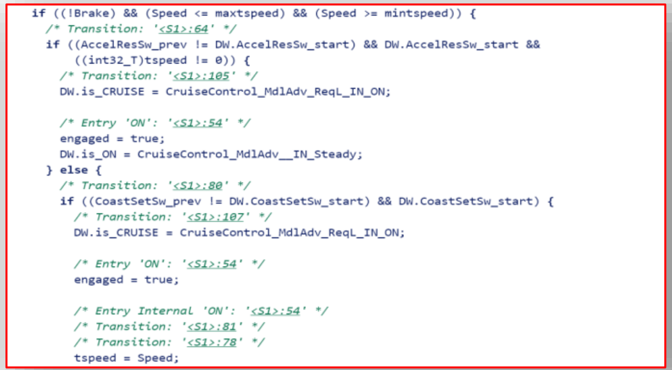
Higher Coverage

New Test Cases

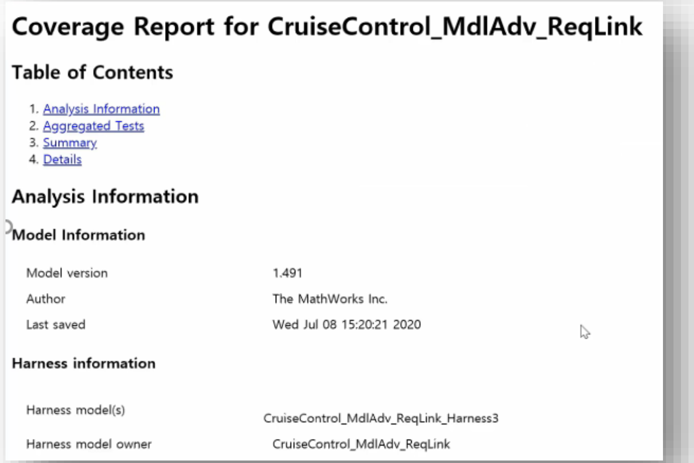


Test Cases





```
if ((!Brake) && (Speed <= maxtspeed) && (Speed >= mintspeed)) {
  /* Transition: '<S1>:64' */
  if ((AccelResSw_prev != DW.AccelResSw_start) && DW.AccelResSw_start &&
      ((int32_T)tspeed != 0)) {
    /* Transition: '<S1>:105' */
    DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

    /* Entry 'ON': '<S1>:54' */
    engaged = true;
    DW.is_ON = CruiseControl_MdlAdv__IN_Steady;
  } else {
    /* Transition: '<S1>:80' */
    if ((CoastSetSw_prev != DW.CoastSetSw_start) && DW.CoastSetSw_start) {
      /* Transition: '<S1>:107' */
      DW.is_CRUISE = CruiseControl_MdlAdv_ReqL_IN_ON;

      /* Entry 'ON': '<S1>:54' */
      engaged = true;

      /* Entry Internal 'ON': '<S1>:54' */
      /* Transition: '<S1>:81' */
      /* Transition: '<S1>:78' */
      tspeed = Speed;
```



Coverage Report for CruiseControl_MdlAdv_ReqLink

Table of Contents

Analysis Information

Model Information

Model version	1.491
Author	The MathWorks Inc.
Last saved	Wed Jul 08 15:20:21 2020

Harness information

Harness model(s)	CruiseControl_MdlAdv_ReqLink_Harness3
Harness model owner	CruiseControl_MdlAdv_ReqLink

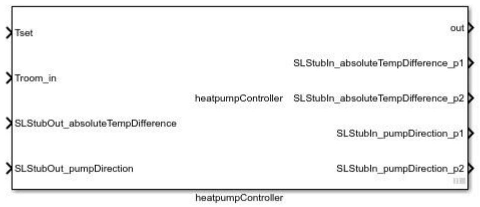
Generate Report

Simulink Code Importer in Simulink Test

- Supports unit test and integration test
- Test setup automation
  - Importing c code to Simulink C Caller block, creating harness, creating test file and running the test file

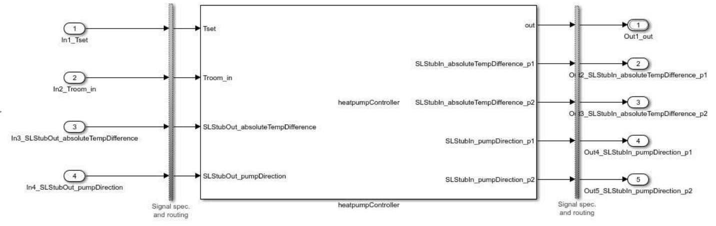
Test Setup Automation

C code



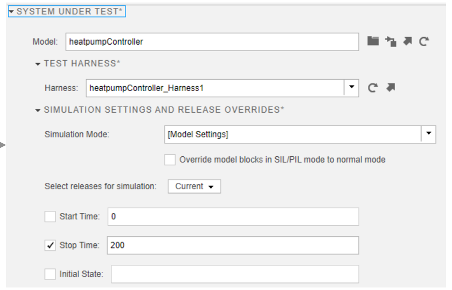
Tset
Troom_in
SLStubOut_absoluteTempDifference
SLStubOut_pumpDirection
out
SLStubIn_absoluteTempDifference_p1
SLStubIn_absoluteTempDifference_p2
SLStubIn_pumpDirection_p1
SLStubIn_pumpDirection_p2
heatpumpController
heatpumpController

Converts the C code functions to Simulink C Caller blocks



1 In1_Tset
2 In2_Troom_in
3 In3_SLStubOut_absoluteTempDifference
4 In4_SLStubOut_pumpDirection
Tset
Troom_in
SLStubOut_absoluteTempDifference
SLStubOut_pumpDirection
heatpumpController
out
SLStubIn_absoluteTempDifference_p1
SLStubIn_absoluteTempDifference_p2
SLStubIn_pumpDirection_p1
SLStubIn_pumpDirection_p2
heatpumpController
Signal spec. and routing
Signal spec. and routing
1 Out1_out
2 Out2_SLStubIn_absoluteTempDifference_p1
3 Out3_SLStubIn_absoluteTempDifference_p2
4 Out4_SLStubIn_pumpDirection_p1
5 Out5_SLStubIn_pumpDirection_p2

Creates an internal harness for each Simulink C Caller block



SYSTEM UNDER TEST*

Model: heatpumpController

TEST HARNESS*

Harness: heatpumpController_Harness1

SIMULATION SETTINGS AND RELEASE OVERRIDES*

Simulation Mode: [Model Settings]

☐ Override model blocks in SIL/PIL mode to normal mode

Select releases for simulation: Current

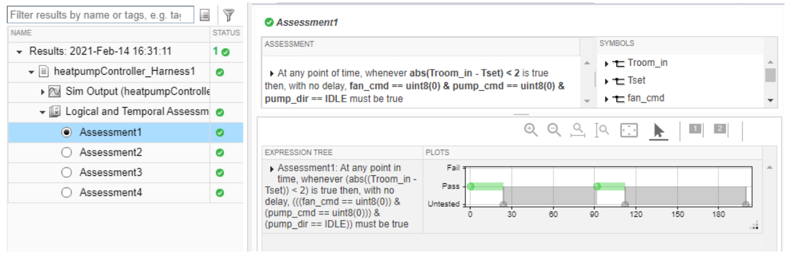
☐ Start Time: 0

☑ Stop Time: 200

☐ Initial State:

