Case Study:
Autonomous Vehicle Sensor Module Software Testing

Integration of VectorCAST with Simulink

Speaker: Jeff Lovell
Senior Systems Engineer, Pi Innovo LLC

Pi Innovo™

Abstract

- Pi Innovo was responsible for development of custom hardware and software, for use as a sensor module within an autonomous passenger car.
- This session presents Pi Innovo's activity as a **case-study of modern application software testing**, which included the use of a combined Simulink and VectorCAST environment for software unit testing. This environment enabled re-use of familiar methods and test-scripts, while adding capability to quantify new code-coverage metrics after Simulink autocode generation.

Pi Innovo

Presentation outline

- **System overview**
- Pi Innovo responsibilities
- Pi Innovo development process
- Purpose and usage of VectorCAST
- Results

Pi Innovo

Autonomous passenger car

- This software application is part of a system within an autonomous passenger car.

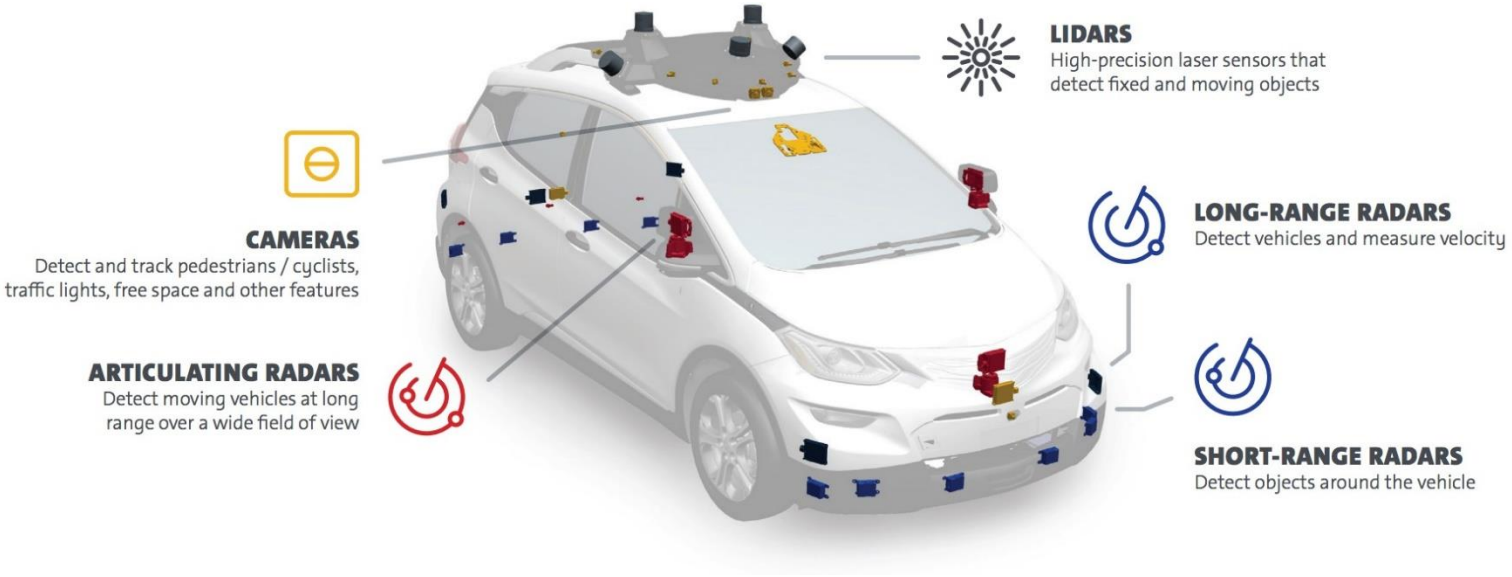
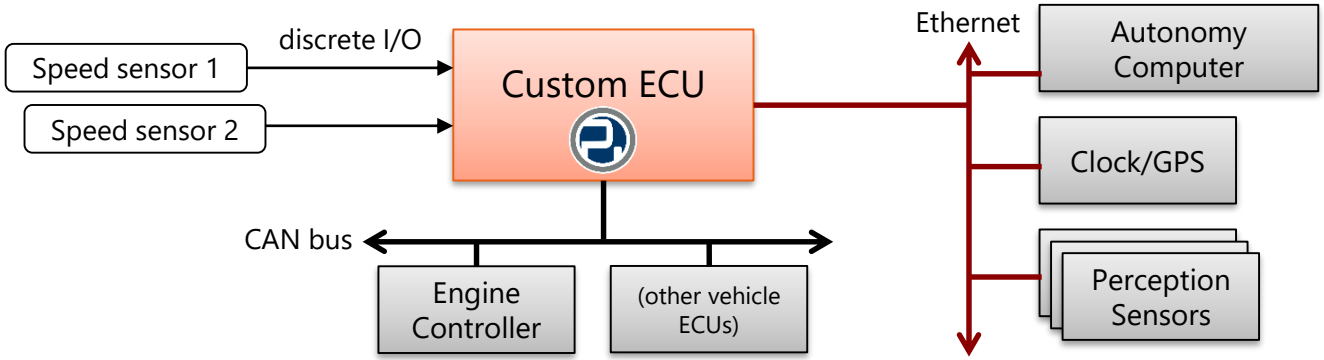


