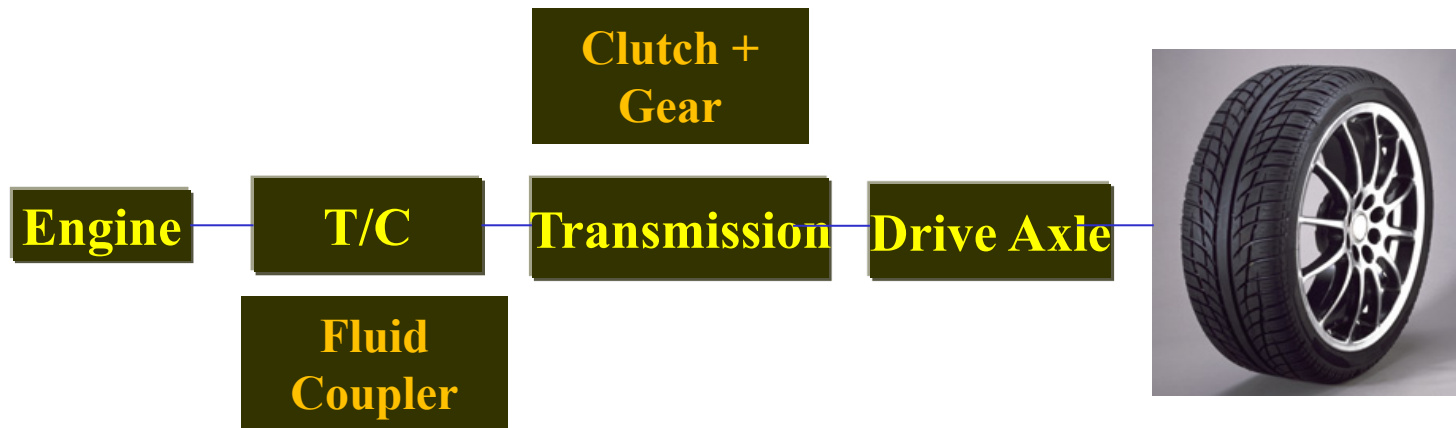


POWERTRAIN CONTROL FEATURE DEVELOPMENT THROUGH MODEL BASED DESIGN

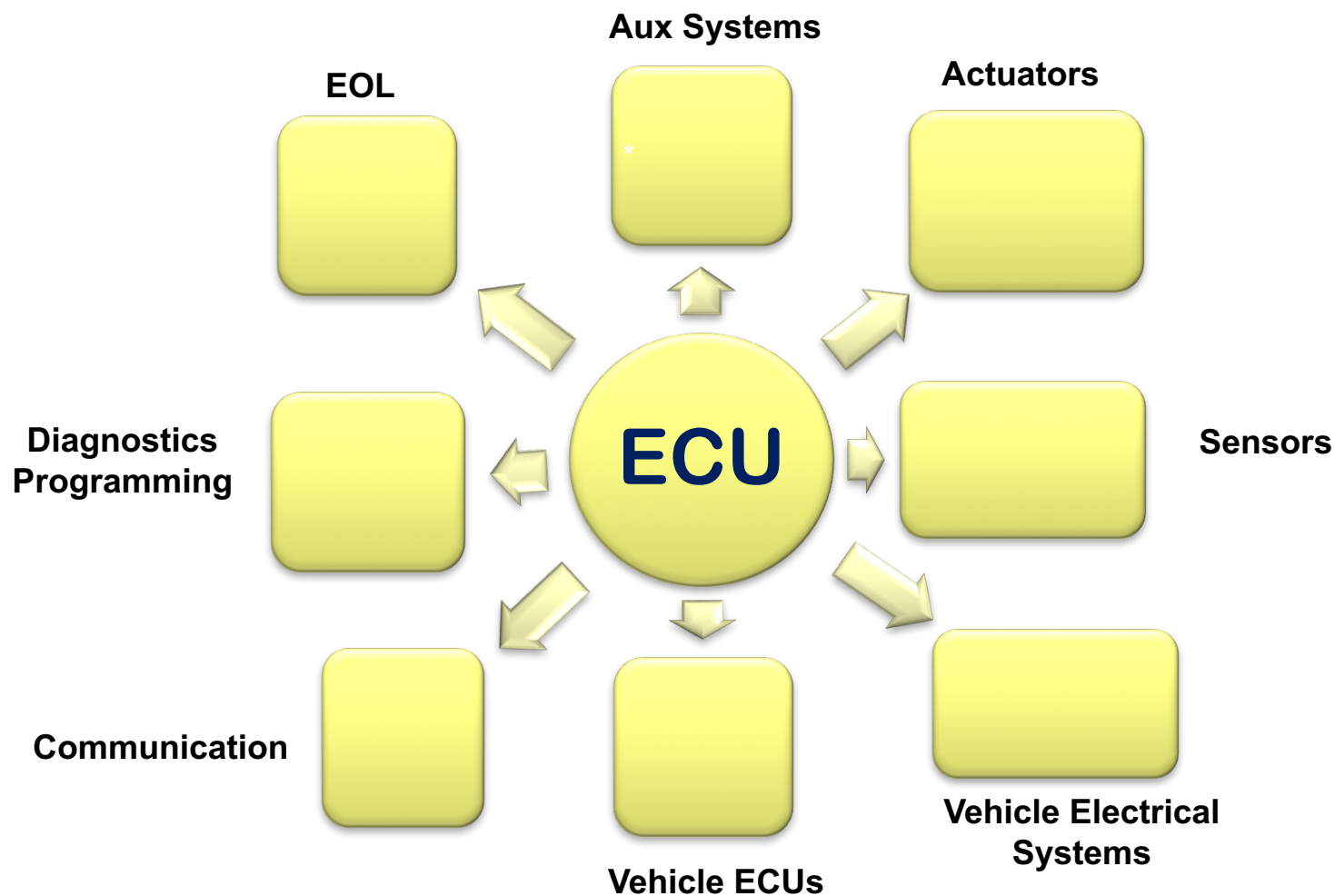
- **Introduction to Powertrain**
- **Powertrain feature – Clutch Control**
- **Requirements**
- **Function Development**
- **Model verification**
- **Design verification & Coverage analysis**
- **Software in loop checking**
- **Auto code generation**
- **Concluding points**