Generates a test file

Test Result



Filter results by name or tags, e.g. ta

NAME	STATUS
Results: 2021-Feb-14 16:31:11	1 ✓
heatpumpController_Harness1	✓
Sim Output (heatpumpControlle	
Logical and Temporal Assessm	✓
Assessment1	✓
Assessment2	✓
Assessment3	✓
Assessment4	✓

Assessment1

ASSESSMENT

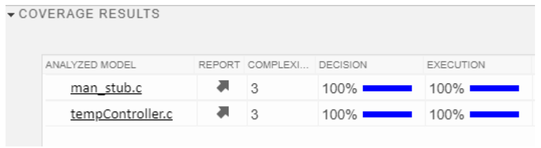
At any point of time, whenever **abs(Troom_in - Tset) < 2** is true then, with no delay, **fan_cmd == uint8(0) & pump_cmd == uint8(0) & pump_dir == IDLE** must be true

SYMBOLS

Troom_in
Tset
fan_cmd

EXPRESSION TREE

Assessment1: At any point in time, whenever (abs((Troom_in - Tset)) < 2) is true then, with no delay, (((fan_cmd == uint8(0)) & (pump_cmd == uint8(0))) & (pump_dir == IDLE)) must be true



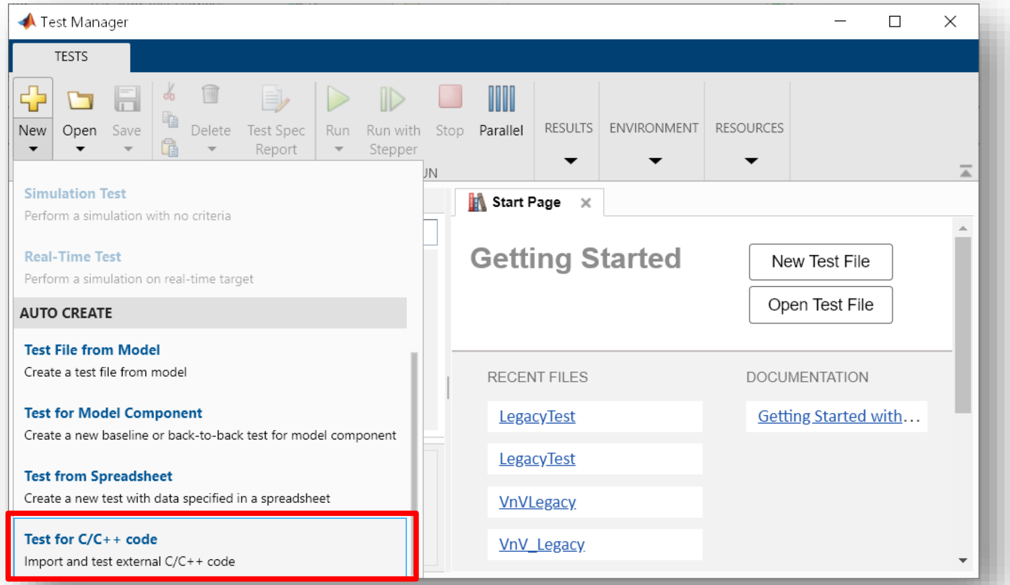
COVERAGE RESULTS

ANALYZED MODEL	REPORT	COMPLEXI...	DECISION	EXECUTION
man_stub.c		3	100%	100%
tempController.c		3	100%	100%

※ For unit tests, additionally creates a sandbox to isolate the imported functions.

Simulink Code Importer in Simulink Test

- Import legacy code using Simulink Code Importer in Simulink Test



Simulation Test
Perform a simulation with no criteria

Real-Time Test
Perform a simulation on real-time target

AUTO CREATE

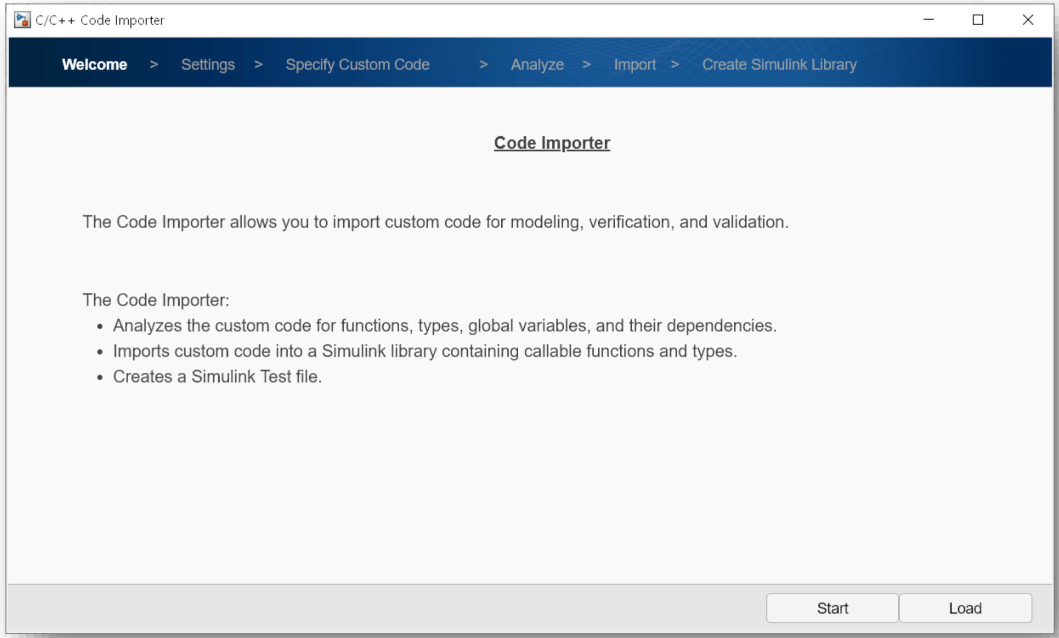
Test File from Model
Create a test file from model

Test for Model Component
Create a new baseline or back-to-back test for model component

Test from Spreadsheet
Create a new test with data specified in a spreadsheet

Test for C/C++ code
Import and test external C/C++ code

Test Manager


C/C++ Code Importer
Welcome > Settings > Specify Custom Code > Analyze > Import > Create Simulink Library
Start
Load

Code Importer

The Code Importer allows you to import custom code for modeling, verification, and validation.

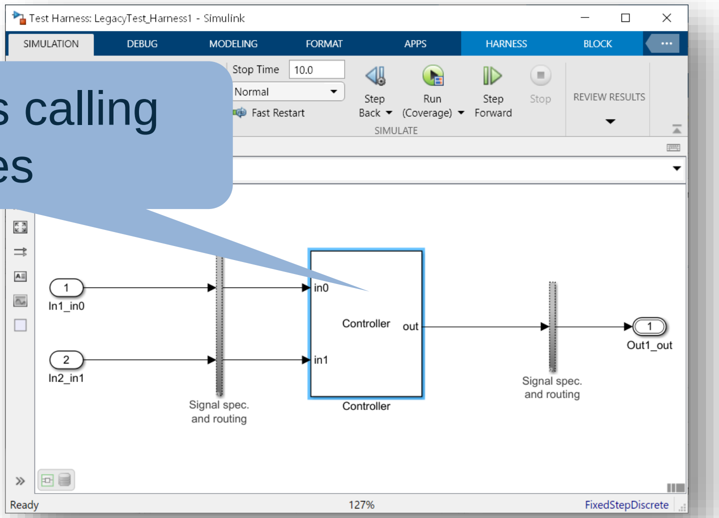
The Code Importer:
- Analyzes the custom code for functions, types, global variables, and their dependencies.
- Imports custom code into a Simulink library containing callable functions and types.
- Creates a Simulink Test file.