image: CNET.com

High-integrity sensor module

- The ECU being developed is a sensor module, responsible for reporting data to an autonomy computing platform.
- The required behaviors included:
  - Measurement and processing of high-resolution wheel position encoders.
  - Data timestamping based on the vehicle's master clock module.
  - Data transmission using 100BASE-T1 automotive ethernet (2-wire).
  - Fail-operational robustness to single-point failures (internal redundancy).



Pi Innovo

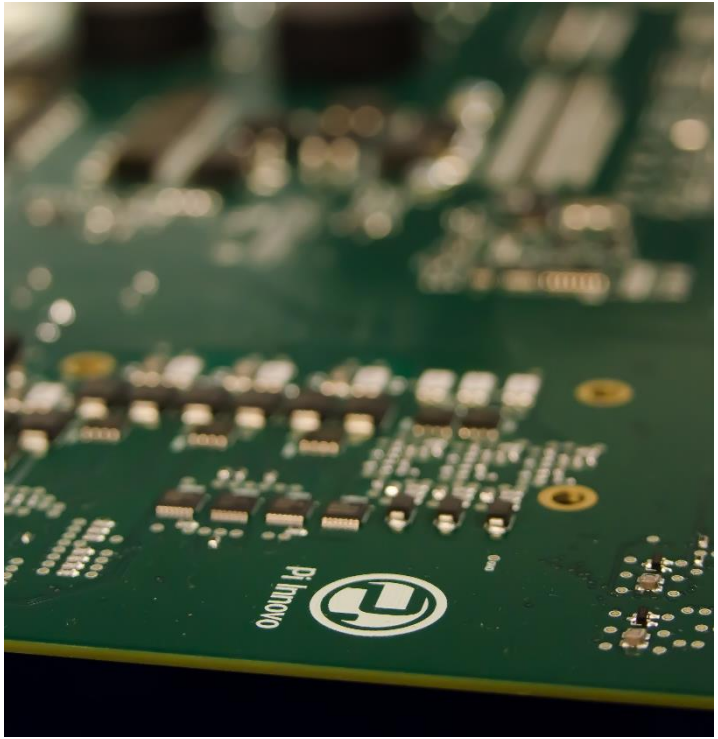
Presentation outline

- System overview
- **Pi Innovo responsibilities**
- Pi Innovo development process
- Purpose and usage of VectorCAST
- Results

Pi Innovo

Pi Innovo responsibilities

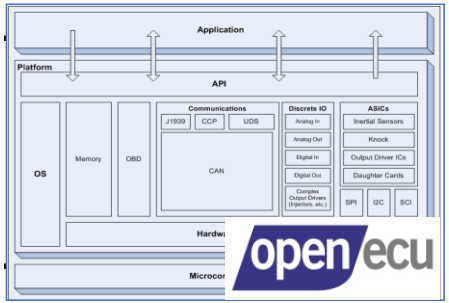
- The OEM assigned Pi Innovo the Tier 1 responsibilities of development and supply of the custom ECU.
- Pi Innovo performs the design and development activities in-house.
- Special tests are outsourced to test houses.
- ECU manufacture is performed on-contract by partner companies.

