Continental
The Future in Motion

MIL/SIL/PIL Approach
A new paradigm in Model Based Development

Narayanamurthy Srinivas, Narendrakumar Panditi

Stefan Schmidt, Ralf Garrelfs

Agenda

1 Motivation - Model Based Development (MBD)

2 Model verified by Simulation (MvS)

3 Case study on MIL/SIL/PIL

4 MIL/SIL/PIL Simulation results in SDA

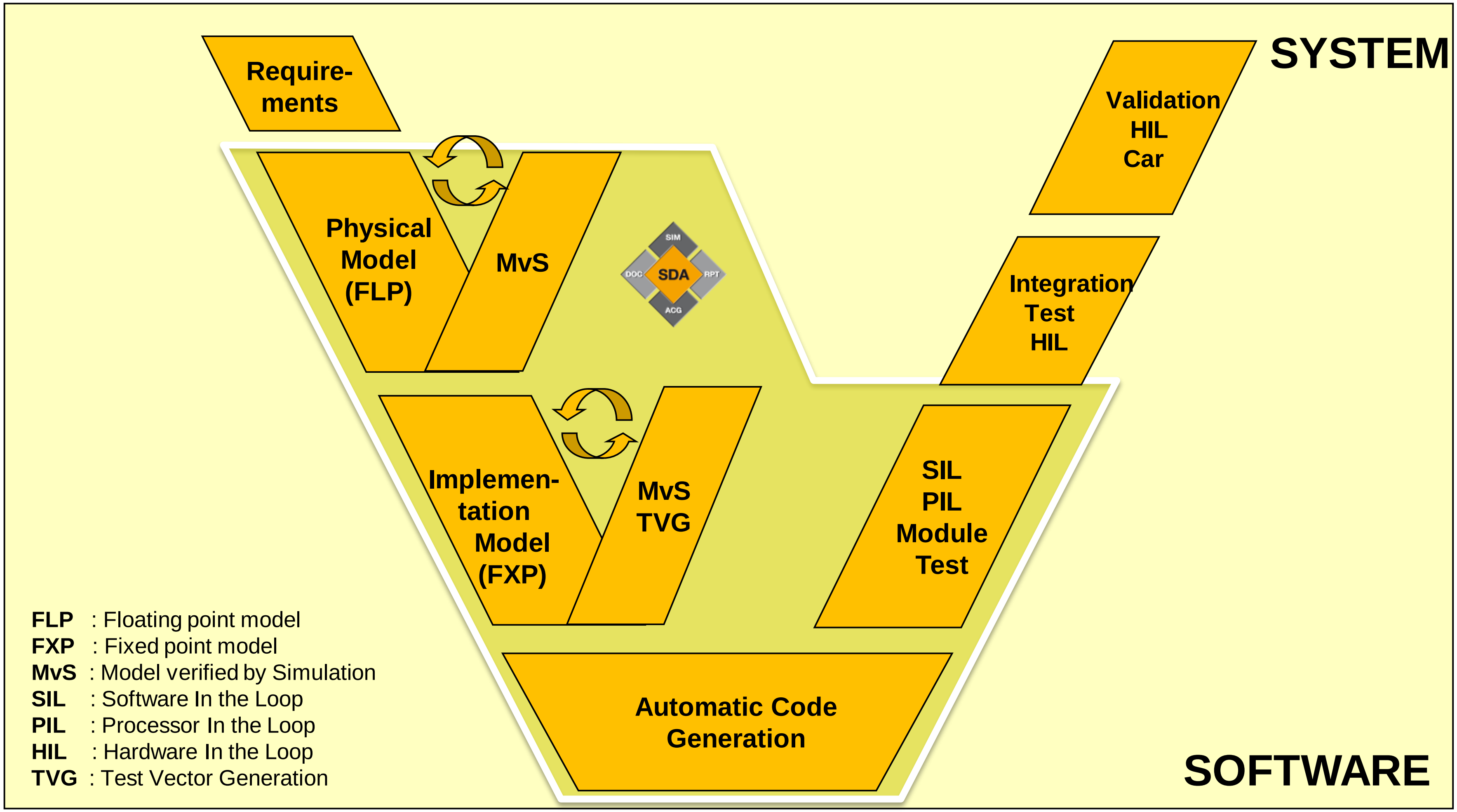
5 Comparison of MIL/SIL/PIL results

6 Conclusion

Motivation - Model Based Development (MBD)

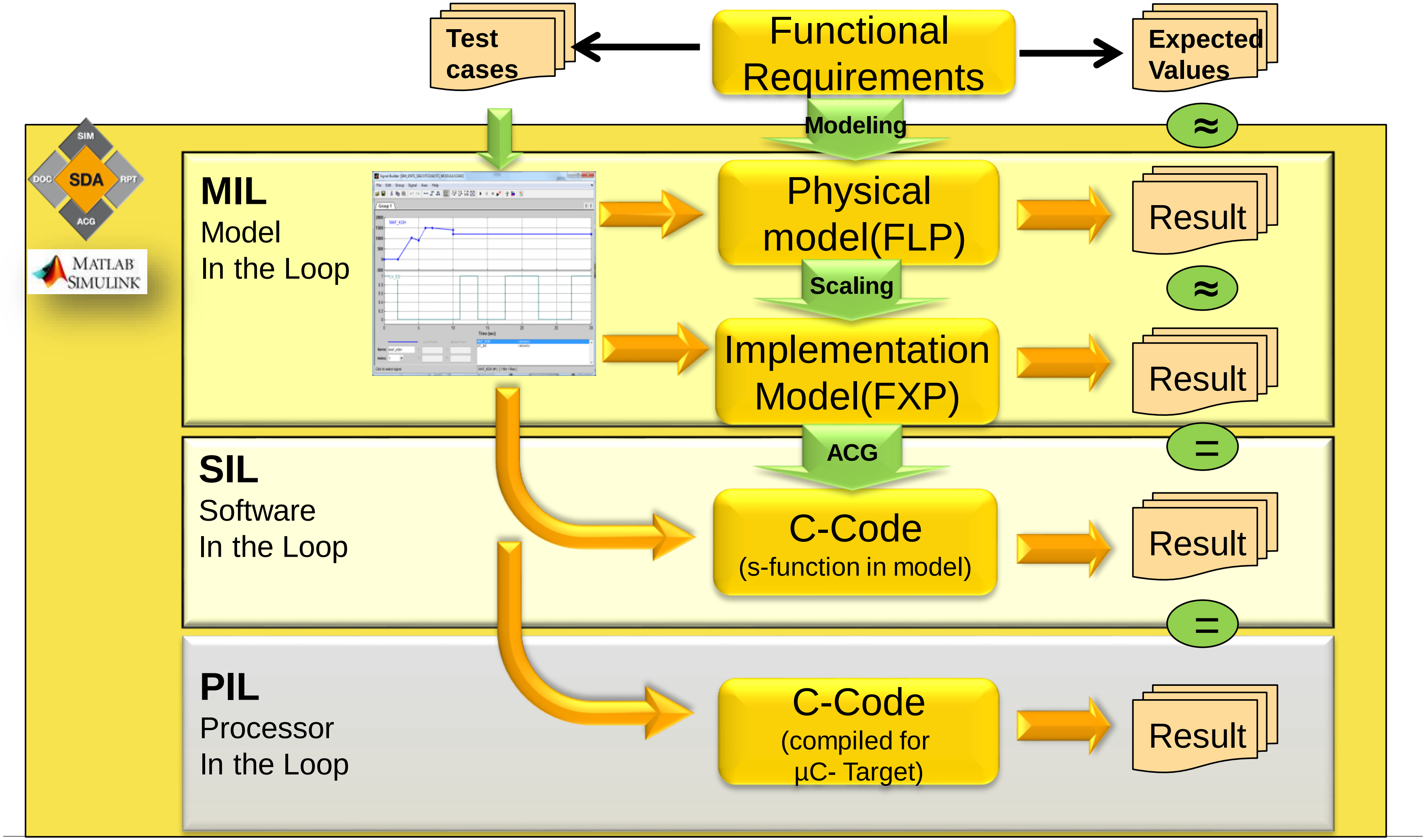
	Manual	Model In the Loop (MIL)	Model In the Loop (MIL) Software In the Loop (SIL) Processor In the Loop (PIL)
Specification Design	Manual in the form of document	Model design using MBD MIL: Model verification	Model design using MBD MIL: Model verification
Coding	Manual coding	Auto code generation (ACG)	Auto code generation (ACG)
Code Verification	Manual prepared test cases to perform Unit Testing	Tool generated test cases to perform unit testing	Reuse MIL test cases SIL : Software verification PIL : Software verification on Target processor or equivalent instruction set simulator

Model Based Development: V- Cycle



FLP	: Floating point model
FXP	: Fixed point model
MvS	: Model verified by Simulation
SIL	: Software In the Loop
PIL	: Processor In the Loop
HIL	: Hardware In the Loop
TVG	: Test Vector Generation