Outline

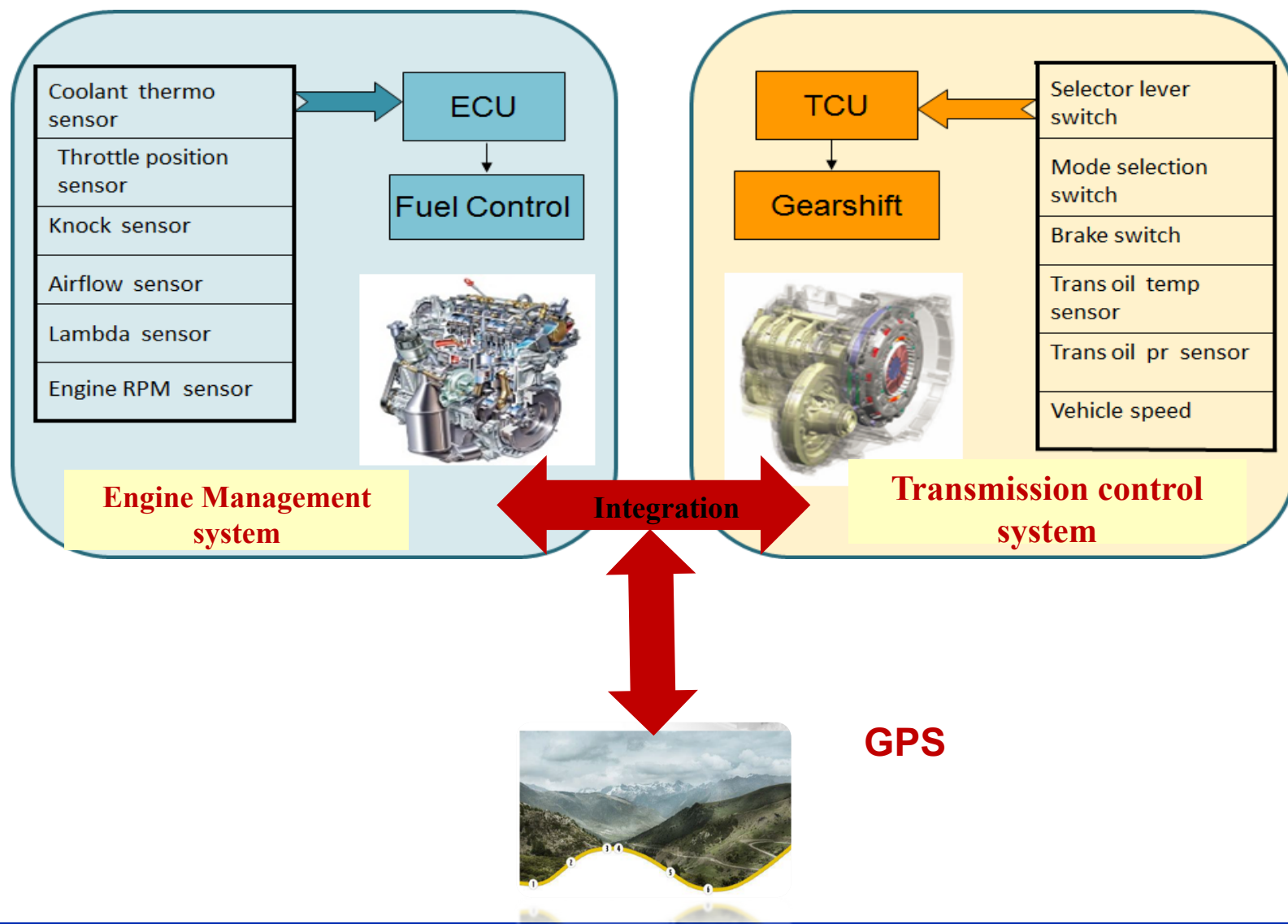
- This presentation describes how Model Based Design (MBD) concept is successfully deployed to address problem and challenges associated in designing complex powertrain control system



Engine Electronics Interfaces



Integrated Powertrain Control



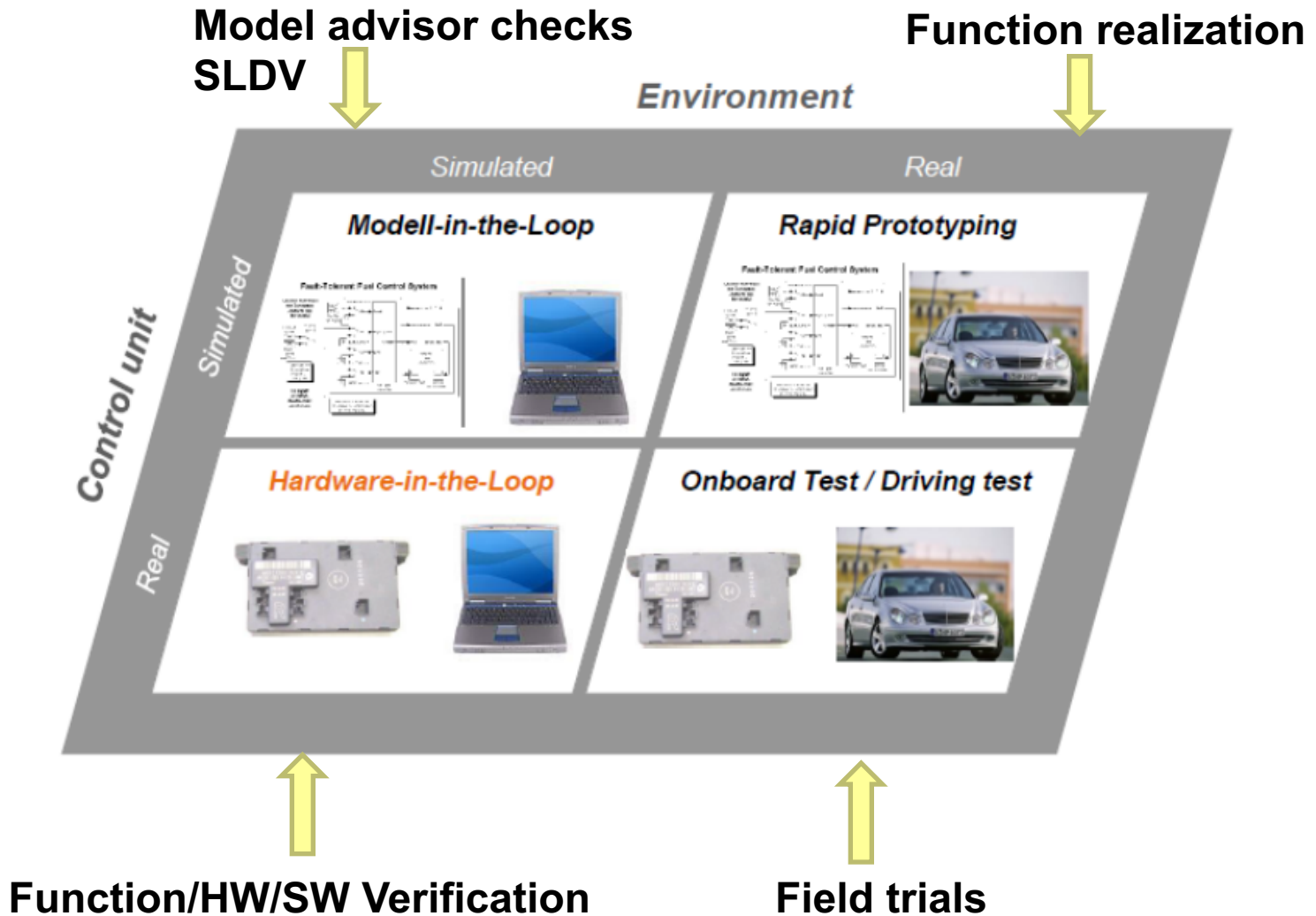
Control Design Challenges

- Powertrain requires to deliver best - performance, emission, Fuel economy, drivability, safety etc.
- Capable of handling **Multi-domain tasking**
- Respond to **tight coupling** of powertrain components
- Manage lot of **interdependencies** and exchange of huge no. of parameters
- Ensure reliable working under all **operating scenarios**