Simulink Code Importer Wizard

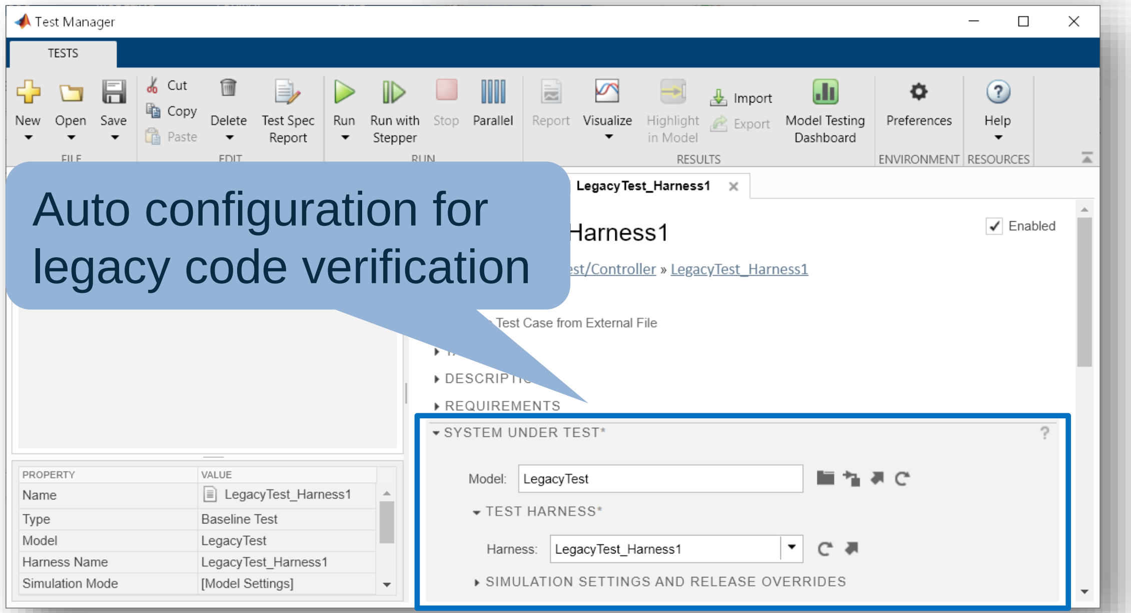
Simulink Code Importer in Simulink Test

- Build library model and test harness
- Automatically configured test setting in Test Manager

Test harness calling Legacy codes



Auto configuration for legacy code verification



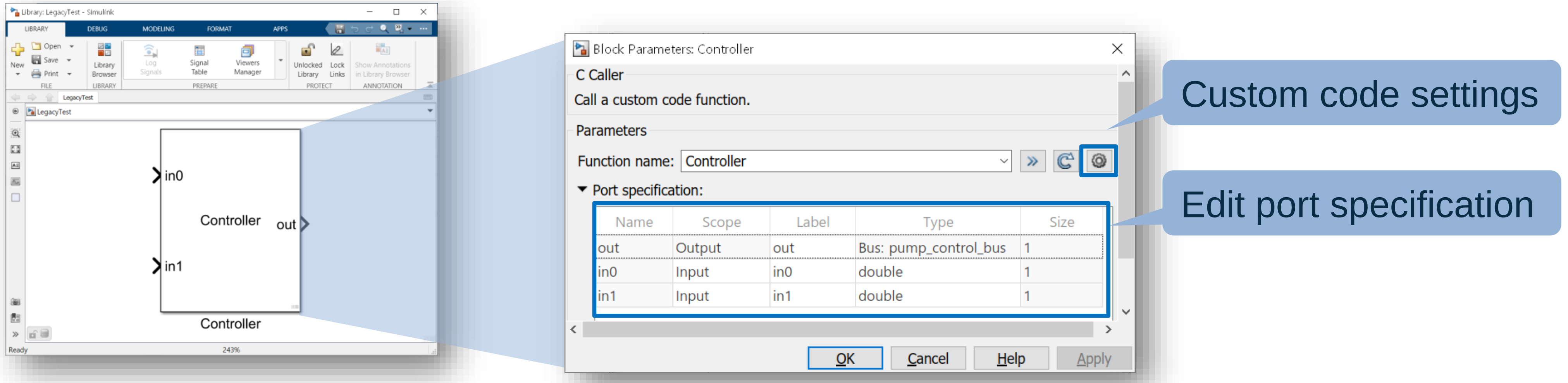
PROPERTY	VALUE
Name	LegacyTest_Harness1
Type	Baseline Test
Model	LegacyTest
Harness Name	LegacyTest_Harness1
Simulation Mode	[Model Settings]

Model: LegacyTest

Harness: LegacyTest_Harness1

Simulink Code Importer in Simulink Test

- Simulink Code Importer calls code using C Caller Block
  - Using C Caller block's features after building library
  - Edit port specification in Block Parameters
  - Configurate custom code settings



Block Parameters: Controller

C Caller

Call a custom code function.

Parameters

Function name: Controller

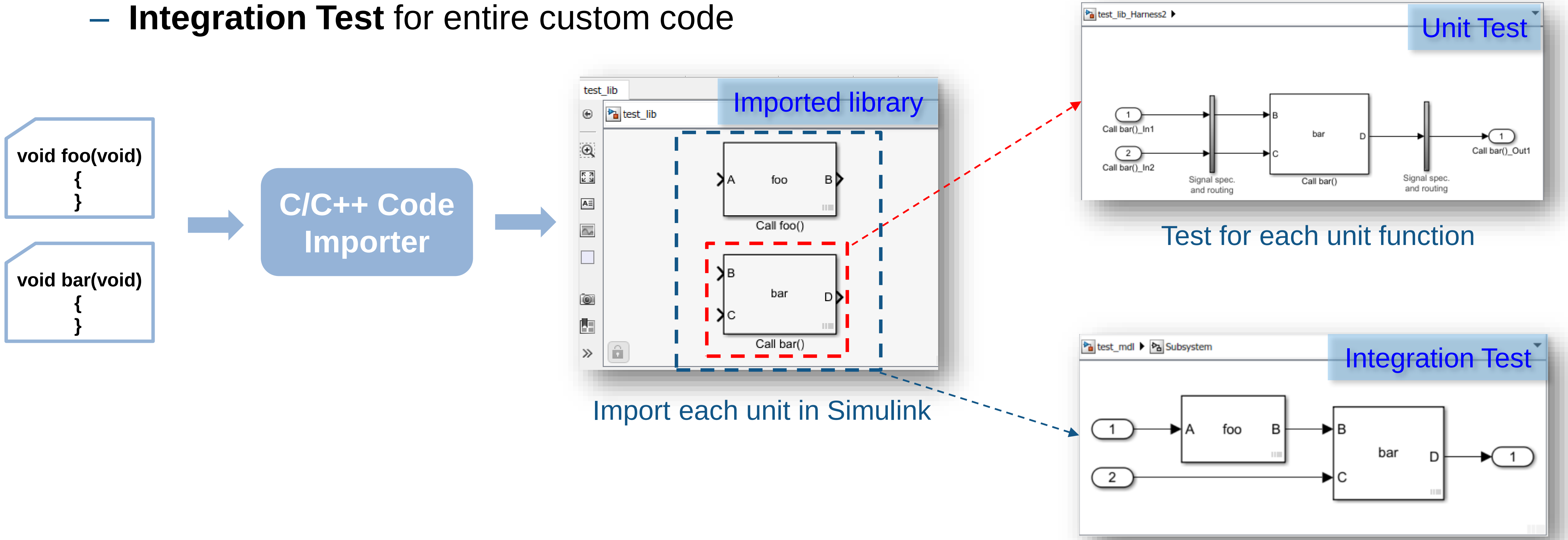
Port specification:

Name	Scope	Label	Type	Size
out	Output	out	Bus: pump_control_bus	1
in0	Input	in0	double	1
in1	Input	in1	double	1

OK Cancel Help Apply

Simulink Code Importer in Simulink Test

- Support two options for C code test
  - **Unit Test** for a subset of custom code
  - **Integration Test** for entire custom code



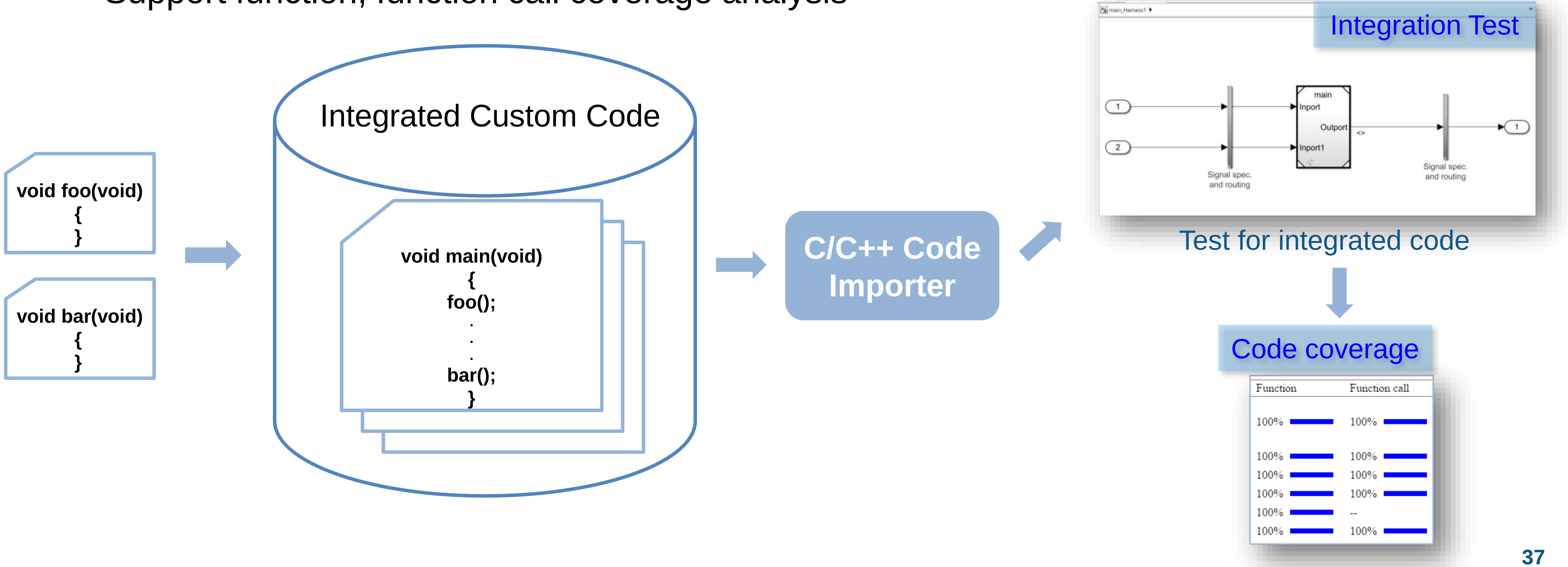
Import each unit in Simulink

Test for each unit function

Integrated model and test

Simulink Code Importer in Simulink Test

- Easy to support Integration Test using Simulink Code Importer
  - Import integrated custom code in Simulink
  - Support function, function call coverage analysis



```
void foo(void)
{
}
```

```
void bar(void)
{
}
```

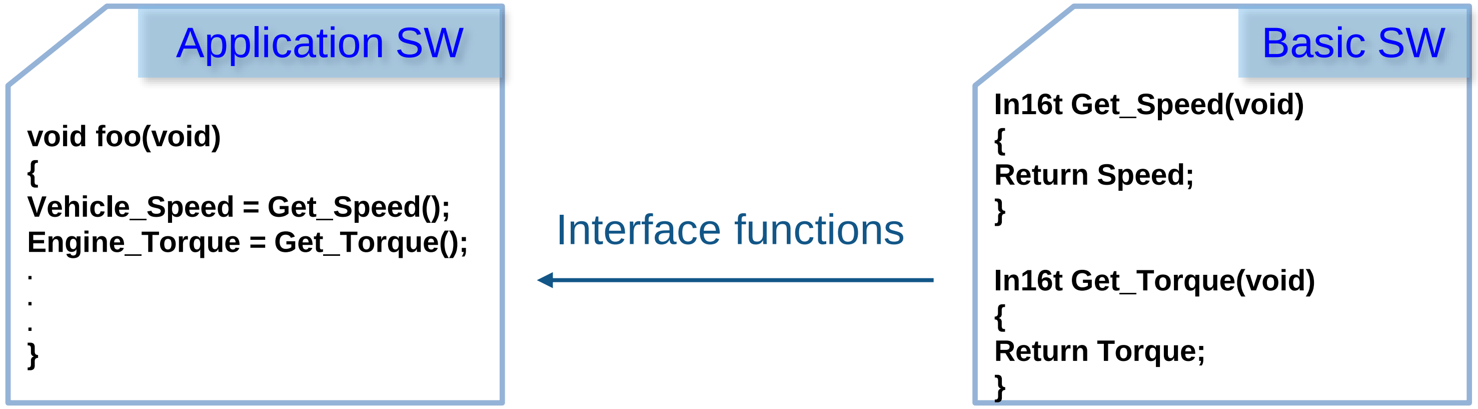
```
void main(void)
{
foo();
.
.
.
bar();
}
```

Function	Function call
100%	100%
100%	100%
100%	100%
100%	100%
100%	--
100%	100%

Simulink Code Importer in Simulink Test

- How to verify unit functions which include interface functions?