Model Verified by Simulation (MvS)



Test cases

Functional Requirements

Expected Values

Modeling

≈

SIM

DOC

SDA

RPT

ACG

MATLAB SIMULINK

MIL
Model
In the Loop

Physical model(FLP)

Result

Scaling

≈

Implementation Model(FXP)

Result

ACG

=

SIL
Software
In the Loop

C-Code
(s-function in model)

Result

=

PIL
Processor
In the Loop

C-Code
(compiled for µC- Target)

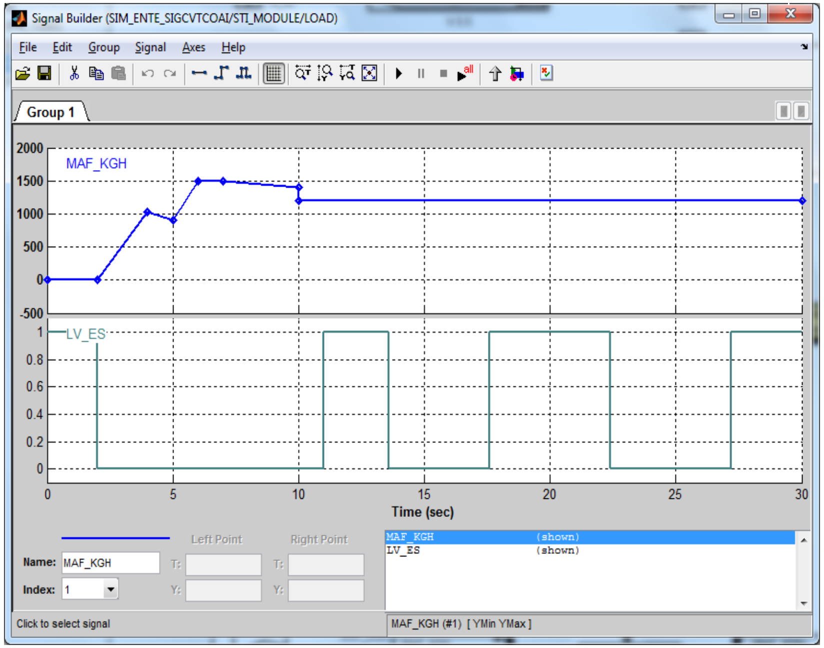
Result

Definition – MIL/SIL/PIL

MIL Model In the Loop	SIL Software In the Loop	PIL Processor In the Loop
Refers to the kind of testing done to verify the accuracy / acceptability of a plant model or a control system. MIL testing means that the model and its environment are simulated in the modeling framework without any physical hardware components.	Refers to the kind of testing done to validate the behavior of the auto generated code used in the controller. The embedded software is tested within a simulated environment model but without any hardware.	Refers to the kind of testing done to validate the referenced model by generating production code using the model reference target. The code is cross-compiled for and executed on a target processor or an equivalent instruction set simulator.
MIL allows testing at early stages of the development cycle.	SIL also allows to verify the code coverage.	PIL level of testing can reveal faults that are caused by the target compiler or by the processor architecture.

Case study on Engine Temperature function

1 Test suite for calculation of load information for coolant temperature model



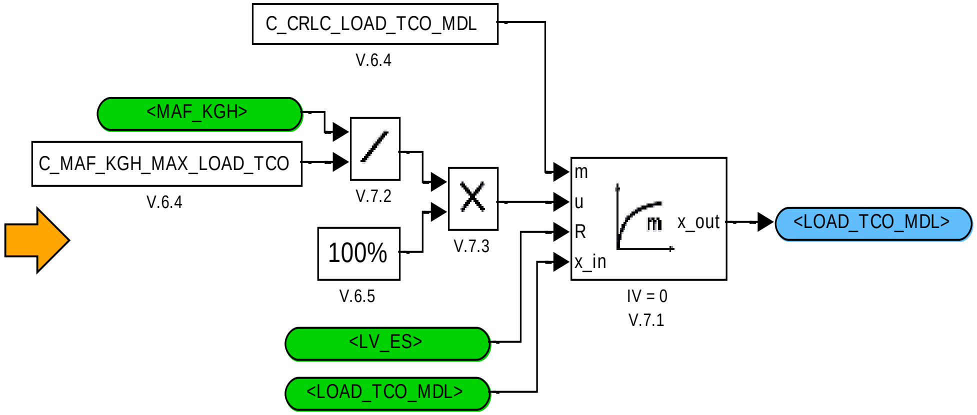
Name: MAF_KGH

Index: 1

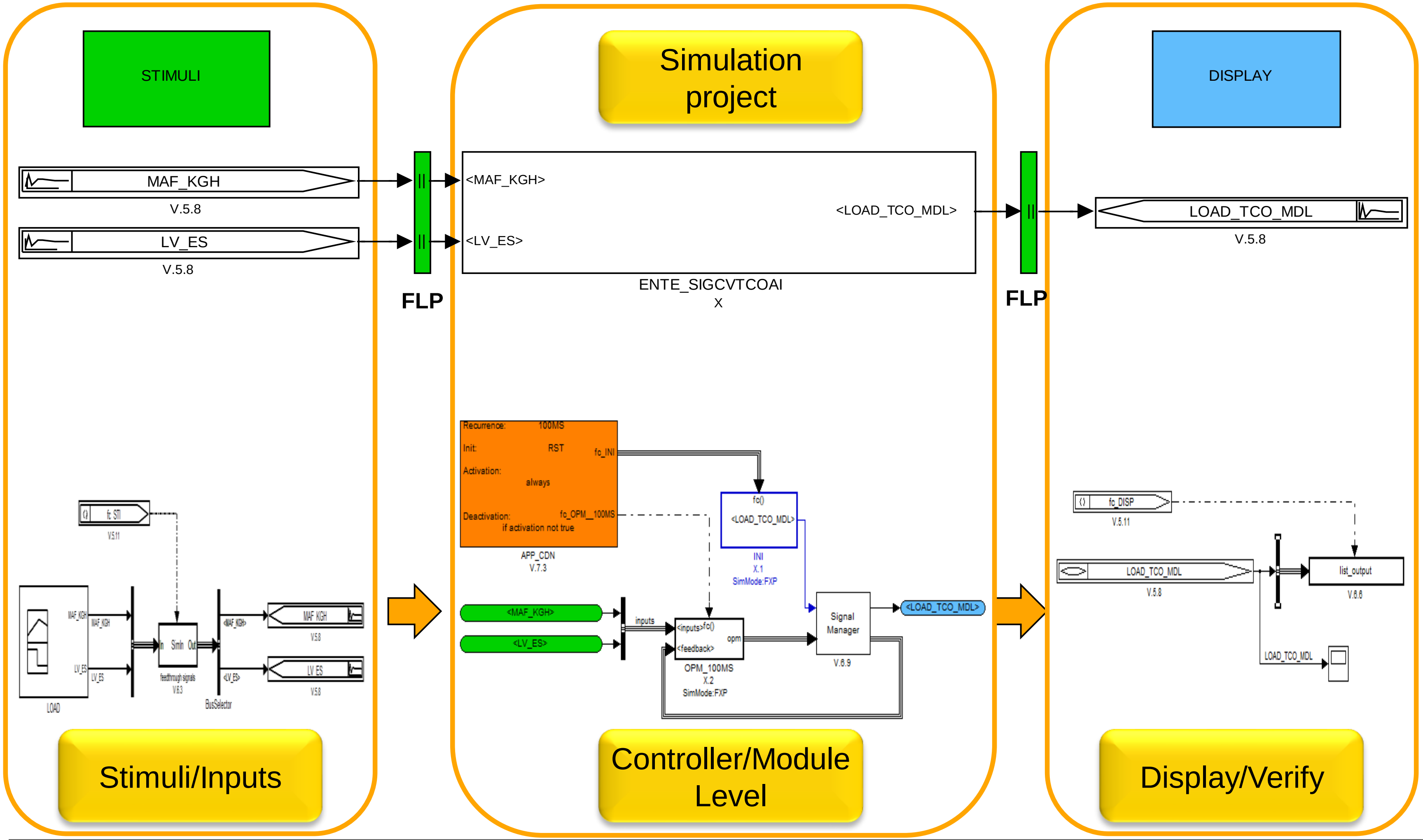
```
MAF_KGH     (shown)
LV_ES       (shown)
```

Click to select signal

MAF_KGH (#1) [YMin YMax]