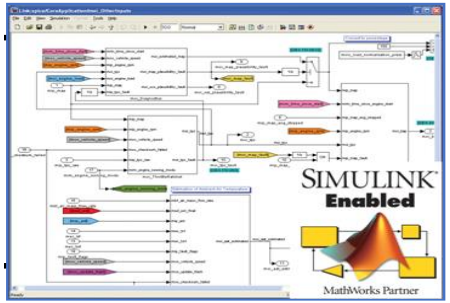


Pi Innovo

Pi Innovo responsibilities

- Custom ECU hardware development
  - New PCB & enclosure
  - DV-PV
- Custom ECU firmware development
  - (New microprocessor type)
  - OS (ported from OpenECU)
  - Hardware drivers
  - Software API
  - V&V
- Custom ECU software development
  - Application logic
  - V&V





Pi Innovo

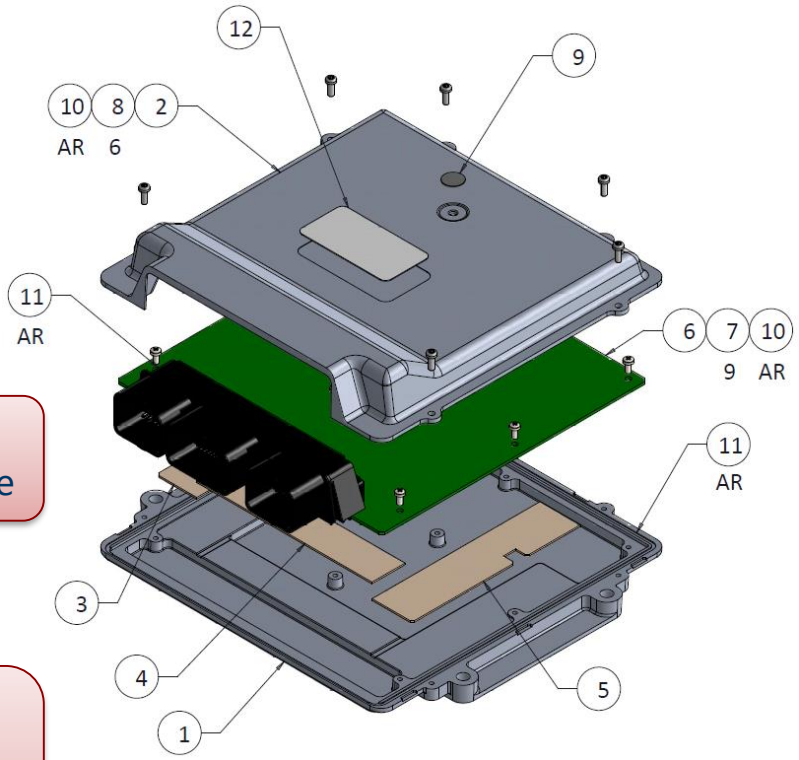
Pi Innovo responsibilities

- Custom ECU hardware development
  - New PCB & enclosure
  - DV-PV
- Custom ECU firmware development
  - (New microprocessor type)
  - OS (ported from OpenECU)
  - Hardware drivers
  - Software API
  - V&V

VectorCAST test cases for C language

- Custom ECU software development
  - Application logic
  - V&V

VectorCAST coverage analysis for Simulink (this presentation)



Pi Innovo

Go-fast timing

- Functional A-samples delivered to customer 11 months after design kickoff (no prior design existed).
- Rapid development requires maximum re-use of existing products and techniques, even for this 'from-scratch' ECU.
  - Modular circuit designs (Pi OpenECU)
  - Hardware development environment
  - Modular software designs (Pi OpenECU)
  - Software development environment
  - **Test tools**
  - QA process (Pi BMS)

Pi Innovo

Presentation outline

- System overview
- Pi Innovo responsibilities
- **Pi Innovo development process**
- Purpose and usage of VectorCAST
- Results