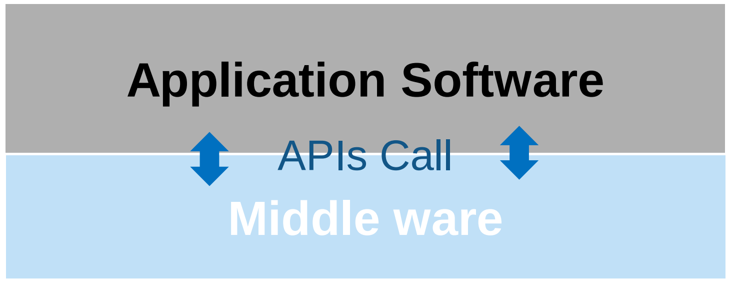
  Ex) Application SW uses interface functions which is provided from Basic SW



```
void foo(void)
{
Vehicle_Speed = Get_Speed();
Engine_Torque = Get_Torque();
.
.
.
}
```

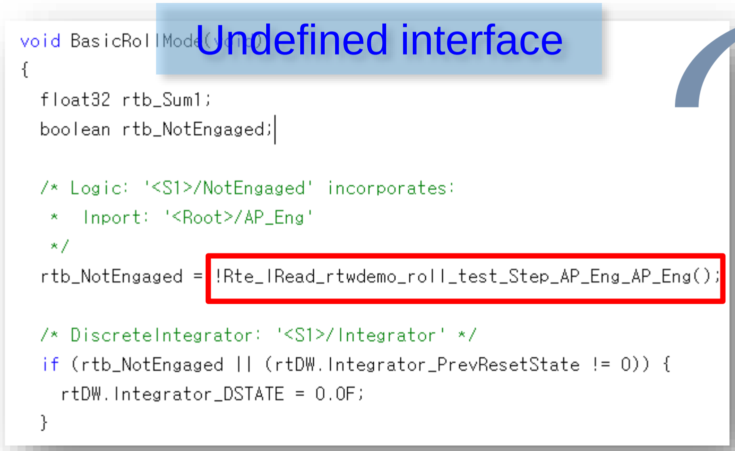
```
In16t Get_Speed(void)
{
Return Speed;
}

In16t Get_Torque(void)
{
Return Torque;
}
```

  Ex) Application SW uses middleware APIs for data interface such as AUTOSAR



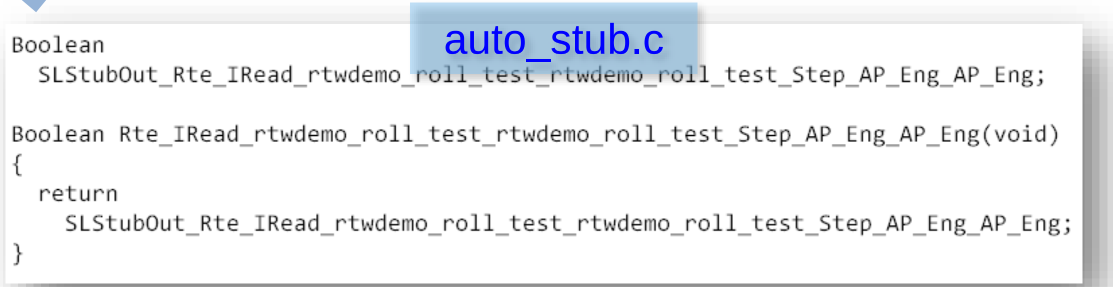
Simulink Code Importer in Simulink Test

- Create Sandbox for C code unit testing
  - **Auto-stub files**: Contains the *auto_stub.h* and *auto_stub.c* files, which are generated only if the imported code has undefined symbols



```
void BasicRollMode(void)
{
  float32 rtb_Sum1;
  boolean rtb_NotEngaged;

  /* Logic: '<S1>/NotEngaged' incorporates:
   *  Inport: '<Root>/AP_Eng'
   */
  rtb_NotEngaged = !Rte_IRead_rtwdemo_roll_test_Step_AP_Eng_AP_Eng();

  /* DiscreteIntegrator: '<S1>/Integrator' */
  if (rtb_NotEngaged || (rtDW.Integrator_PrevResetState != 0)) {
    rtDW.Integrator_DSTATE = 0.0F;
  }
```

Parser analysis &
auto stub generation

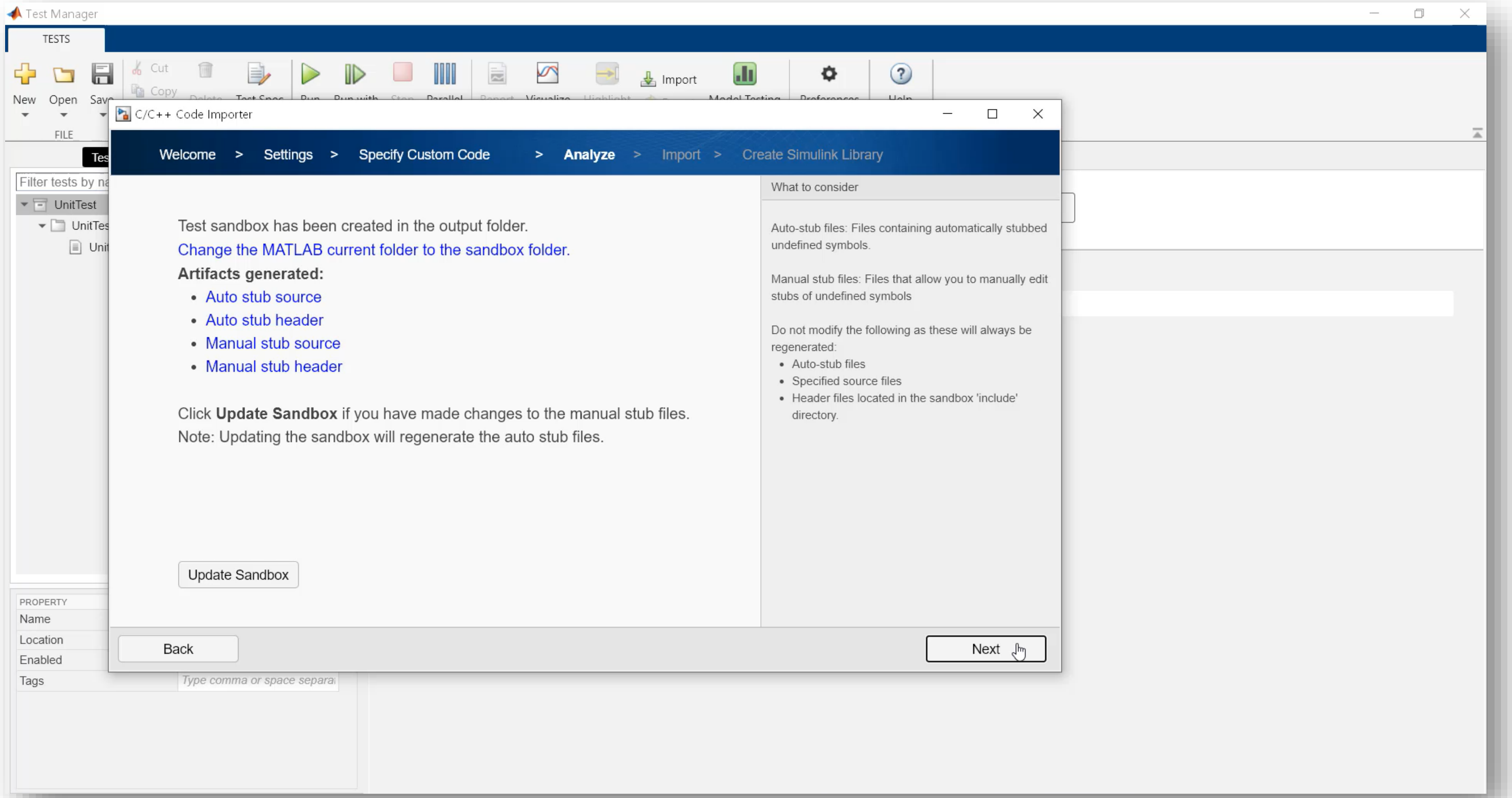


```
Boolean
  SLStubOut_Rte_IRead_rtwdemo_roll_test_rtwdemo_roll_test_Step_AP_Eng_AP_Eng;

Boolean Rte_IRead_rtwdemo_roll_test_rtwdemo_roll_test_Step_AP_Eng_AP_Eng(void)
{
  return
    SLStubOut_Rte_IRead_rtwdemo_roll_test_rtwdemo_roll_test_Step_AP_Eng_AP_Eng;
}
```