Model In the Loop (MIL): Floating point model



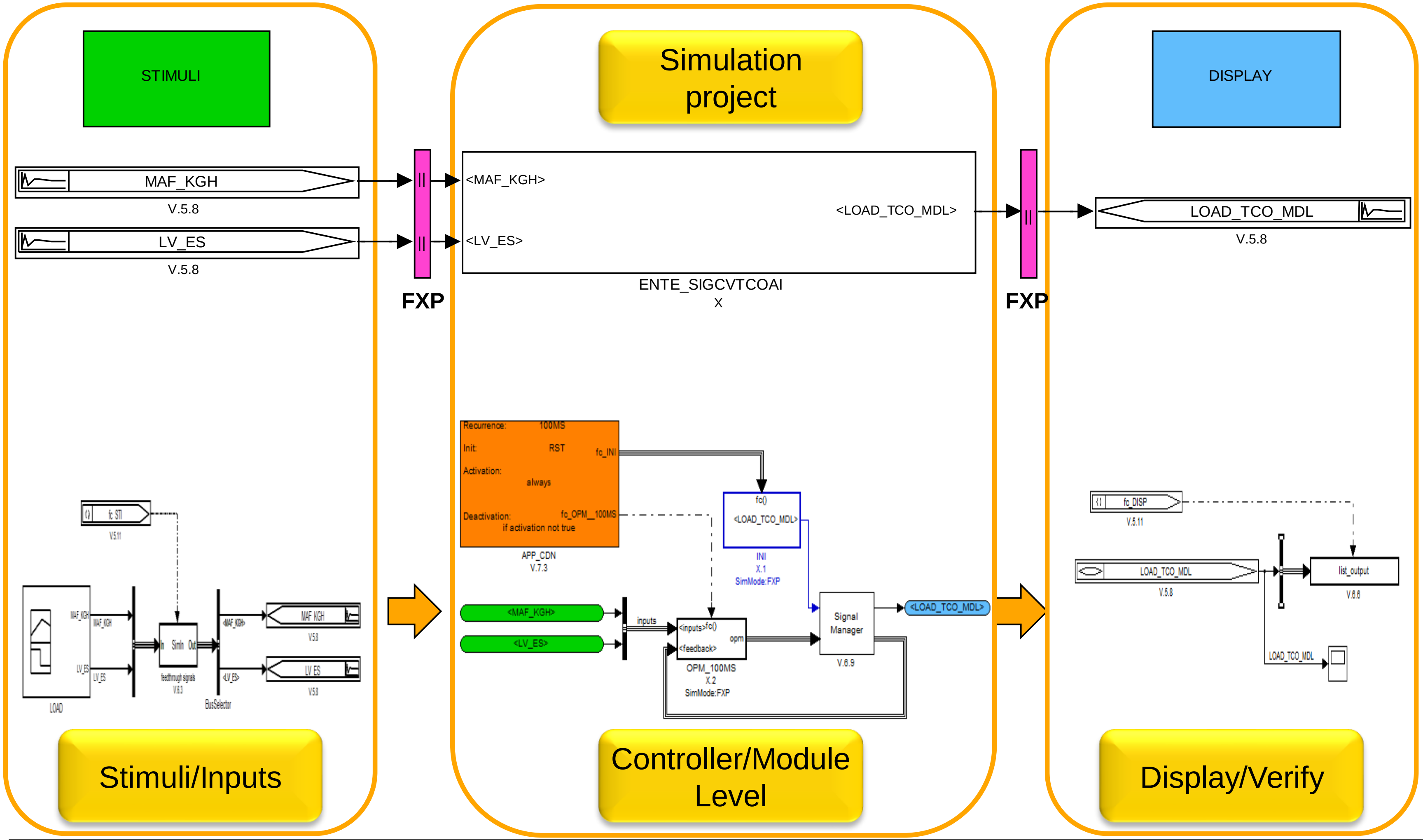
Simulation project

Stimuli/Inputs

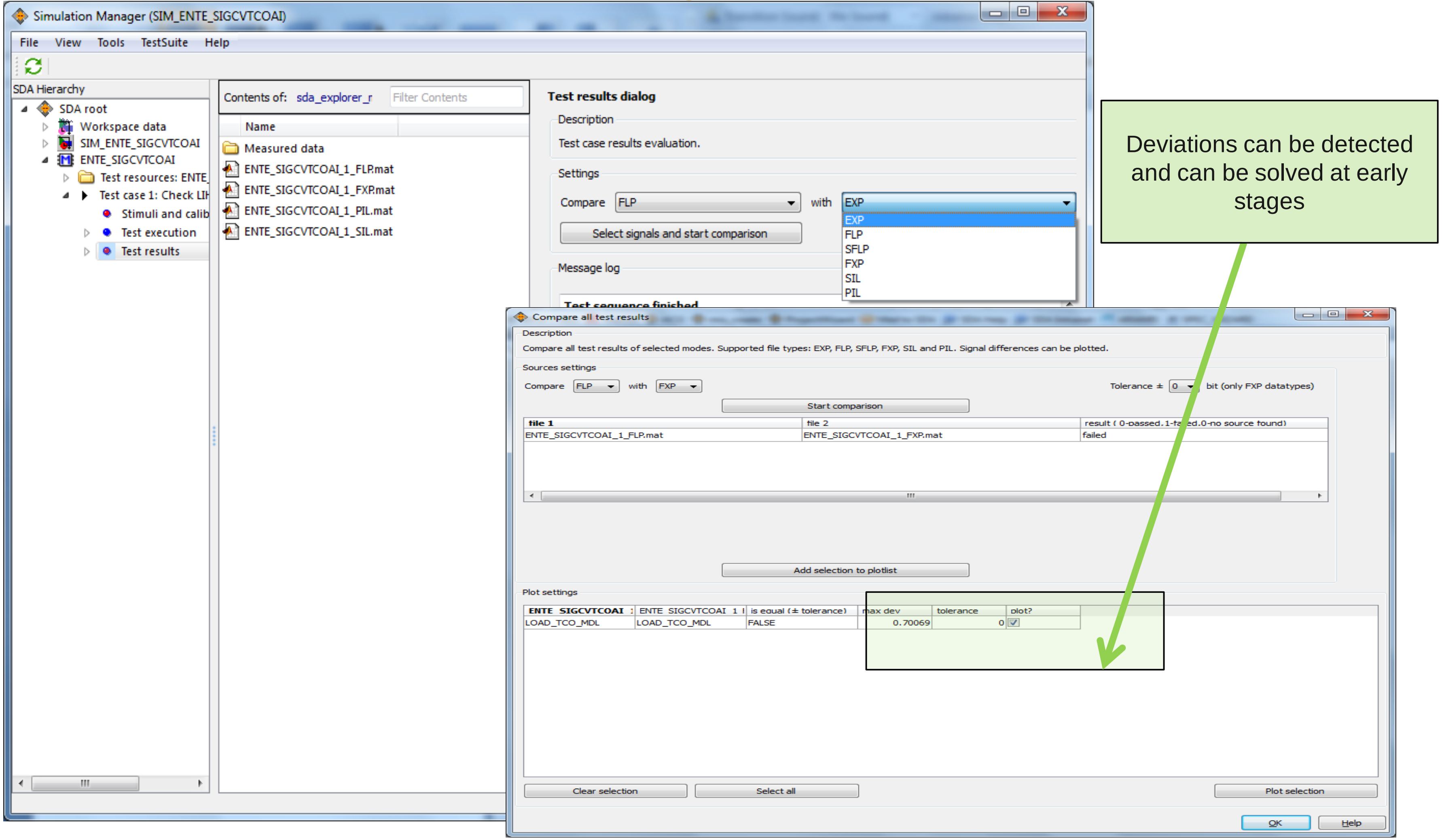
Controller/Module Level

Display/Verify

Model In the Loop (MIL): Fixed point model



MvS: SDA Simulation Manager



Test case results evaluation.

Compare FLP with EXP

Test sequence finished

Compare all test results of selected modes. Supported file types: EXP, FLP, SFLP, FXP, SIL and PIL. Signal differences can be plotted.