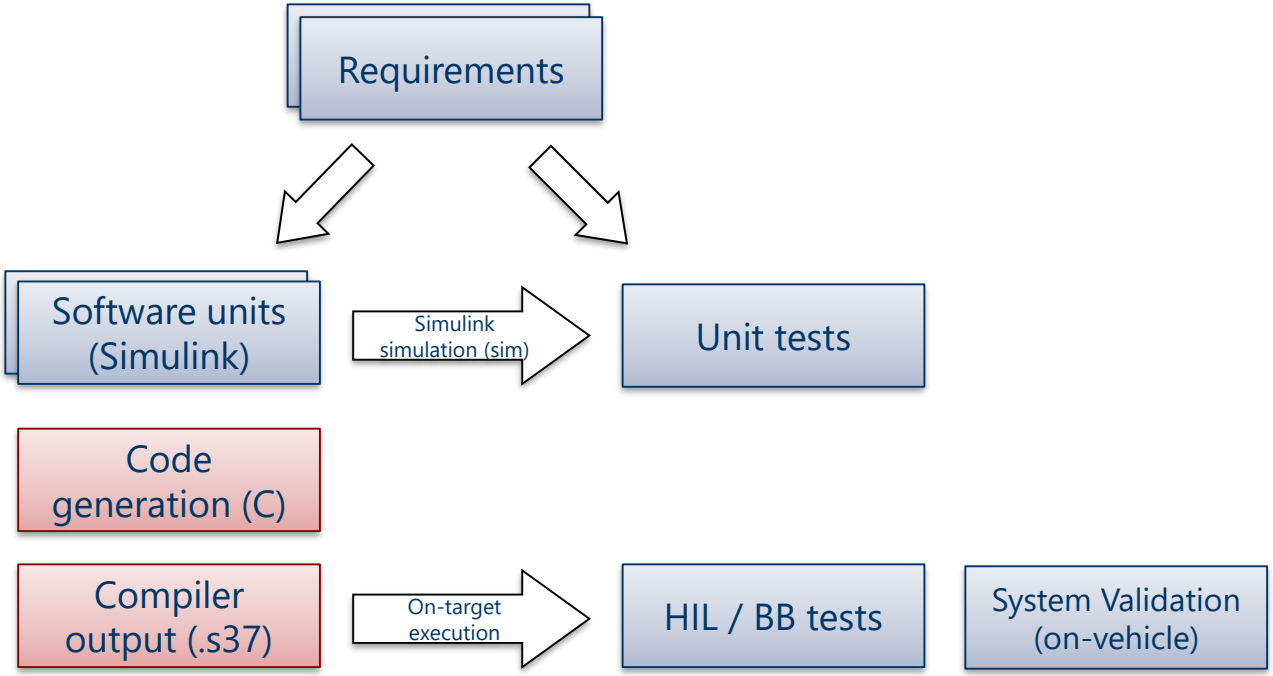
Pi Innovo

Product quality

Pi Innovo's Business Management System (BMS) is based around the requirements of ISO9001:2015 and ISO15504 Automotive SPICE and provides a full suite of operational and project-level processes serving the Organization as a whole, Quality Assurance, Project/Program Management and Engineering departments.

Pi Innovo

Model-based application software development (traditional vehicle systems)



Presentation outline

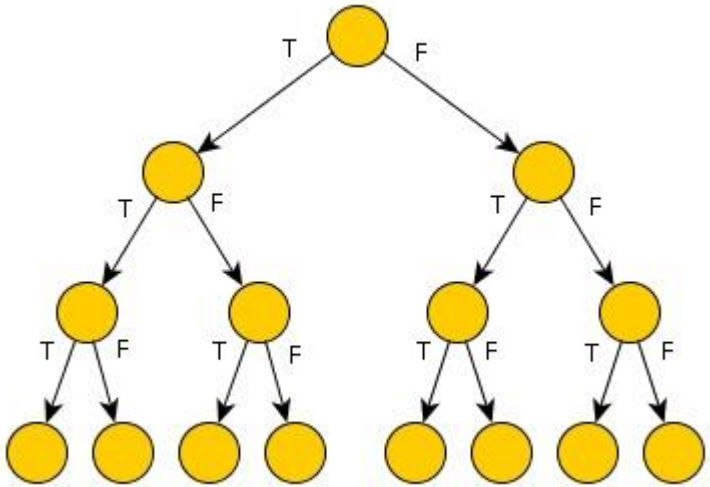
- System overview
- Pi Innovo responsibilities
- Pi Innovo development process
- **Purpose and usage of VectorCAST**
- Results