  - **Manual stub files**: Contains the *man_stub.h* and *man_stub.c* files, which you can use to manually stub symbols
  - **Aggregated header**: Contains all definitions of functions, interfaces which are related unit function

DEMO: Simulink Code Importer in Simulink Test

Import C code and Create Sandbox for unit testing



Test sandbox has been created in the output folder.
Change the MATLAB current folder to the sandbox folder.
Artifacts generated:
- Auto stub source
- Auto stub header
- Manual stub source
- Manual stub header

Click **Update Sandbox** if you have made changes to the manual stub files.
Note: Updating the sandbox will regenerate the auto stub files.

What to consider

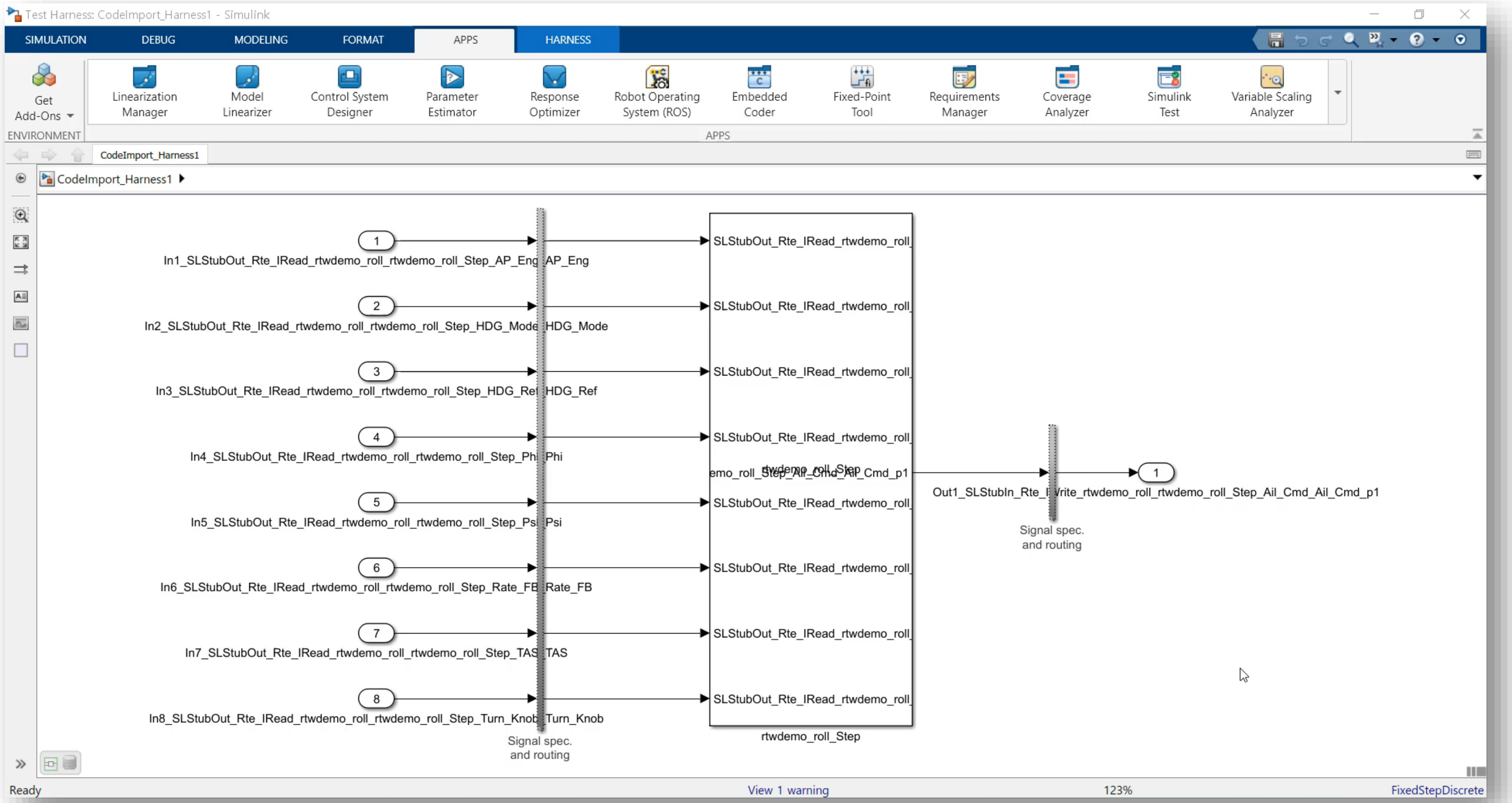
Auto-stub files: Files containing automatically stubbed undefined symbols.

Manual stub files: Files that allow you to manually edit stubs of undefined symbols

Do not modify the following as these will always be regenerated:
- Auto-stub files
- Specified source files
- Header files located in the sandbox 'include' directory.

