


Report Generated by Test Manager

Title: Test
Author: Sonko Muhammed R.
Date: 29-Oct-2024 12:11:32

Test Environment

Platform: PCWIN64
MATLAB: (R2022b)

Summary

Name	Outcome	Duration (Seconds)
 DiagReq1		4.149

DiagReq1

Test Result Information

Result Type: Test Case Result
Parent: None
Start Time: 29-Oct-2024 12:07:22
End Time: 29-Oct-2024 12:07:26
Outcome: Passed

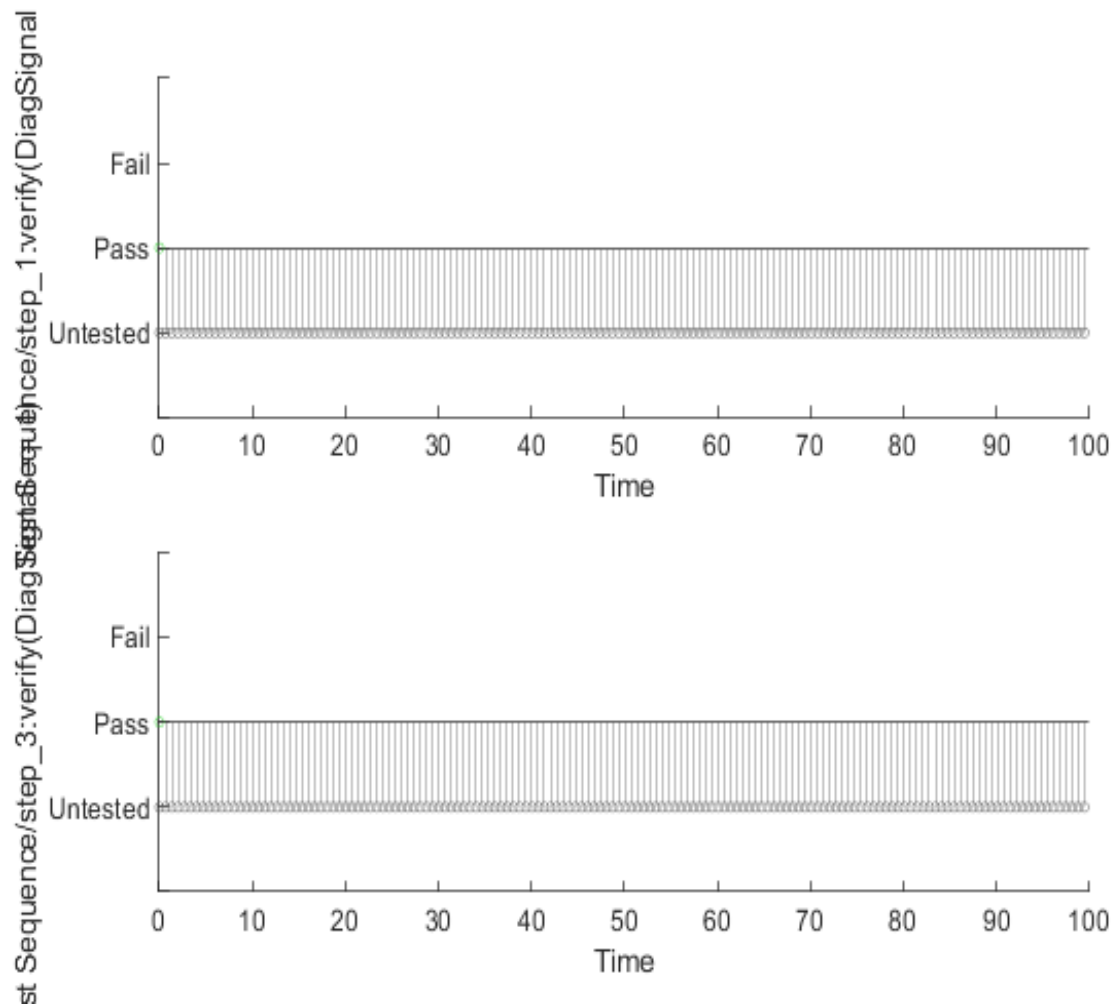
Test Case Information

Name: DiagReq1
Type: Simulation Test

Verify Result

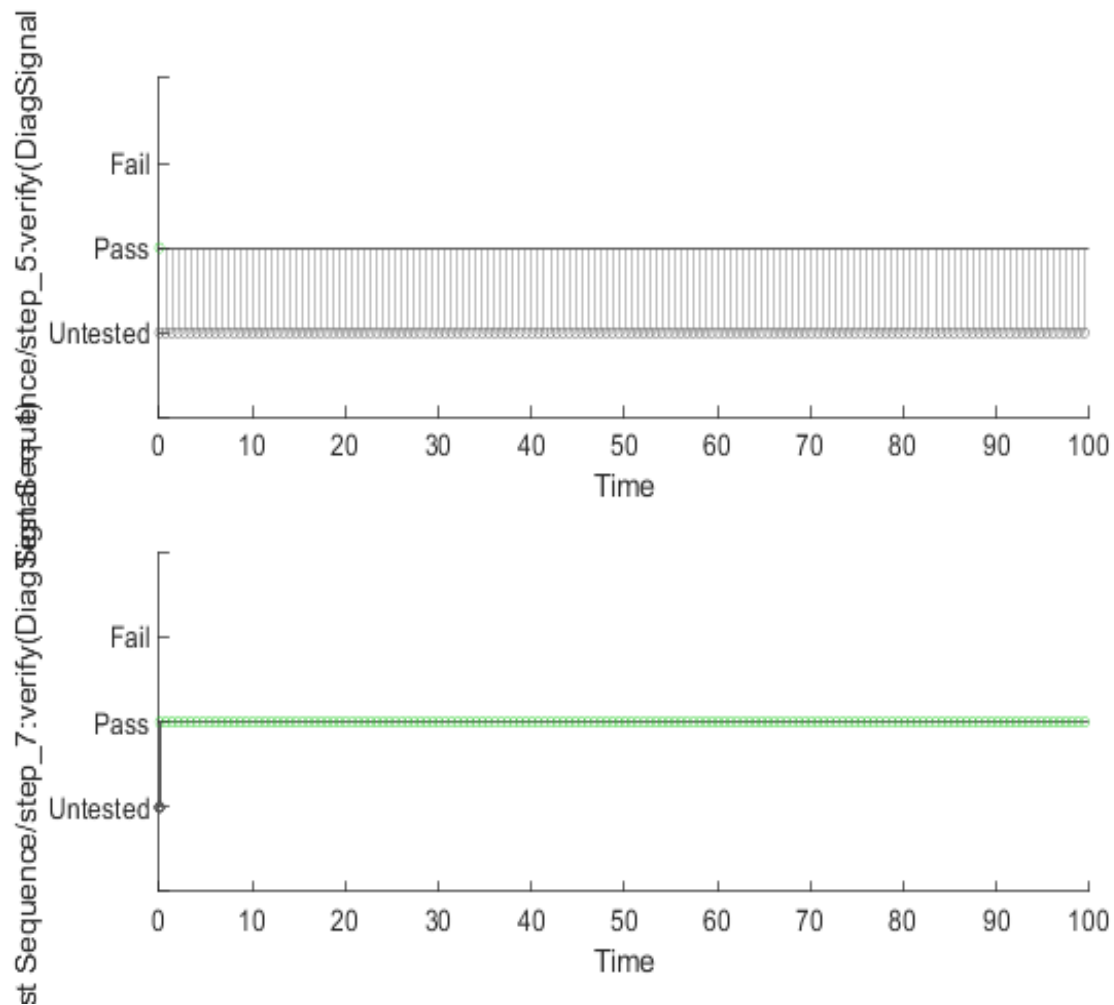
Name		Link to Plot
✔ Test Sequence/step_1.verify(DiagSignal == 1)		Link
✔ Test Sequence/step_3.verify(DiagSignal == 1)		Link
✔ Test Sequence/step_5.verify(DiagSignal == 0)		Link
✔ Test Sequence/step_7.verify(DiagSignal == 1)		Link

Name	
✔ Test Sequence/step_1.verify(DiagSignal == 1)	
✔ Test Sequence/step_3.verify(DiagSignal == 1)	



[Back to Report Summary](#)[Back to Signal Summary](#)

Name	
✓	Test Sequence/step_5:verify(DiagSignal == 0)
✓	Test Sequence/step_7:verify(DiagSignal == 1)



[Back to Report Summary](#)[Back to Signal Summary](#)

Simulation

System Under Test Information

Model:	Defrost_3
Harness:	Defrost_3_Harness1
Harness Owner:	Defrost_3/Subsystem2
Release:	Current
Simulation Mode:	normal

Override SIL or PIL	0
Mode:	
Configuration Set:	Configuration1
Start Time:	0
Stop Time:	100
Checksum:	3135072260 1869269428 925703688 2968458461
Simulink Version:	10.6
Model Version:	1.0
User ID:	muham
Model Path:	C:\Users\muham\Desktop\MST_IESE_2023\ALTEN\ Model Based Design\Defrost_3.slx