Pi Innovo

Additional V&V required for high-integrity ECU software

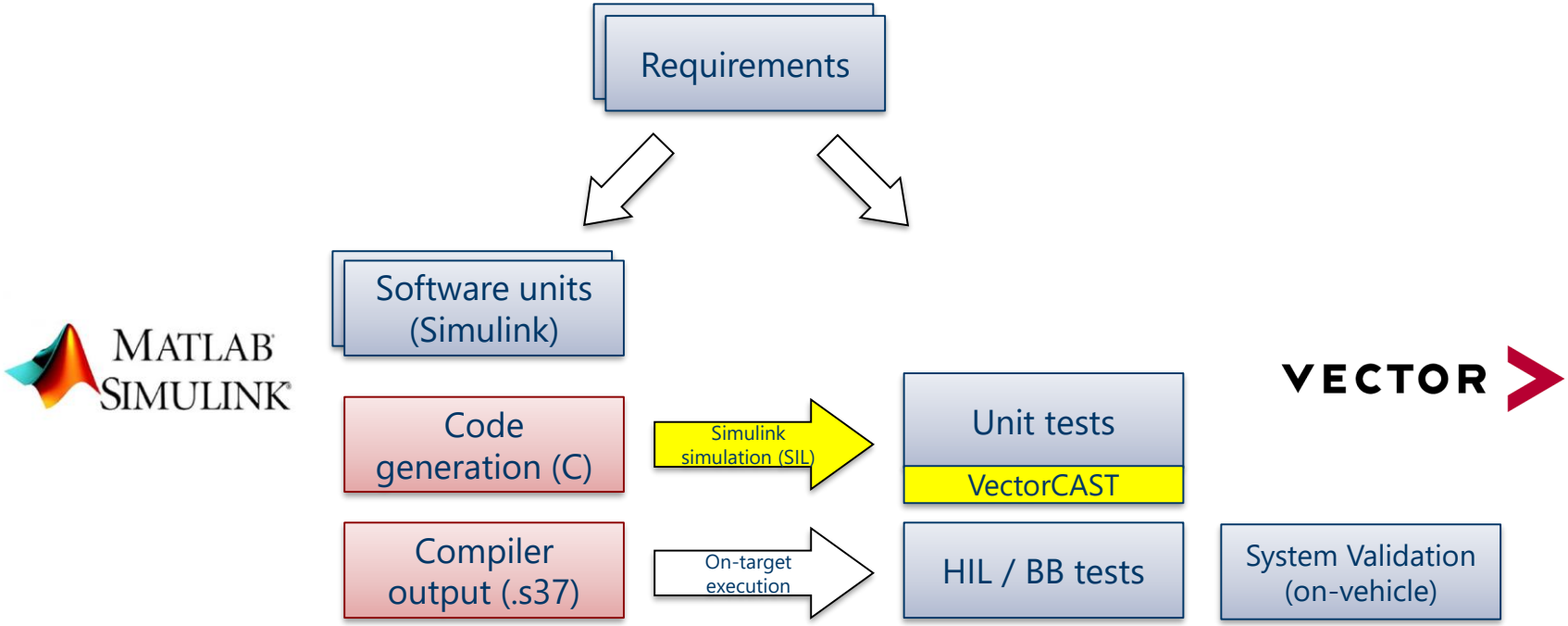
- This product required formal measurements of code coverage during software testing.
  - Branch coverage
  - Statement coverage
  - Modified condition / decision coverage (MCDC)

- **GAP:** these concepts apply to the C-language source; not our model-based source.
  - We therefore require a method to **monitor code coverage of the autogenerated C** program, during our unit tests.



Pi Innovo

Model-based application software development (high-integrity applications)



Pi Innovo

Model-based application software development (Solution)

Configured Simulink simulation to SIL-mode (generates C and compiles for execution on x86 PC)

Unit tests (scripts) for Simulink simulation do not change. (Pi Innovo scripting environment using Excel & Simulink)

VectorCAST C++ (for C) instruments the auto-generated C, and records coverage metrics while monitoring its execution.

Software units (Simulink)

Code generation (C)

Compiler output (.s37)

Simulink simulation (SIL)

On-target execution

Unit tests

VectorCAST

HIL / BB tests

System Validation (on-vehicle)

Tool qualification

- VectorCAST is a software tool the industry recognizes from prior experience, and approves for high-integrity development.
- (Tool is ‘pre-approved’ by major OEM’s - saves project time vs new tool cert.)

Pi Innovo

Presentation outline

- System overview
- Pi Innovo responsibilities
- Pi Innovo development process
- Purpose and usage of VectorCAST
- **Results**

Pi Innovo

Results example

- Simulink controls the software execution, VectorCAST monitors the coverage.
- Coverage reports (shown here) are saved as evidence of the test result.

Aggregate Coverage Report

Configuration Data

Environment Name:	vcast
Date of Report Creation:	9 JAN 2018
Time of Report Creation:	11:49:38 AM