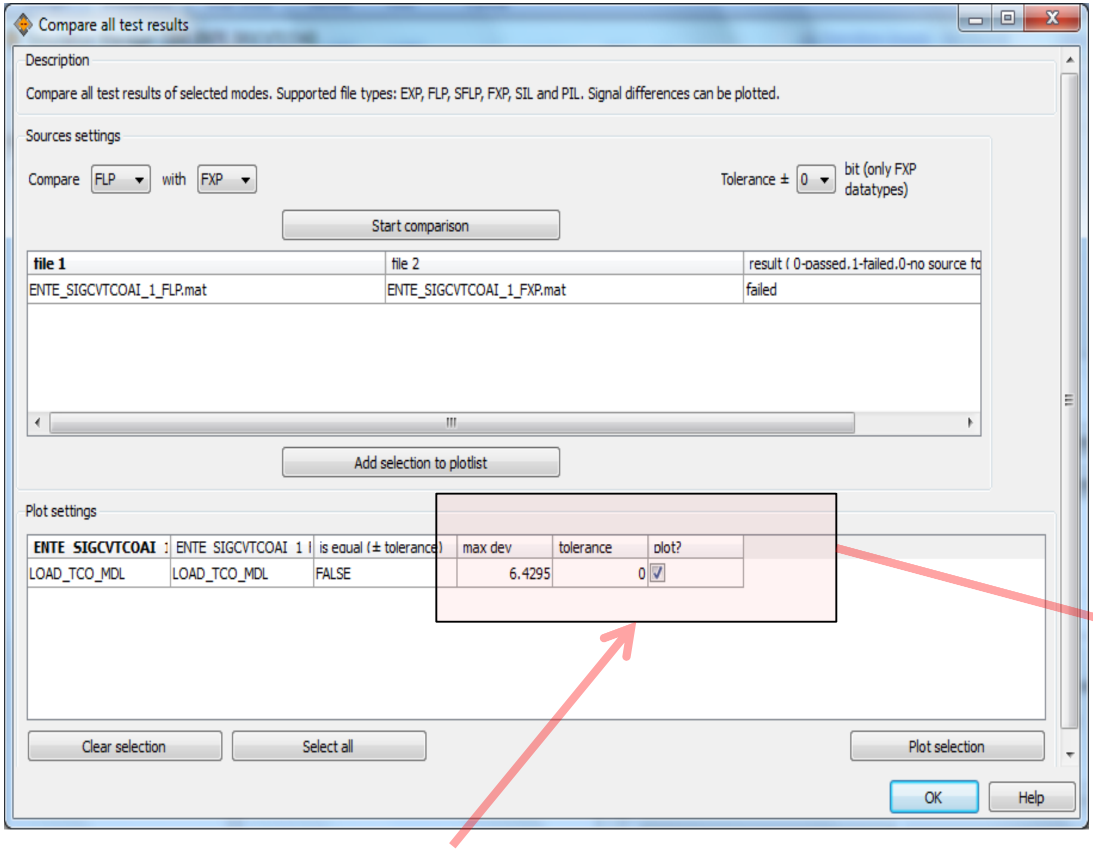
Compare FLP with FXP Tolerance ± 0 bit (only FXP datatypes)

file 1	file 2	result (0-passed. 1-fa ed. 0-no source found)
ENTE_SIGCVTCOAI_1_FLP.mat	ENTE_SIGCVTCOAI_1_FXP.mat	failed

ENTE SIGCVTCOAI 1	ENTE SIGCVTCOAI 1 I	is equal (± tolerance)	max dev	tolerance	plot?
LOAD_TCO_MDL	LOAD_TCO_MDL	FALSE	0.70069	0	☑

Deviations can be detected and can be solved at early stages

Comparison results MIL - FLP/FXP (Error)



Compare all test results

Description

Compare all test results of selected modes. Supported file types: EXP, FLP, SFLP, FXP, SIL and PIL. Signal differences can be plotted.

Sources settings

Compare: FLP with FXP

Tolerance ± 0 bit (only FXP datatypes)

file 1	file 2	result (0-passed.1-failed.0-no source fc
ENTE_SIGCVTCOAI_1_FLP.mat	ENTE_SIGCVTCOAI_1_FXP.mat	failed

Plot settings

ENTE_SIGCVTCOAI_1	ENTE_SIGCVTCOAI_1_I	is equal (± tolerance)	max dev	tolerance	plot?
LOAD_TCO_MDL	LOAD_TCO_MDL	FALSE	6.4295	0	☑

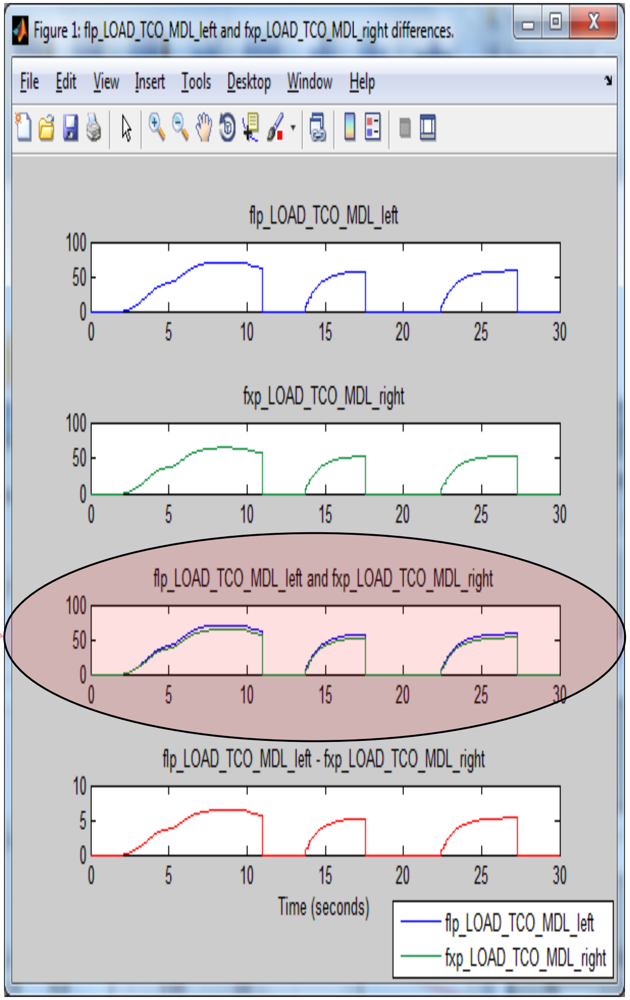
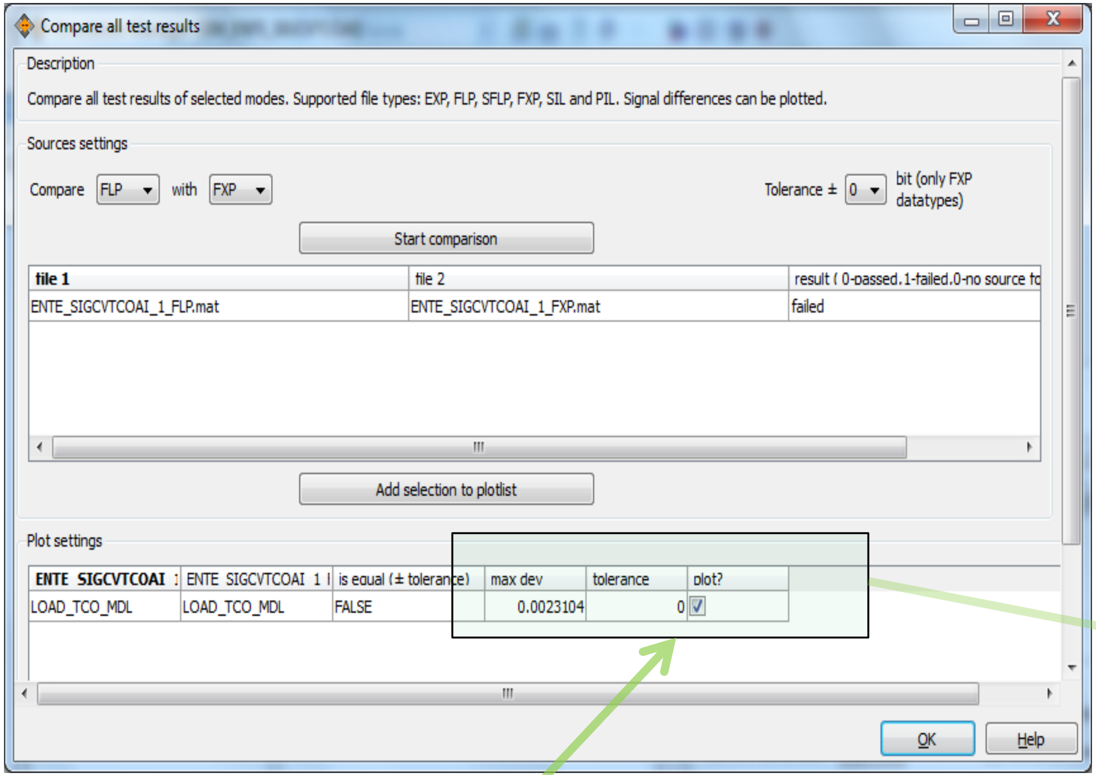


Figure 1: flp_LOAD_TCO_MDL_left and fxp_LOAD_TCO_MDL_right differences.

Deviations due to wrong scaling

Comparison results - FLP/FXP (Corrected Case)



Compare all test results

Description

Compare all test results of selected modes. Supported file types: EXP, FLP, SFLP, FXP, SIL and PIL. Signal differences can be plotted.