Machine Name:	DESKTOP-OF247DK
Solver Name:	VariableStepDiscrete
Solver Type:	Variable-Step
Max Step Size:	0.0025000000000000001
Simulation Start Time:	2024-10-29 12:07:22
Simulation Stop Time:	2024-10-29 12:07:25
Platform:	PCWIN64

Simulation Logs:

This temporal logic expression has no side-effect. It either needs to be used in an expression, or removed if deemed unnecessary.

Step '[step 5](#)' in Test Sequence '[Defrost 3 Harness1/Test Sequence](#)':

```
after(2.5,msec), verify(DiagSignal == 0);
```

Using the '==' operator to compare expressions of type double in the 'verify' statement can produce unexpected results.

Step '[step 1](#)' in Test Sequence '[Defrost 3 Harness1/Test Sequence](#)':

```
verify(DiagSignal == 1);
```

Using the '==' operator to compare expressions of type double in the 'verify' statement can produce unexpected results.

Step '[step_3](#)' in Test Sequence '[Defrost 3 Harness1/Test Sequence](#)':

```
verify(DiagSignal == 1);
```

Using the '==' operator to compare expressions of type double in the 'verify' statement can produce unexpected results.

Step '[step_5](#)' in Test Sequence '[Defrost 3 Harness1/Test Sequence](#)':

```
after(2.5,msec), verify(DiagSignal == 0);
```

Using the '==' operator to compare expressions of type double in the 'verify' statement can produce unexpected results.

Step '[step_7](#)' in Test Sequence '[Defrost 3 Harness1/Test Sequence](#)':

```
verify(DiagSignal == 1);
```

[Back to Report Summary](#)