Aggregate Coverage

Code Coverage for Unit: IMR_FeatureLogic.c

```
-- Coverage Type: Statement+MC/DC
-- Unit: IMR_FeatureLogic.c
-- Test Case: Aggregate
3 218.1 (T)(F)      fabs(rtb_Add_oo - rtb_Add_kp) > 1.0) &&
                      ((!(
3 218.2 (T)(F)      (rtu_IMRBus_g & localC->FixPtBitwiseOperator1_le) > 0)) &&
3 218.3 (T)(F)      rtb_RelationalOperator2));
                    /* Switch: '<S106>/Switch' incorporates:
                     *  Constant: '<S106>/Constant'
                     *  DataTypeConversion: '<S106>/Data Type Conversion'
                     *  Logic: '<S102>/Logical Operator'
                     *  Sum: '<S106>/Sum1'
                     *  UnitDelay: '<S106>/Unit Delay1'
                     */
3 219   (T)(F)      if (!
3 219.1 (T)(F)      rtb_LogicalOperator4_h) {
3 220    *            rtb_NewValue_pd_idx_0 = localDW->UnitDelay1_DSTATE_1[0] +
                        0.0020000000000074536;
                    } else {
3 221    *            rtb_NewValue_pd_idx_0 = 0.0;
                    }
3 222   (T)(F)      if (!
3 222.1 (T)(F)      rtb_LogicalOperator5_i) {
3 223    *            rtb_NewValue_pd_idx_1 = localDW->UnitDelay1_DSTATE_1[1] +
                        0.0020000000000074536;
                    } else {
3 224    *            rtb_NewValue_pd_idx_1 = 0.0;
                    }
3 225   (T)(F)      if (!
3 225.1 (T)(F)      rtb_RelationalOperator2) {
3 226    *            rtb_NewValue_pd_idx_2 = localDW->UnitDelay1_DSTATE_1[2] +
                        0.0020000000000074536;
                    } else {
3 227    *            rtb_NewValue_pd_idx_2 = 0.0;
                    }
```

Unit: IMR_FeatureLogic.c
Subprogram: IMR_Main
Condition: # 432

```
Source line: 2373
Actual Expression is: localDW->IMR_Logic_MODE
Condition "a" (Ca) is: localDW->IMR_Logic_MODE
Simplified Expression is: a
|----+----+------+----|
|Row |Ca  |Rslt  |Pa  |
|----+----+------+----|
|*1  |T   |T     |2   |
|----+----+------+----|
|*2  |F   |F     |1   |
|----+----+------+----|
Pa => a pair was satisfied (1/2)
      1/2
Pairs satisfied: 1 of 1 ( 100% )
```

Metrics

Unit	Subprogram	Complexity	Statements	Branches	Pairs
IMR_FeatureLogic.c	IMR_FBM	10	45 / 45 (100%)	178 / 181 (98%)	69 / 72 (95%)
	IMR_Main_Start	1	79 / 79 (100%)	1 / 1 (100%)	
	IMR_Main	78	433 / 433 (100%)	753 / 753 (100%)	226 / 226 (100%)
TOTALS	3	89	556 / 556 (100%)	932 / 935 (99%)	295 / 298 (98%)
GRAND TOTALS	**3**	**89**	**556 / 556 (100%)**	**932 / 935 (99%)**	**295 / 298 (98%)**

Pi Innovo

VectorCAST/Cover 6.4 - C:\SVN_working\Anchor\Development\Branches\SW_Features\MDE\VectorCAST\vcast.vcp

File Edit View Environment Project Coverage Static Analysis Tools Window Help

Environment View

Filename	S	B	P
MDE_FeatureLogic.c	✓	✓	✓
MDE_WrapperModel.c			
MDE_WrapperModel_capi.c			
rt_nonfinite.c			

Name

- Coverage Data
  - Covered By Analysis
    - CBA_MDE_FeatureLogic
  - Test Results
    - Results
    - Results - #2
    - Results - #3
    - Results - #4
    - Results - #5
    - Results - #6
    - Results - #7
    - Results.Aug212018163523973

Messages

Message | Error

Python Module file_hooks Reloaded
Manage Project is Open
Obtaining COVER License...
Obtained COVER License Successfully
Processing options file C:\SVN_working\Dolly\Development\Trunk
Coverage Environment is Opened
Processing options file C:\SVN_working\Anchor\Development\Bra
Coverage Environment is Opened
Automatically naming result file: C:/SVN_working/Anchor/Develo
Adding result: CBA_MDE_FeatureLogic - #2
Opening test case window
Opening test case window