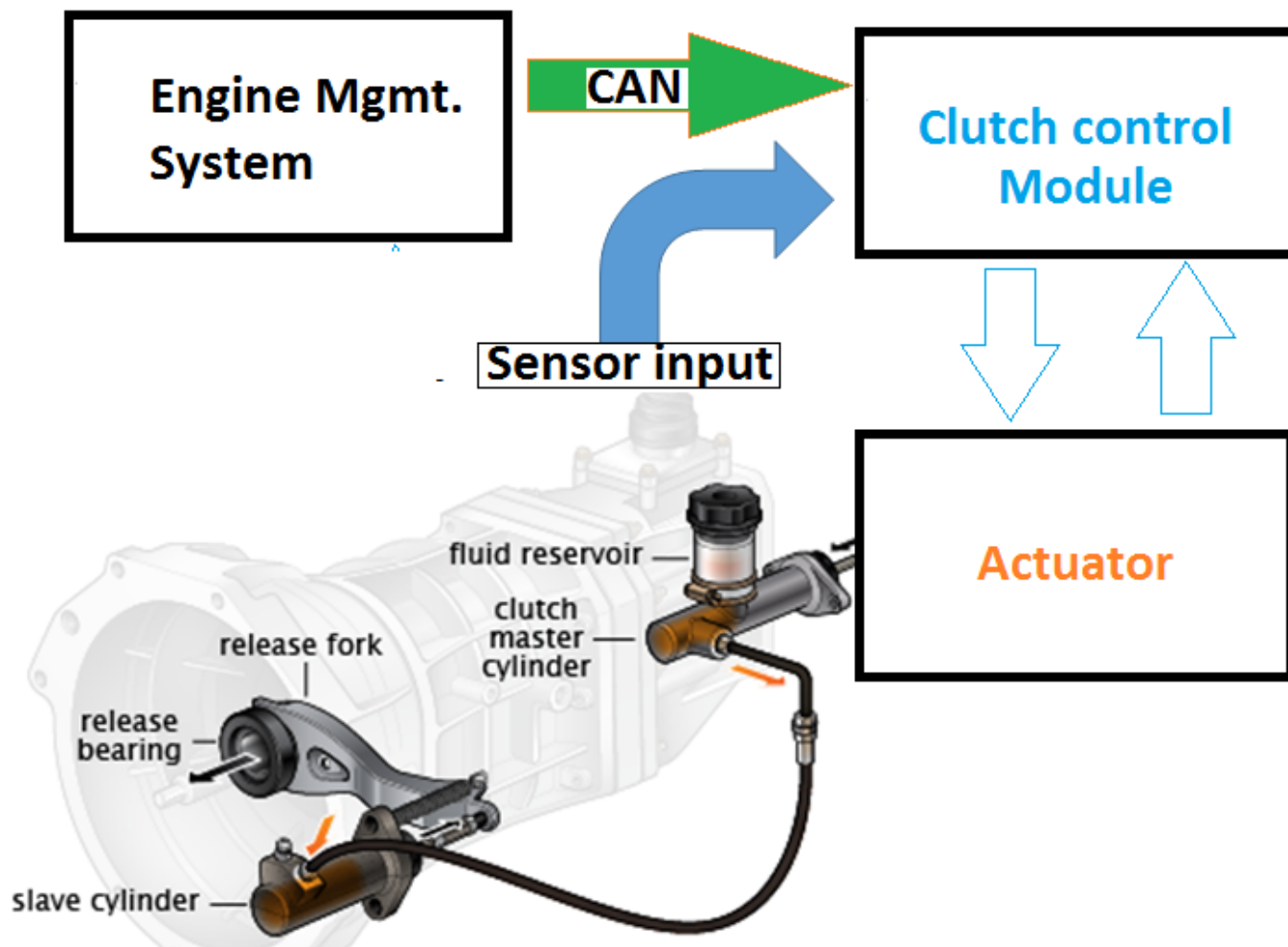
Approach

- Conventional design approach has limitations in terms of analysis, testing, risk mitigation & confidence building
- MBD provides platform for quick building of control design and verification
- Support step by step design integrated with testing throughout the development cycle
- Provide ease of modification to refine algorithm to build optimum system
- Tools used
 - Matlab, Simulink, Stateflow
 - Embedded Coder
 - Simulink Design Verifier
 - Model Advisor

MBDS & Test technologies



Powertrain Control feature-Clutch Control



Key Challenges

- **Auto Clutch control is combination of Manual and Automated actions**
- **Critical success factors**
 - To be in sync & respond near real time basis
 - Clutch actuation trigger with driver input
 - Require continuous motoring of clutch movement & act accordingly
 - Fault management & safety
 - Any mismatch in timing may cause component damage and affect drivability
 - **Need to incorporate self learning & neural logic to build right control mechanism**

Requirements

- Stateflow and Simulink are used for gap analysis
- Multiple iterations of review and discussions were performed
- Requirements were in the form of text / diagrams
- Referencing of interfacing inputs and feedbacks
- Some High level requirement examples –
 - While changing gear clutch should be disengaged
 - While Brake pedal pressed for time-T, then Clutch Should be Disengaged
 - User can select reverse gear ONLY when vehicle speed is zero
 - Current gear & next gear should be identified and clutch shift timings to be varied accordingly

Requirement Reviews & Discussions

- While change in gear Clutch should be disengaged
- While Brake pedal press for T time then Clutch Should be Disengaged
- User can select reverse gear ONLY when vehicle speed is zero