DEMO: Simulink Code Importer in Simulink Test

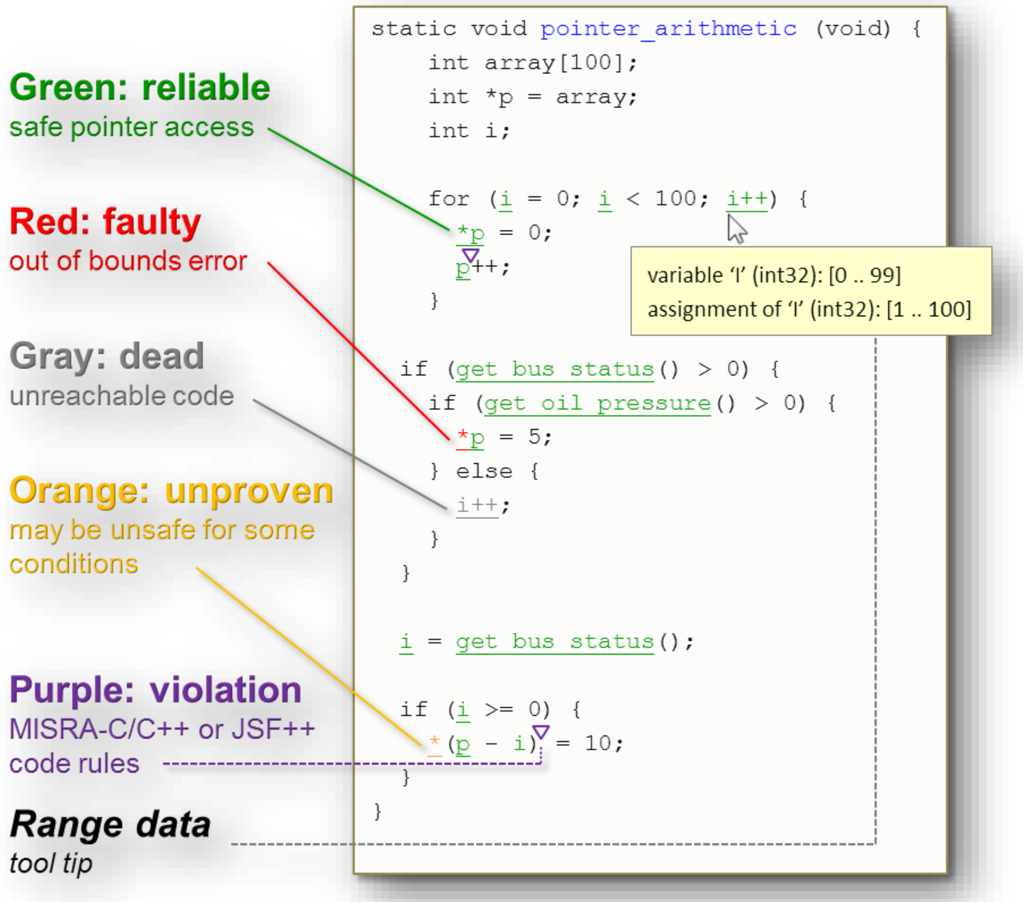
Test case generation and unit test in Test Manager



Test Harness: CodeImport_Harness1 - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
HARNESS
Get Add-Ons
ENVIRONMENT
Linearization Manager
Model Linearizer
Control System Designer
Parameter Estimator
Response Optimizer
Robot Operating System (ROS)
Embedded Coder
Fixed-Point Tool
Requirements Manager
Coverage Analyzer
Simulink Test
Variable Scaling Analyzer
APPS
CodeImport_Harness1
In1_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_AP_Eng
In2_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_HDG_Mode
In3_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_HDG_Ref
In4_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_Phi
In5_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_Psi
In6_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_Rate_FB
In7_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_TAS
In8_SLStubOut_Rte_IRead_rtwdemo_roll_rtwdemo_roll_Step_Turn_Knob
AP_Eng
HDG_Mode
HDG_Ref
Phi
Psi
Rate_FB
TAS
Turn_Knob
SLStubOut_Rte_IRead_rtwdemo_roll
rtwdemo_roll_Step
Signal spec. and routing
Out1_SLStubIn_Rte_IWrite_rtwdemo_roll_rtwdemo_roll_Step_Ail_Cmd_Ail_Cmd_p1
Ready
View 1 warning
123%
FixedStepDiscrete

Static Code Analysis with Polyspace

- Code metrics and standards
  - Comment density, cyclomatic complexity,…
  - MISRA and Cybersecurity standards
  - Support for DO-178, ISO 26262, ….
- Bug finding and code proving
  - Check data and control flow of software
  - Detect bugs and security vulnerabilities
  - Prove absence of runtime errors

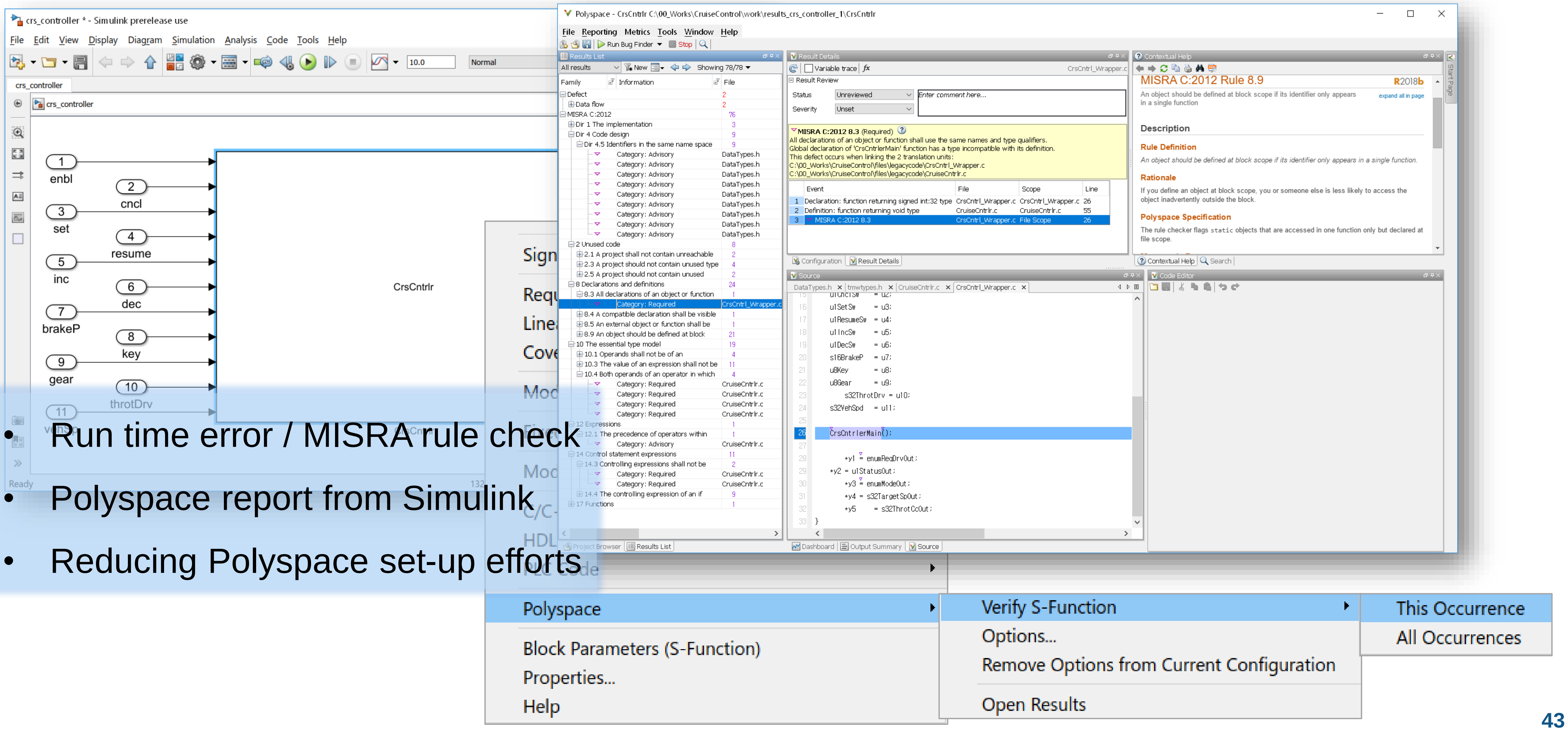


```
static void pointer_arithmetic (void) {
    int array[100];
    int *p = array;
    int i;

    for (i = 0; i < 100; i++) {
        *p = 0;
        p++;
    }

  if (get_bus_status() > 0) {
    if (get_oil_pressure() > 0) {
        *p = 5;
    } else {
        i++;
    }
  }

  i = get_bus_status();

  if (i >= 0) {
    *(p - i) = 10;
  }
}
```

```
variable ‘I’ (int32): [0 .. 99]
assignment of ‘I’ (int32): [1 .. 100]
```