Coverage | Coverage Analysis Editor

Coverage results for MDE_FeatureLogic.c

	Subp	Cond	(T)	(F)	Statements 100%	Branches 100%	Pairs 100%
90	3	1.2	(T)				
94	3	2	*				Test_PWORK.DataP
95	3	3	*				Test_PWORK.TimeP
96	3	4	*				.PrevIndex;
97	3	5	*		real_T t = ((MDE_WrapperModel_M->Timing.clockTick0) * 0.01);		
					/* Get index */		
100	3	6	(T)	(F)	if (		
100	3	6.1	(T)	(F)	t <= pTimeValues[0]) {		
101	3	7	*		currTimeIndex = 0;		
					} else		
102	3	8	(T)	(F)	if (		
102	3	8.1	(T)	(F)	t >= pTimeValues[9]) {		
103	3	9	*		currTimeIndex = 8;		
					} else {		
105	3	10	(A)	(F)	if (		
105	3	10.1	(A)	(F)	t < pTimeValues[currTimeIndex]) {		
106	3	11	(A)	(A)	while (		
106	3	11.1	(A)	(A)	t < pTimeValues[currTimeIndex]) {		
107	3	12	A		currTimeIndex--;		
					}		
					} else {		
110	3	13	(T)	(F)	while (		
110	3	13.1	(T)	(F)	t >= pTimeValues[currTimeIndex + 1]) {		
111	3	14	*		currTimeIndex++;		
					}		
					}		
					}		
116	3	15	*		localDW->SIL_Test_IWORK.PrevIndex = currTimeIndex;		
					/* Post output */		
					{		
120	3	16	*		real_T t1 = pTimeValues[currTimeIndex];		
121	3	17	*		real_T t2 = pTimeValues[currTimeIndex + 1];		
122	3	18	(A)	(F)	if (		
122	3	18.1	(A)	(F)	t1 == t2) {		
123	3	19	(A)	(A)	if (		
123	3	19.1	(A)	(A)	t < t1) {		
124	3	20	A		rtb_SIL_Test = pDataValues[currTimeIndex];		

Autogenerated C
(from Simulink)

Jobs

Instrumented files: 13/13 (100%) ...king/Anchor/Development/Branches/SW_Features/MDE/VectorCAST/

Pi Innovo

VectorCAST/Cover 6.4 - C:\SVN_working\Anchor\Development\Branches\SW_Features\MDE\VectorCAST\vcast.vcp

File Edit View Environment Project Coverage Static Analysis Tools Window Help

Environment View

Filename | S | B | P

- MDE_FeatureLogic.c
- MDE_WrapperModel.c
- MDE_WrapperModel_capi.c

Results - #2

Results - #3

Results - #4

Coverage | Coverage Analysis Editor

Coverage results for MDE_FeatureLogic.c

	Subp	Cond	(T)	(F)	Statements 100%	Branches 100%	Pairs 100%
90	3	1.2	(T)	(F)	rtu_BUS_Inp == 0));		
					/* FromWorkspace: '<S56>/SIL_Test' */		
					{		
94	3	2	*		real_T *pDataValues = (real_T *) localDW->SIL_Test_PWORK.DataP		
95	3	3	*		real_T *pTimeValues = (real_T *) localDW->SIL_Test_PWORK.TimeP		
96	3	4	*		int_T currTimeIndex = localDW->SIL_Test_IWORK.PrevIndex;		
97	3	5	*		real_T t = ((MDE_WrapperModel_M->Timing.clockTick0) * 0.01);		
					/* Get index */		
100	3	6	(T)	(F)	if (		
100	3	6.1	(T)	(F)	t <= pTimeValues[0]) {		
101	3	7	*		currTimeIndex = 0;		
					} else		
102	3	8	(T)	(F)	if (		
102	3	8.1	(T)	(F)	t >= pTimeValues[9]) {		
103	3	9	*		currTimeIndex = 8;		
					} else {		
105	3	10	(A)	(F)	if (		
105	3	10.1	(A)	(F)	t < pTimeValues[currTimeIndex]) {		
106	3	11	(A)	(A)	while (		
			(A)	(A)	t < pTimeValues[currTimeIndex]) {		
	3	12	A		currTimeIndex--;		
					}		
					} else {		
110	3	13	(T)	(F)	while (		
110	3	13.1	(T)	(F)	t >= pTimeValues[currTimeIndex + 1]) {		
111	3	14	*		currTimeIndex++;		
					}		
					}		
					}		
116	3	15	*		localDW->SIL_Test_IWORK.PrevIndex = currTimeIndex;		
					/* Post output */		
					{		
120	3	16	*		real_T t1 = pTimeValues[currTimeIndex];		
121	3	17	*		real_T t2 = pTimeValues[currTimeIndex + 1];		
122	3	18	(A)	(F)	if (		
122	3	18.1	(A)	(F)	t1 == t2) {		
123	3	19	(A)	(A)	if (		
123	3	19.1	(A)	(A)	t < t1) {		
124	3	20	A		rtb_SIL_Test = pDataValues[currTimeIndex];		