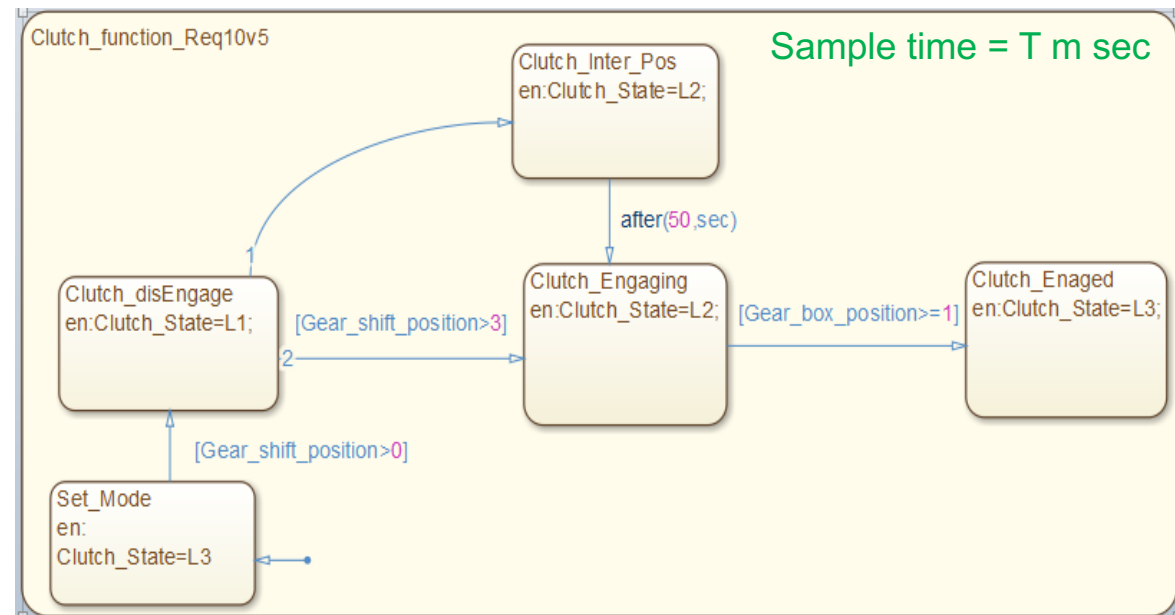
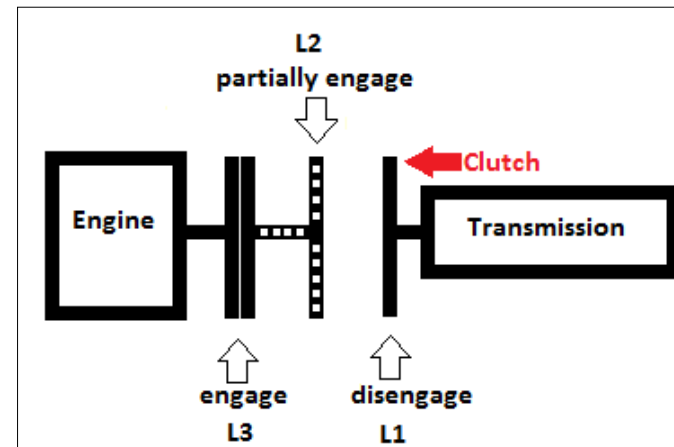
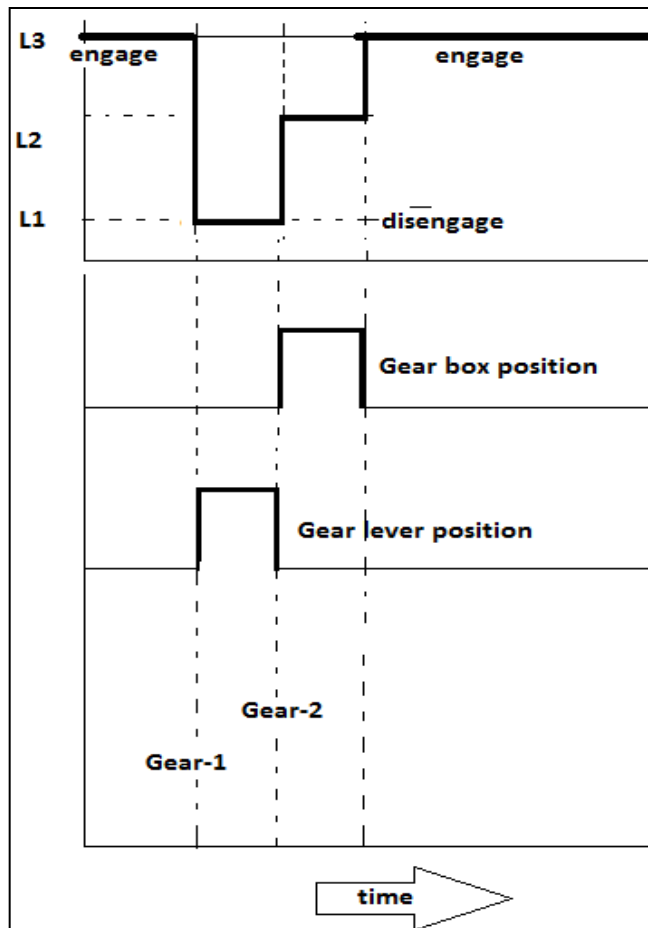