Sources settings

Compare FLP ▾ with FXP ▾ Tolerance ± 0 ▾ bit (only FXP datatypes)

Start comparison

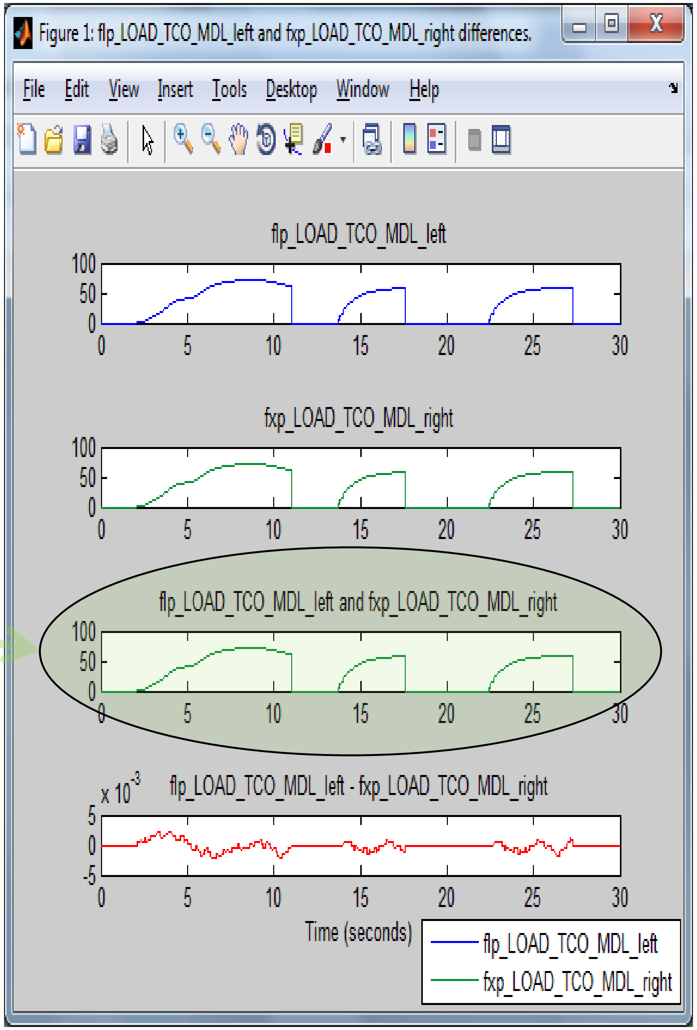
file 1	file 2	result (0-passed, 1-failed, 0-no source fo
ENTE_SIGCVTCOAI_1_FLP.mat	ENTE_SIGCVTCOAI_1_FXP.mat	failed

Add selection to plotlist

Plot settings

ENTE_SIGCVTCOAI_	ENTE_SIGCVTCOAI_1_I	is equal (± tolerance)	max dev	tolerance	plot?
LOAD_TCO_MDL	LOAD_TCO_MDL	FALSE	0.0023104	0	☑

OK Help



Deviations are with in the resolution

Present Situation after MIL

1 Random test cases are generated to test production code.

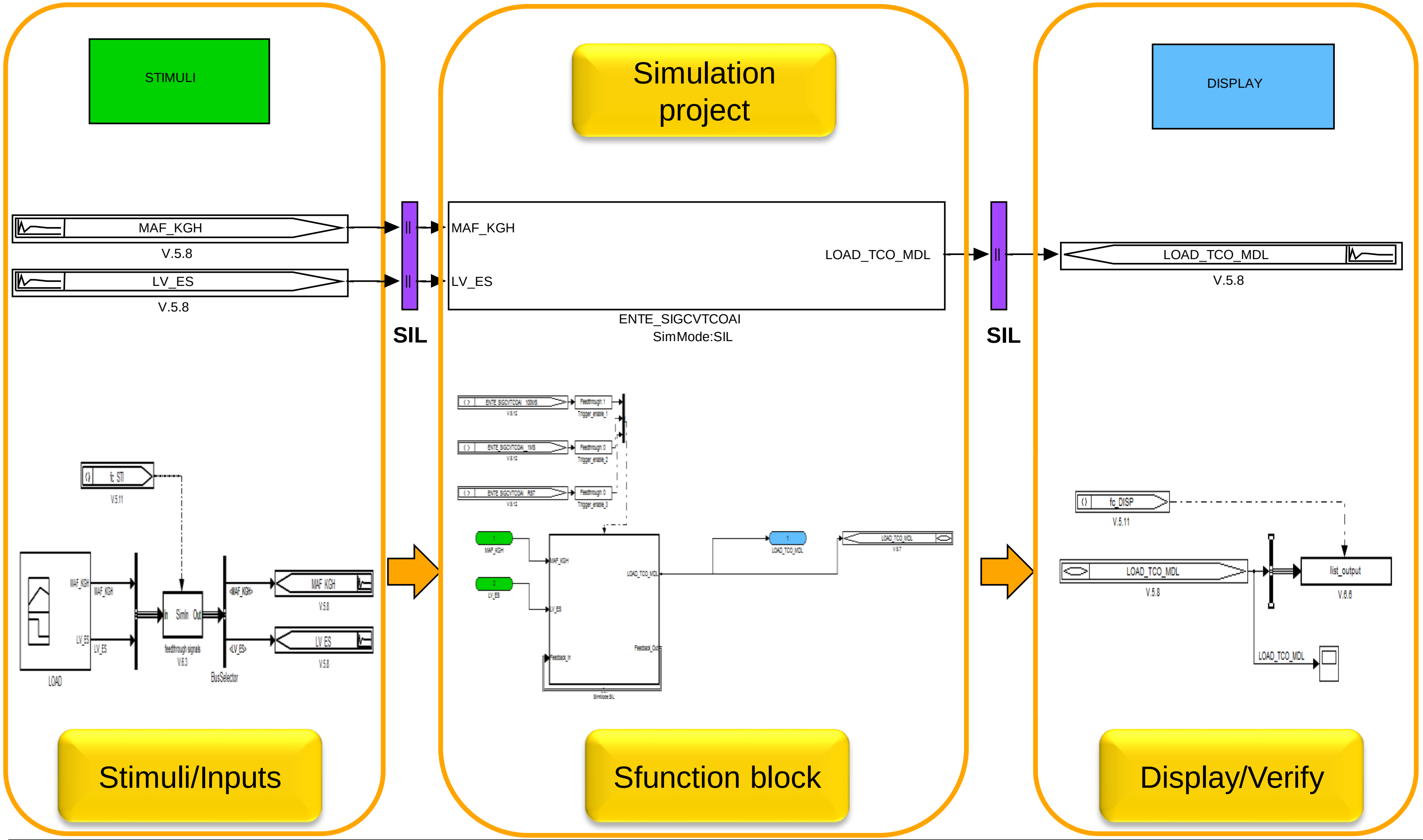
2 Execute generated test cases in the project environment.