Obtaining COVER License...
Obtained COVER License Successfully
Processing options file C:\SVN_working\Dolly\Development\Trunk
Coverage Environment is Opened
Processing options file C:\SVN_working\Anchor\Development\Bra
Coverage Environment is Opened
Automatically naming result file: C:/SVN_working/Anchor/Develo
Adding result: CBA_MDE_FeatureLogic - #2
Opening test case window
Opening test case window

Jobs

Instrumented files: 13/13 (100%) ...king/Anchor/Development/Branches/SW_Features/MDE/VectorCAST/

Lines of code are reported as covered by the test…

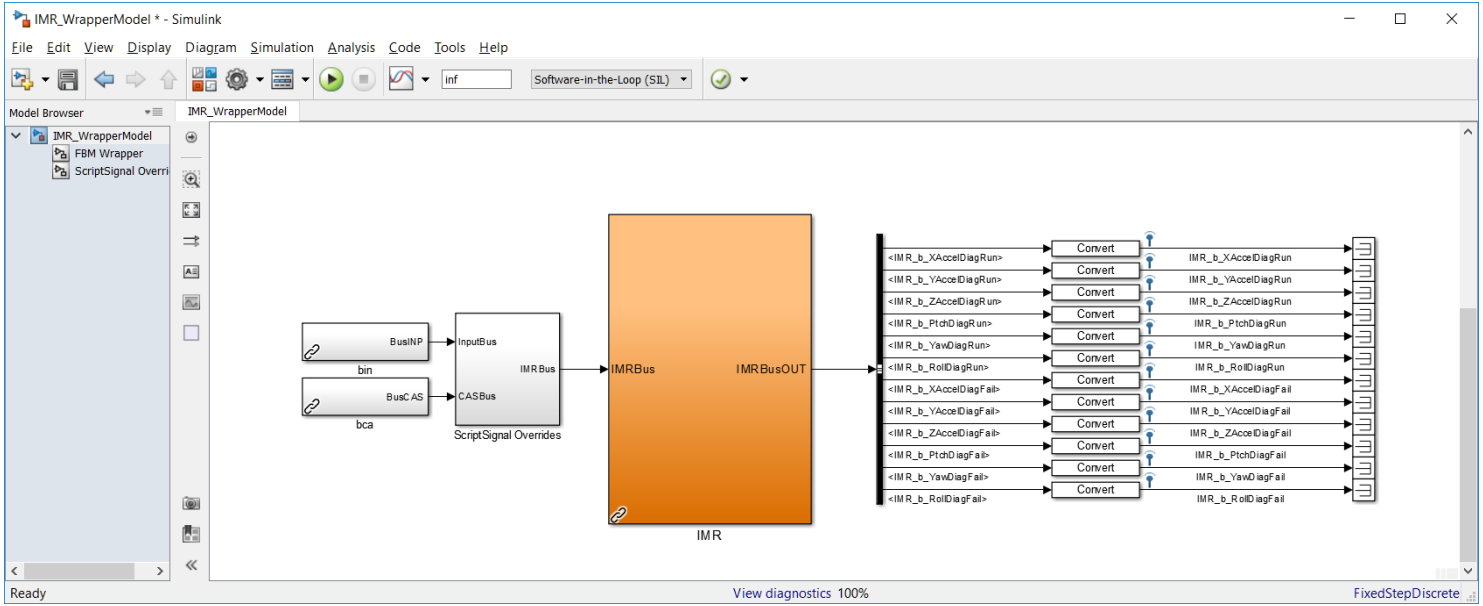
…or by human analysis.

(Necessary when impossible cases are present in autogenerated C)

Pi Innovo

Impact

- This approach enabled re-use of a familiar testing method (Simulink simulation), while adding the required code coverage measurements.
- Re-use of test scripts and simulation wrappers created substantial time-savings, and enabled accelerated delivery of product to the customer.



Pi Innovo

(end)

Jeff Lovell

Senior Systems Engineer, Pi Innovo LLC

Jeff.Lovell@Pi-Innovo.com

Pi Innovo LLC
47023 W. Five Mile Road
Plymouth
MI 48170-3765
United States of America

+1 734 656 0140

pi-innovo.com

Pi Innovo is an expert partner for the design and development of innovative electronics systems to the automotive, transportation, defense, industrial, and aviation industries. Our uniquely adaptable business engagements, based on Pi Team services and OpenECU products, enjoy a strong reputation for delivering results of the highest quality, providing outstanding value for our customers.

OpenECU is a wide range of adaptable, field-ready products and intellectual property designed to accelerate electronics system development. The philosophy behind OpenECU is the creation of modular, reusable technology that is implemented to volume production standards and is fully “open” to custom configuration and further development.