Initial requirements in text

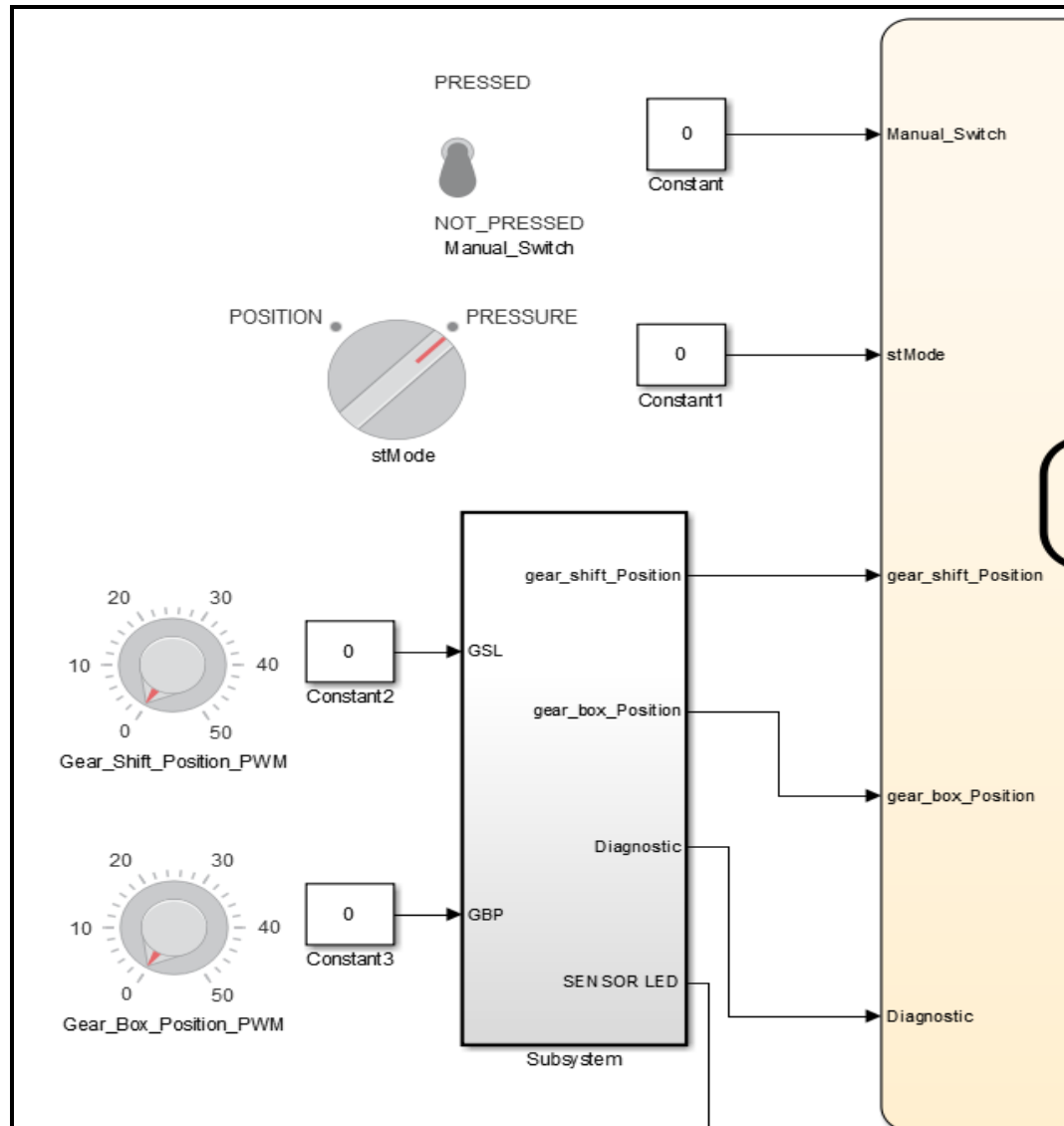
-
- Added tuning parameters(timing, calibration parameter)
 - While modeling missing parameters, relationships, interfaces were identified and corrected.
 - Stateflow enabled to define transition, conditions and actions in the control logic
-