3 More effort is required to prepare test cases to verify production code.

4 Completely different test cases are used to verify model and generated code.

1 Wouldn’t it be nice to reuse the MIL test cases for test of the Automatically Generated Code ?

Software in the Loop: SIL



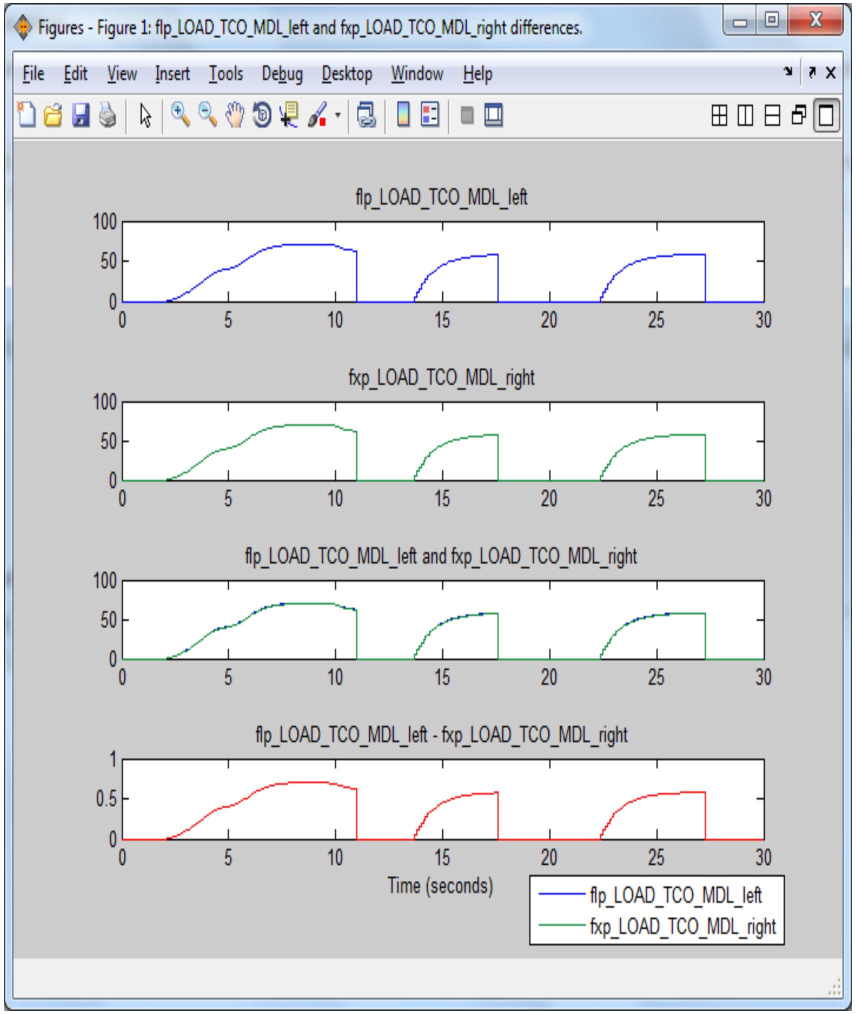
Simulation project

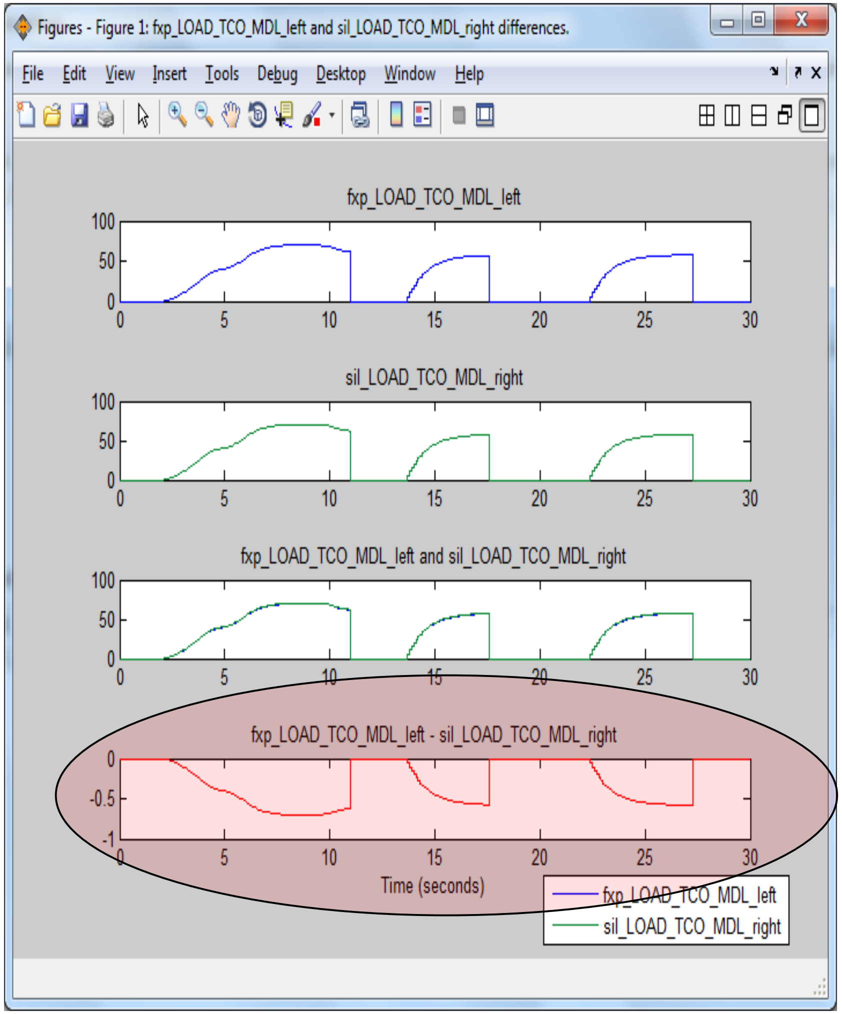
Stimuli/Inputs

Sfunction block

Display/Verify

Comparison results - MIL/SIL (wrong case)


Figures - Figure 1: flp_LOAD_TCO_MDL_left and fxp_LOAD_TCO_MDL_right differences.
File
Edit
View
Insert
Tools
Debug
Desktop
Window
Help
flp_LOAD_TCO_MDL_left
fxp_LOAD_TCO_MDL_right
flp_LOAD_TCO_MDL_left and fxp_LOAD_TCO_MDL_right
flp_LOAD_TCO_MDL_left - fxp_LOAD_TCO_MDL_right
100
50
0
1
0.5
0
5
10
15
20
25
30
Time (seconds)
flp_LOAD_TCO_MDL_left
fxp_LOAD_TCO_MDL_right


Figures - Figure 1: fxp_LOAD_TCO_MDL_left and sil_LOAD_TCO_MDL_right differences.
File
Edit
View
Insert
Tools
Debug
Desktop
Window
Help
fxp_LOAD_TCO_MDL_left
sil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left and sil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left - sil_LOAD_TCO_MDL_right
100
50
0
-0.5
-1
5
10
15
20
25
30
Time (seconds)
fxp_LOAD_TCO_MDL_left
sil_LOAD_TCO_MDL_right

Comparison results - MIL/SIL (correct case)

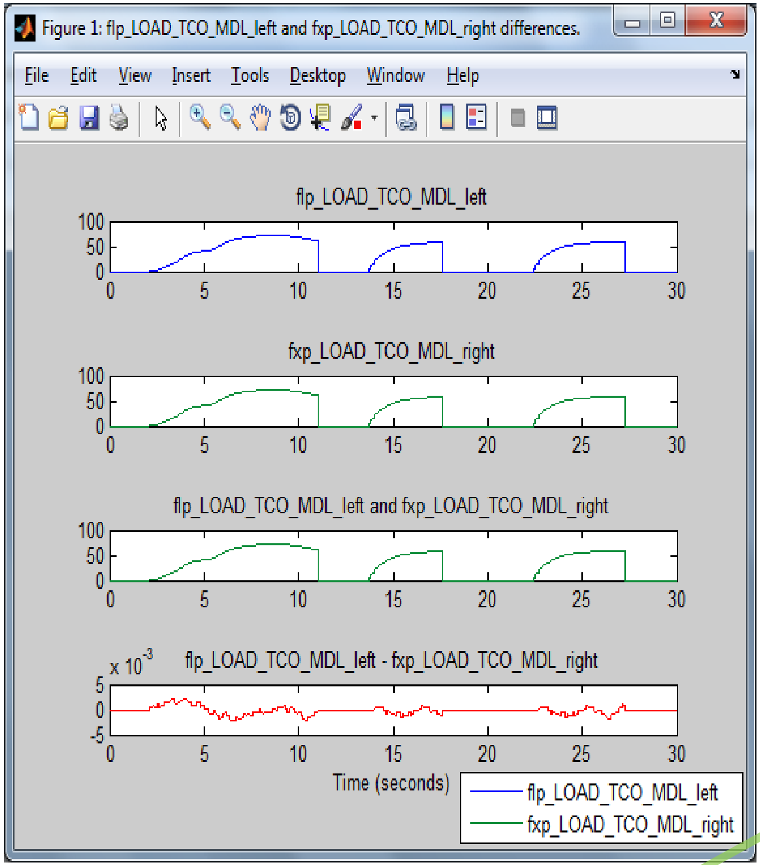


Figure 1: flp_LOAD_TCO_MDL_left and fxp_LOAD_TCO_MDL_right differences.

File Edit View Insert Tools Desktop Window Help
flp_LOAD_TCO_MDL_left
fxp_LOAD_TCO_MDL_right
flp_LOAD_TCO_MDL_left and fxp_LOAD_TCO_MDL_right
x 10^-3
flp_LOAD_TCO_MDL_left - fxp_LOAD_TCO_MDL_right
100
50
0
5
-5
0 5 10 15 20 25 30
Time (seconds)
flp_LOAD_TCO_MDL_left
fxp_LOAD_TCO_MDL_right