Results from Polyspace Code Prover

Static Code Analysis with Polyspace and Simulink



```
Family                                                   Information              File
Defect                                                                            2
  Data flow                                                                       2
MISRA C:2012                                                                      76
  Dir 1 The implementation                                                        3
  Dir 4 Code design                                                               9
    Dir 4.5 Identifiers in the same name space                                    9
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
                                                         Category: Advisory       DataTypes.h
  2 Unused code                                                                   8
    2.1 A project shall not contain unreachable                                   2
    2.3 A project should not contain unused type                                  4
    2.5 A project should not contain unused                                       2
  8 Declarations and definitions                                                  24
    8.3 All declarations of an object or function                                 1
                                                         Category: Required       CrsCntrl_Wrapper.c
    8.4 A compatible declaration shall be visible                                 1
    8.5 An external object or function shall be                                   1
    8.9 An object should be defined at block                                      21
  10 The essential type model                                                     19
    10.1 Operands shall not be of an                                              4
    10.3 The value of an expression shall not be                                  11
    10.4 Both operands of an operator in which                                    4
                                                         Category: Required       CruiseCntrlr.c
                                                         Category: Required       CruiseCntrlr.c
                                                         Category: Required       CruiseCntrlr.c
                                                         Category: Required       CruiseCntrlr.c
  12 Expressions                                                                  1
    12.1 The precedence of operators within                                       1
                                                         Category: Advisory       CruiseCntrlr.c
  14 Control statement expressions                                                11
    14.3 Controlling expressions shall not be                                     2
                                                         Category: Required       CruiseCntrlr.c
                                                         Category: Required       CruiseCntrlr.c
    14.4 The controlling expression of an if                                      9
  17 Functions                                                                    1
```

Result Details — Variable trace — CrsCntrl_Wrapper.c

Result Review

Status: Unreviewed
Severity: Unset
Enter comment here...

MISRA C:2012 8.3 (Required)
All declarations of an object or function shall use the same names and type qualifiers.
Global declaration of 'CrsCntrlerMain' function has a type incompatible with its definition.
This defect occurs when linking the 2 translation units:
C:\00_Works\CruiseControl\files\legacycode\CrsCntrl_Wrapper.c
C:\00_Works\CruiseControl\files\legacycode\CruiseCntrlr.c

	Event	File	Scope	Line
1	Declaration: function returning signed int:32 type	CrsCntrl_Wrapper.c	CrsCntrl_Wrapper.c	26
2	Definition: function returning void type	CruiseCntrlr.c	CruiseCntrlr.c	55
3	MISRA C:2012 8.3	CrsCntrl_Wrapper.c	File Scope	26

Configuration | Result Details

Contextual Help

MISRA C:2012 Rule 8.9 — R2018b

An object should be defined at block scope if its identifier only appears in a single function — expand all in page

Description

Rule Definition

An object should be defined at block scope if its identifier only appears in a single function.

Rationale

If you define an object at block scope, you or someone else is less likely to access the object inadvertently outside the block.

Polyspace Specification

The rule checker flags `static` objects that are accessed in one function only but declared at file scope.

Source — DataTypes.h | tmwtypes.h | CruiseCntrlr.c | CrsCntrl_Wrapper.c

```
    u1CnclSw     = u2;
    u1SetSw      = u3;
    u1ResumeSw   = u4;
    u1IncSw      = u5;
    u1DecSw      = u6;
    s16BrakeP    = u7;
    u8Key        = u8;
    u8Gear       = u9;
        s32ThrotDrv = u10;
    s32VehSpd    = u11;

    CrsCntrlerMain();

        *y1 = enumReqDrvOut;
    *y2 = u1StatusOut;
        *y3 = enumModeOut;
        *y4 = s32TargetSpOut;
        *y5     = s32ThrotCcOut;
}
```

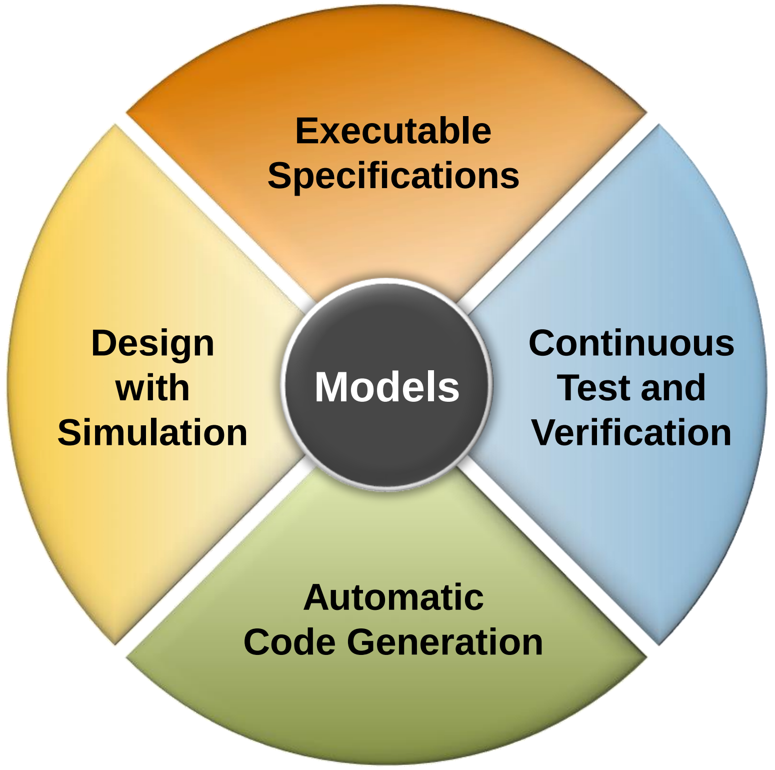
- Run time error / MISRA rule check
- Polyspace report from Simulink
- Reducing Polyspace set-up efforts

Agenda

- How to get started MBD with Legacy Code?
- Legacy Code Integration using Simulink
- Legacy Code Verification
- **Key Takeaways**

Key Takeaways

- **How to get started MBD?**
  - Verify Legacy Code using Simulink
  - Experiment with a Small Piece of the Project
  - Adopt Full MBD to Project
- **Legacy Code Integration**
  - Legacy code tool, C Caller Block, C Function Block, Simulink Code Importer
- **Legacy Code Verification in Simulink**
  - High flexibility for test input
  - Automation of verification workflow
  - Easy test case management and nice visualization


Executable Specifications
Design with Simulation
Models
Continuous Test and Verification
Automatic Code Generation

MATLAB EXPO

Thank you

MathWorks®