Modeling & Requirement refinements

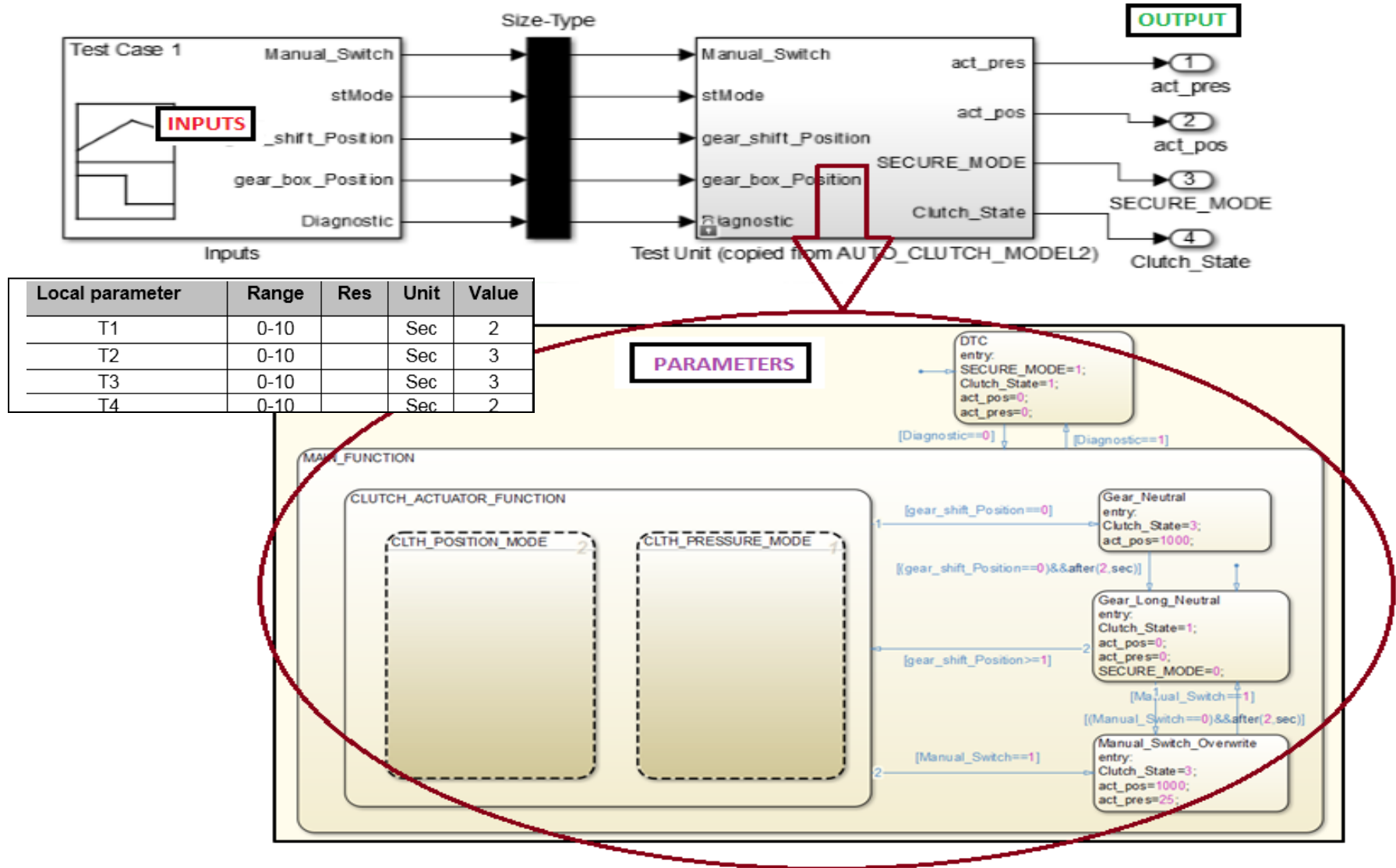
Derived Control Logic



Inputs simulation



Function development

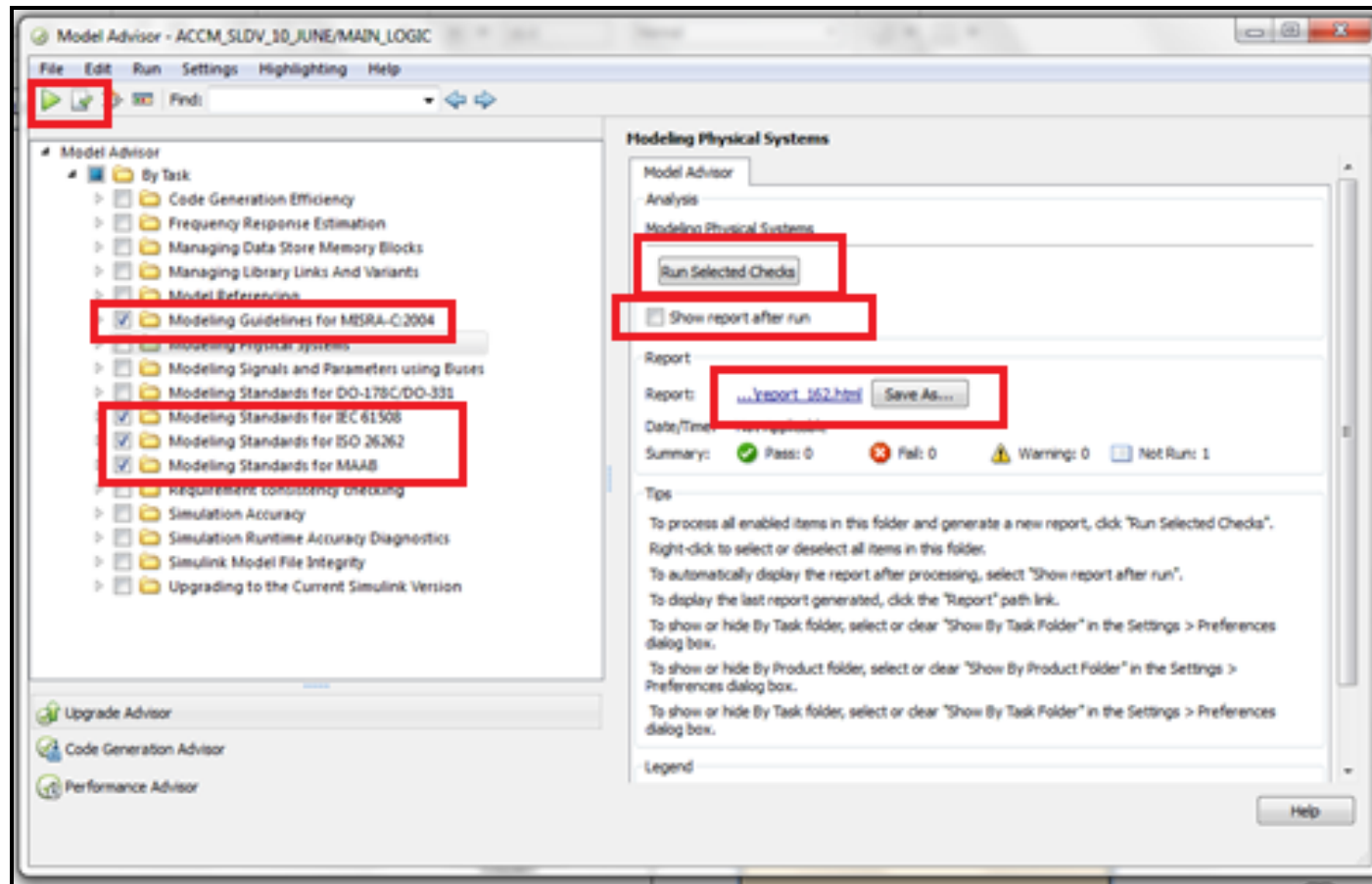


Design Verification

- Requires to check following
 - Individual function
 - Possible real time failure
 - Impact of calibration limits
 - Diagnostics
 - Safety functions
- Following tools were used as part of design verification:
 - Model Advisor
 - Simulink Design Verification
 - Software in Loop

Model Analysis

- Following standard guideline checks were performed:
 - MATHWORKS Automotive Advisor Board
 - MISRA 2004
 - ISO 26262
 - IEC 61582



Model Advisor-Report

Model Advisor Report - Clutch_Eng_DEngage_1_dec_2015.mdl

Simulink version: 8.5

Model version: 1.96

System: Clutch_Eng_DEngage_1_dec_2015/AUTO_CLUTCH MODEL

Current run: 12-Dec-2015 11:14:29

Run Summary

Pass	Fail	Warning	Not Run	Total
 125	 2	 42	 47	216

Check for blocks not recommended for MISRA-C:2004 compliance

Identify blocks that are not supported or recommended for MISRA-C:2004 compliant code generation.

Passed

Blocks that are not supported or recommended for MISRA-C:2004 compliant code generation were not found in subsystem.

Check for Lookup Table blocks using cubic spline interpolation or extrapolation methods.