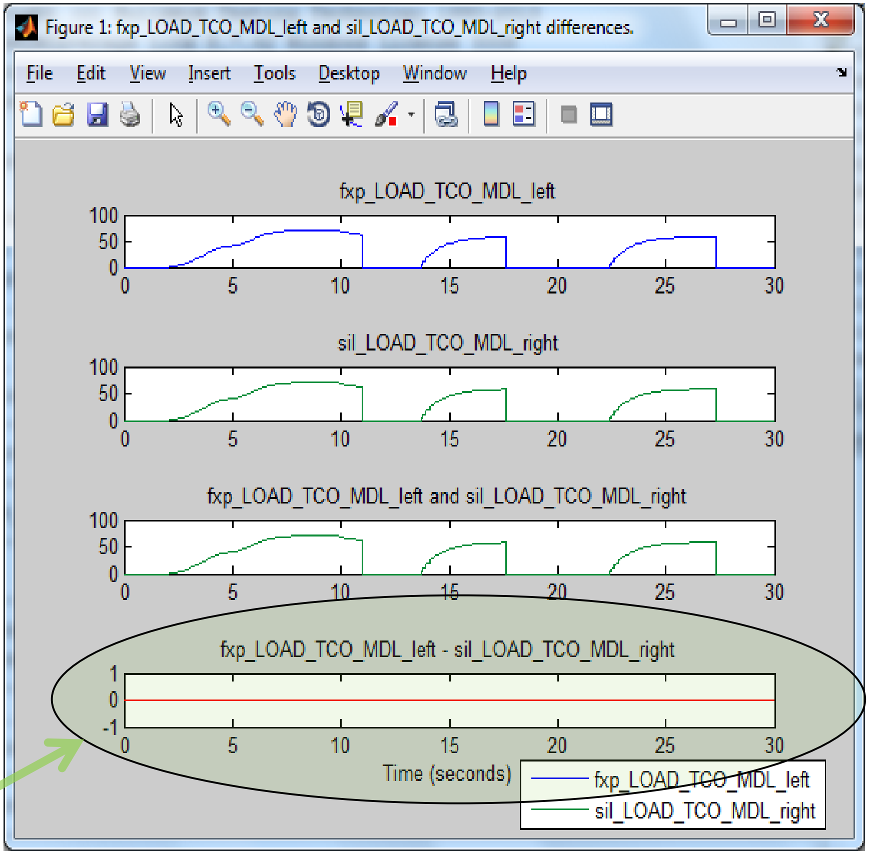
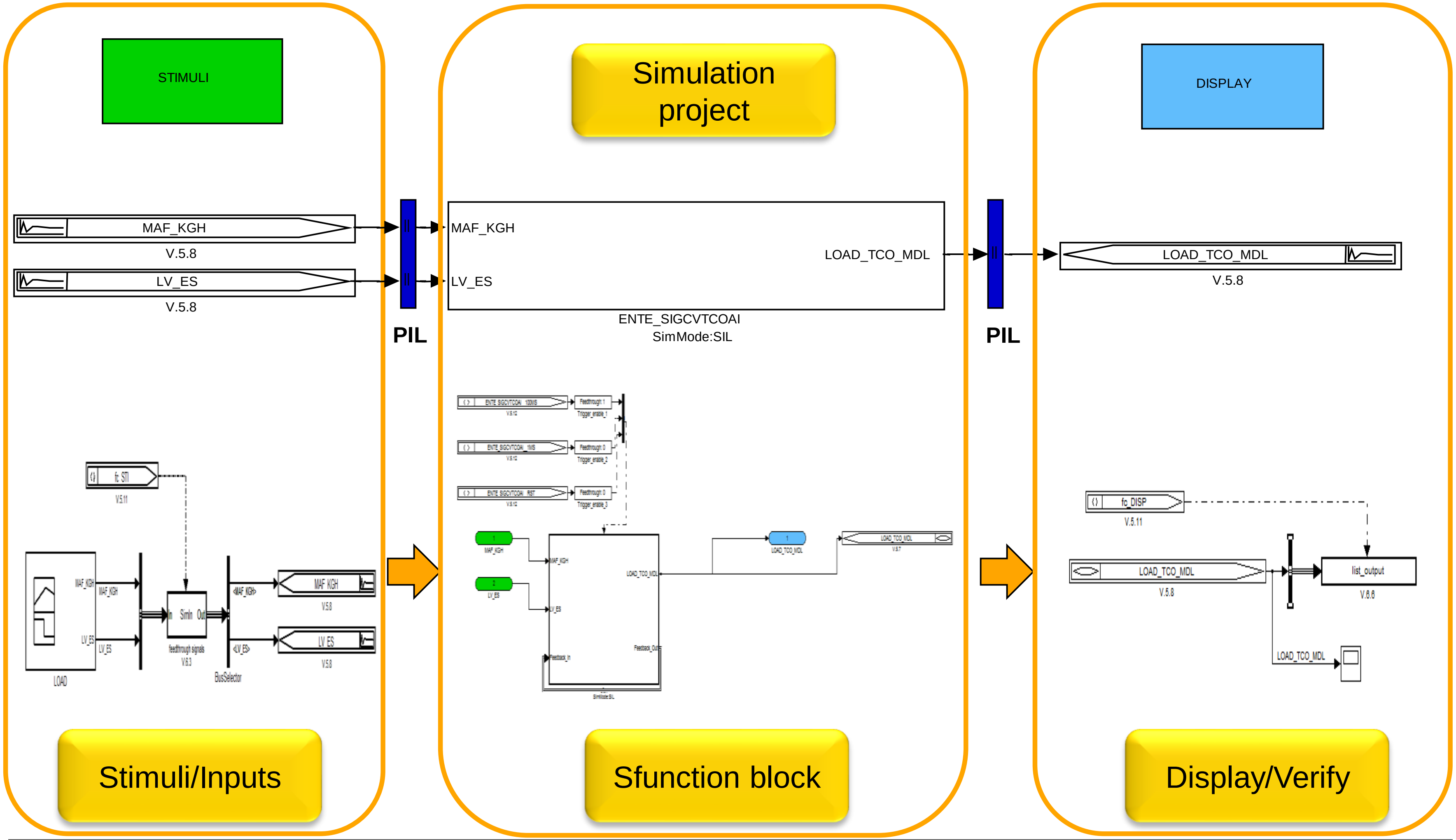


Figure 1: fxp_LOAD_TCO_MDL_left and sil_LOAD_TCO_MDL_right differences.

File Edit View Insert Tools Desktop Window Help
fxp_LOAD_TCO_MDL_left
sil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left and sil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left - sil_LOAD_TCO_MDL_right
100
50
0
1
-1
0 5 10 15 20 25 30
Time (seconds)
fxp_LOAD_TCO_MDL_left
sil_LOAD_TCO_MDL_right

No Deviation

Processor In the Loop: PIL



STIMULI

MAF_KGH
V.5.8

LV_ES
V.5.8

PIL

Simulation project

MAF_KGH

LV_ES

LOAD_TCO_MDL

ENTE_SIGCVTCOAI
SimMode:SIL

PIL

DISPLAY

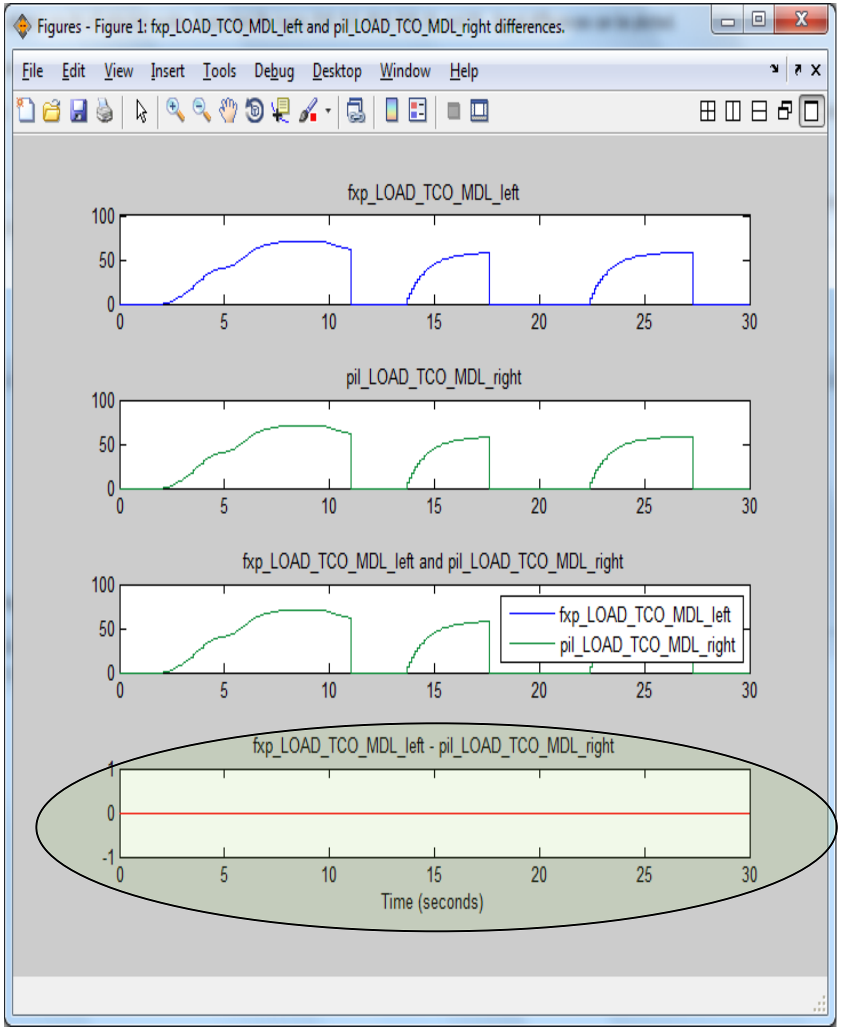
LOAD_TCO_MDL
V.5.8

Stimuli/Inputs

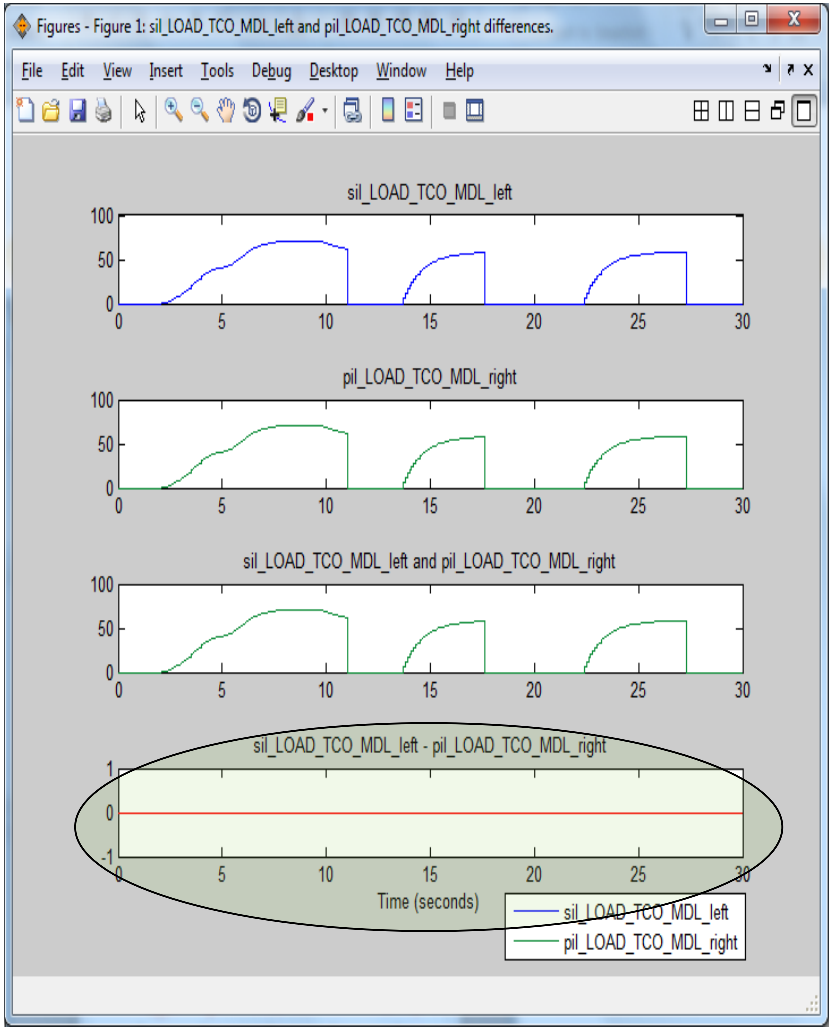
Sfunction block

Display/Verify

Comparison results - SIL/PIL

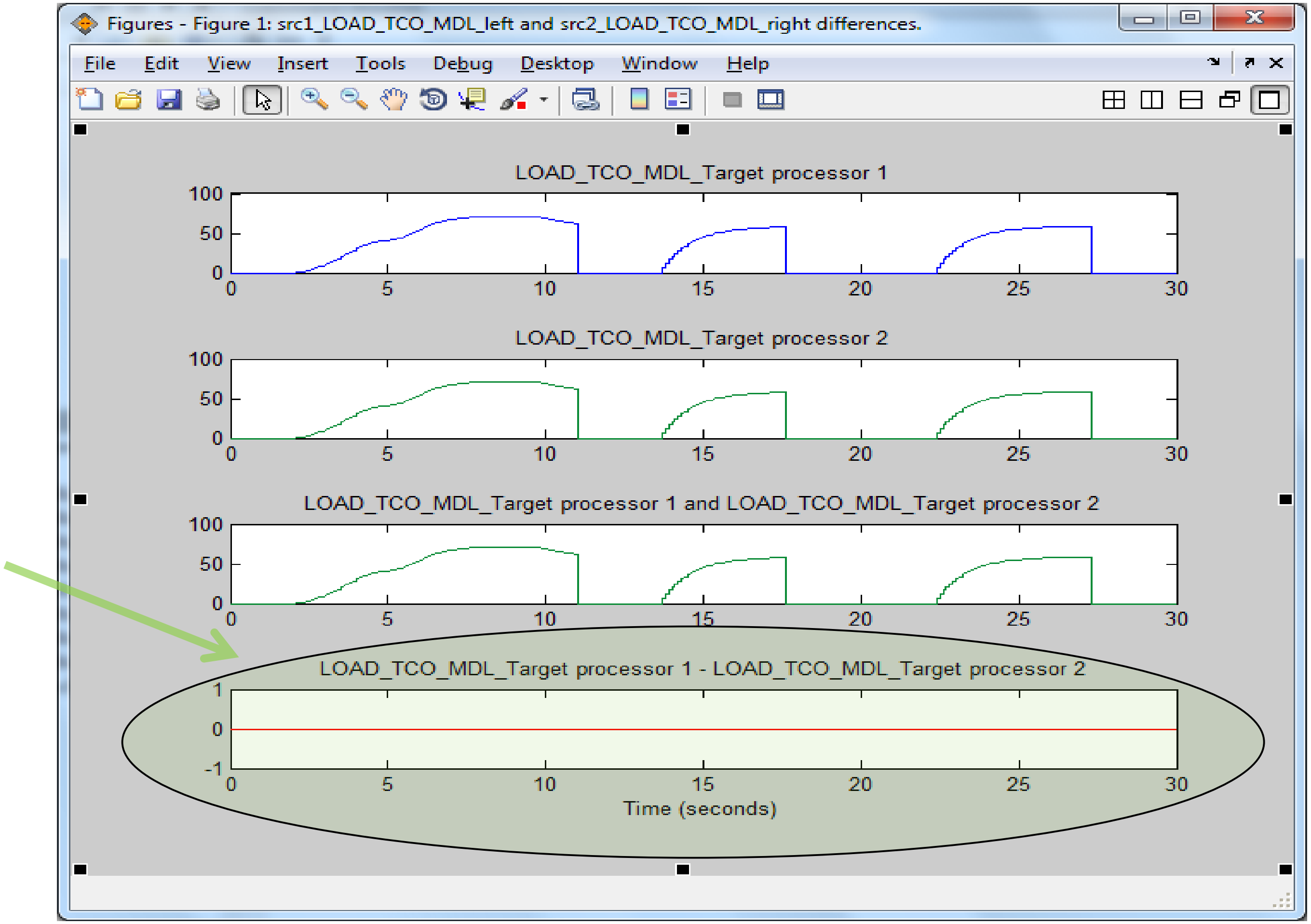

Figures - Figure 1: fxp_LOAD_TCO_MDL_left and pil_LOAD_TCO_MDL_right differences.

File Edit View Insert Tools Debug Desktop Window Help
fxp_LOAD_TCO_MDL_left
pil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left and pil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left
pil_LOAD_TCO_MDL_right
fxp_LOAD_TCO_MDL_left - pil_LOAD_TCO_MDL_right
100
50
0
1
-1
0 5 10 15 20 25 30
Time (seconds)


Figures - Figure 1: sil_LOAD_TCO_MDL_left and pil_LOAD_TCO_MDL_right differences.

File Edit View Insert Tools Debug Desktop Window Help
sil_LOAD_TCO_MDL_left
pil_LOAD_TCO_MDL_right
sil_LOAD_TCO_MDL_left and pil_LOAD_TCO_MDL_right
sil_LOAD_TCO_MDL_left - pil_LOAD_TCO_MDL_right
100
50
0
1
-1
0 5 10 15 20 25 30
Time (seconds)
sil_LOAD_TCO_MDL_left
pil_LOAD_TCO_MDL_right

PIL results for different target processors - Reusability


Figures - Figure 1: src1_LOAD_TCO_MDL_left and src2_LOAD_TCO_MDL_right differences.
File Edit View Insert Tools Debug Desktop Window Help
LOAD_TCO_MDL_Target processor 1
LOAD_TCO_MDL_Target processor 2
LOAD_TCO_MDL_Target processor 1 and LOAD_TCO_MDL_Target processor 2
LOAD_TCO_MDL_Target processor 1 - LOAD_TCO_MDL_Target processor 2
100
50
0
1
-1
0 5 10 15 20 25 30
Time (seconds)

No Deviations

Conclusion

1 Necessary test effort can be essentially minimized across simulations.

2 Tests suites are portable and reusable.

3 Cost-efficient consistent testing for all phases of the development:
One test suite for all development phases (MIL, SIL, PIL).

4 Early malfunction detection.

5 Eases the updating of test suites for changed requirements.

6 Shorter development process resulting in significant time-to-market advantage.

Thank you

for your attention!