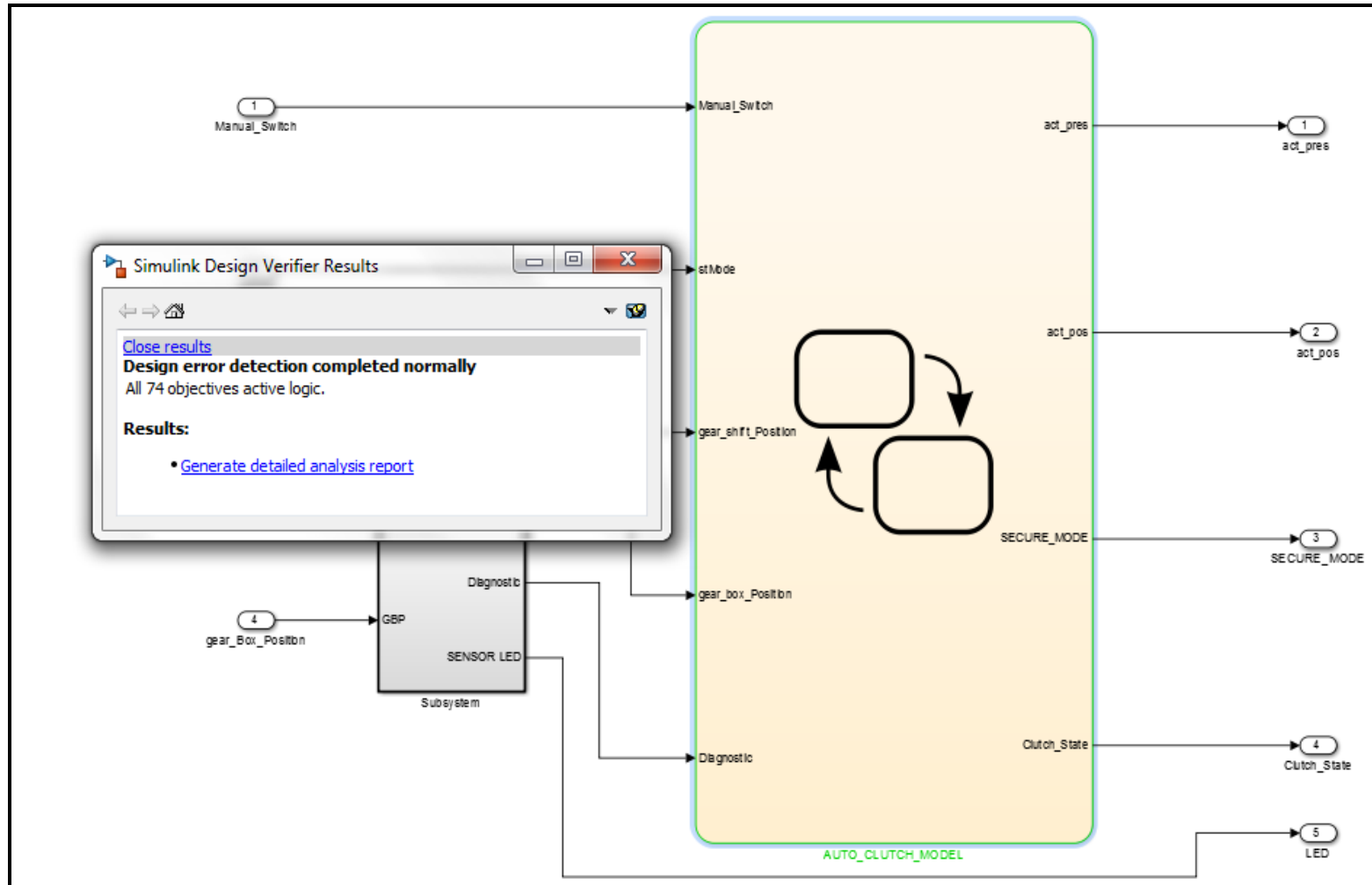
Passed

No Lookup Table blocks using cubic spline interpolation or extrapolation methods found.

Warnings are Corrected after analysis(e.g.)

- Identify signal labels that are not correct for C variable names.
- Check Simulink block or Stateflow objects that do not link to a requirement documents
- Identify mismatches between names of Stateflow ports and the associated signals.

Design verification



SLDV - Results

- Dead Logic Detection

Objectives Status	
Number of Objectives:	98
Dead Logic:	11
Active Logic:	87

- Division by zero

Objectives Status	
Number of Objectives:	0

- Integer Overflow

Objectives Status	
Number of Objectives:	2
Objectives Proven Valid:	1
Objectives Undecided when the Analysis was Stopped:	1

- Check Specified Intermediate Minimum and Maximum Values

Objectives Status	
Number of Objectives:	0

- Out of bound Array

Objectives Status	
Number of Objectives:	0

Model Coverage Analysis

- Generated Input data

Generated Input Data

Time	0-1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Step	1-2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	
Manual_Switch	0	0	1	1	0	1	1	1	1	2	1	0	0	2	1	0	2	0	1	0	2	0	1	0	0	0	1	0	0	2	2	0	0	0.5	2	1	0	0	0.5	2	1	0	0	0.5	2	1	0	0
stMode	0	0	0	0	0	0	0	0	0	0	1	0	0	0.5	0	0	0	0	0	0	0.5	1	1	0	0.5	0	0	0	0.5	0	0	0	0	1	0	0	0	0.5	0	0	1	0	0.5	0	0	0	0	
gear_shift_Position	0	0	3	3	0	0	3	3	3	1	1	3	1	-1	3	3	0	3	3	3	2	3	3	3	3	1	1	3	3	0	1	0	1	3	2	1	3	3	3	2	1	3	3	3	3	3	3	3
gear_box_Position	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
Diagnostic	0	1	1	0	2	0	0	1	0	2	0	0	0	1	0	0	2	1	0	0	2	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Coverage Report

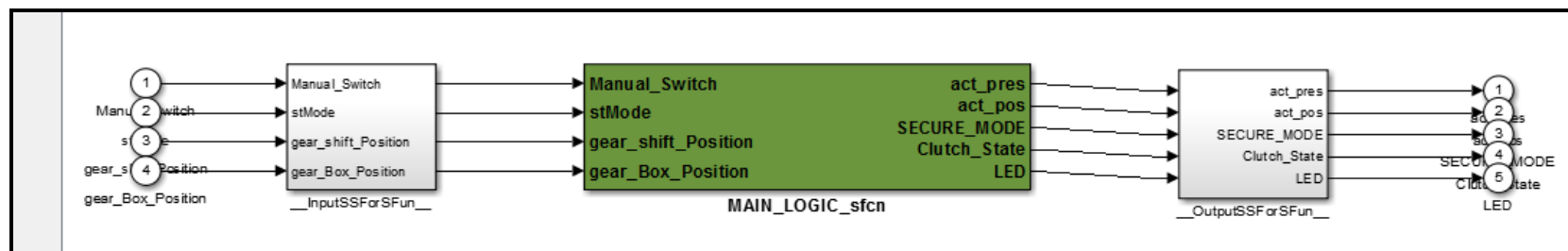
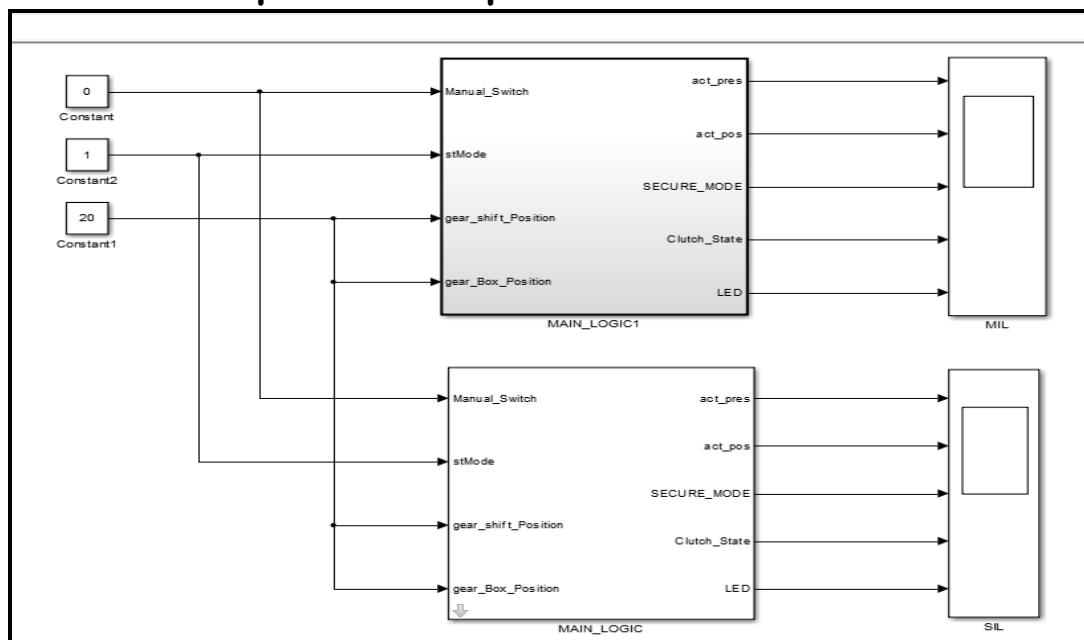
Model Hierarchy/Complexity:

Test 1

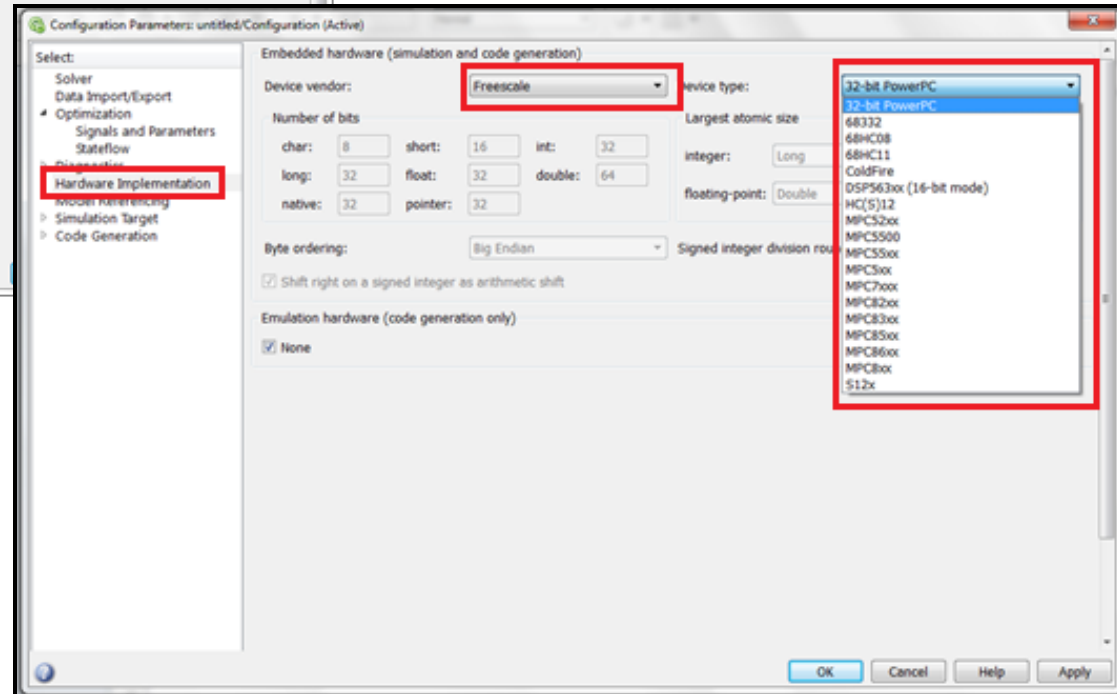
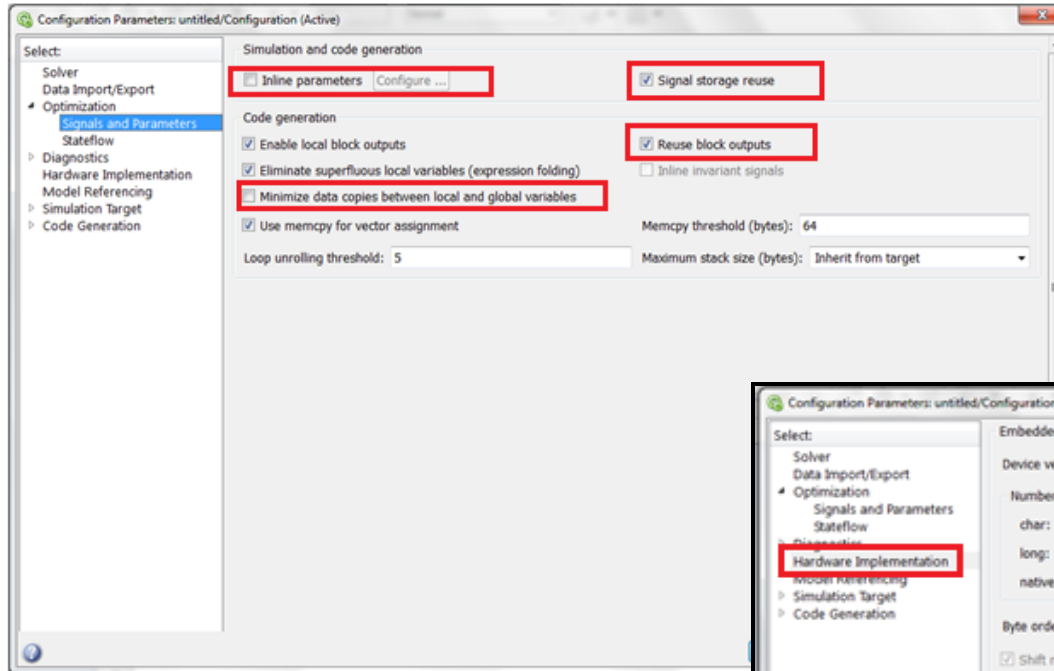
		D1	C1	MCDC
<u>AUTO CLUTCH MODEL5</u>	51	97%	69%	38%
... <u>AUTO CLUTCH MODEL</u>	50	97%	69%	38%
..... <u>SF: AUTO CLUTCH MODEL</u>	49	97%	69%	38%
..... <u>SF: MAIN FUNCTION</u>	46	97%	69%	38%
..... <u>SF: CLUTCH ACTUATOR FUNCTION</u>	32	95%	67%	33%
..... <u>SF: CLTH POSITION MODE</u>	16	95%	67%	33%
..... <u>SF: CLTH PRESSURE MODE</u>	16	95%	67%	33%

Software in Loop

- Results of SIL are compared with Model test results
- Same test cases can be used
- Test source code on development computer



Auto code Generation



Concluding points

- ✓ Automotive Electronics has become competitive necessity to stay ahead of competitors
- ✓ Increased risk of product defects due to Complex E/E architecture ,large number of ECU & seamless networking
- ✓ Test & development strategies to ensure flawless functioning of individual and all ECU's together
- ✓ Way forward is to deploy robust development process, In depth testing, V&V
- ✓ Use of qualified and proven tools & services

Thanks for your attention



Summary

- As on now, attempt is made to check proof of concept
- Require thorough checking on testbed & on vehicle to firm up control strategies
- Important benefit derived is identifying gaps while modeling and same were addressed and verified
- Evolved model around 10 times to reach final executable requirements
- Realized: Timings, State transition sequence is a key parameter to achieve synchronization
- Need to incorporate self learning & neural logic to build right control mechanism