

Place Attributes:

Place Names	Initial Markings
Primary	1
PrimaryErratic	0
PrimaryFPorFN	0
PrimaryFailed	0
PrimaryLayerErratic	0
PrimaryLayerSilent	0
PrimarySG	1
PrimarySGFailed	0
PrimarySGSilent	0
PrimarySilent	0
PrioritySelectors	2
SafeState	0
Safing	1
SafingErratic	0
SafingFPorFN	0
SafingFailed	0
SafingLayerErratic	0
SafingLayerSilent	0
SafingSG	1
SafingSGFailed	0
SafingSGSilent	0
SafingSilent	0
UnsafeState	0

Timed Activity:	CCF
	Rate
Distribution Parameters	$\text{fr\_complex} * (1\text{-p\_individual}) * ((\text{Primary}\text{->Mark})+(\text{PrimarySG}\text{->Mark})+(\text{Safing}\text{->Mark})+(\text{SafingSG}\text{->Mark}))$
Activation Predicate	(none)
Reactivation Predicate	(none)
Case Distributions	<div>case 1</div> <div><math>(1\text{-p\_individual-p\_ccf3of4-p\_ccf4of4})/(1\text{-p\_individual})</math></div> <div>case 2</div> <div><math>p\_ccf3of4/(1\text{-p\_individual})</math></div> <div>case 3</div> <div><math>p\_ccf4of4/(1\text{-p\_individual})</math></div>

Timed Activity:	PrimaryFailure
	Rate
Distribution Parameters	$\text{fr\_complex} * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	PrimarySGFailure
	Rate
Distribution Parameters	$\text{fr\_complex} * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	PrioritySelectorsFailure
	Rate
Distribution Parameters	$\text{fr\_simple} * \text{PrioritySelectors}\text{->Mark}()$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	SafingFailure
	Rate
Distribution Parameters	$\text{fr\_complex} * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	SafingMRM
-----------------	-----------

Rate	
Distribution Parameters	$r\_MRM$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	SafingSGFailure
	Rate
Distribution Parameters	$t_{complex} * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Instantaneous Activity:	PrimaryFailureType
Case Distributions	case 1  1-p_erratic case 2  p_erratic

Instantaneous Activity:	PrimarySGFailureType
Case Distributions	case 1  p_erratic case 2  1-p_erratic

Instantaneous Activity:	SafingFailureType
Case Distributions	case 1  1-p_erratic case 2  p_erratic

Instantaneous Activity:	SafingSGFailureType
Case Distributions	case 1  p_erratic case 2  1-p_erratic

Instantaneous Activity:	prebufferedMRM
Case Distributions	case 1  1-p_MRM case 2  p_MRM

Instantaneous Activities Without Cases:	
CatastrophicFailure	
PrimaryLayerErraticFailure	
PrimaryLayerSilentFailure	
SafingLayerErraticFailure	
SafingLayerSilentFailure	

Input Gate:	CheckCatastrophicFailure
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && (PrimaryLayerErratic->Mark()==1    PrimaryLayerSilent->Mark()+SafingLayerErratic->Mark()==2)
Function	:

Input Gate:	CheckNonCatastrophicFailure
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && (PrioritySelectors->Mark()==0    SafingLayerSilent->Mark()+PrimaryLayerSilent->Mark()==2)
Function	:

Input Gate:	CheckPrimaryLayerErratic
Predicate	PrimaryLayerSilent->Mark()+PrimaryLayerErratic->Mark()==0 && PrimaryErratic->Mark()+PrimaryFPorFN->Mark()==2
Function	:

Input Gate:	CheckPrimaryLayerSilent
Predicate	PrimaryLayerSilent->Mark()+PrimaryLayerErratic->Mark()==0 && (PrimarySGSilent->Mark()+PrimarySilent->Mark())>=1    PrimaryErratic->Mark()+PrimarySG->Mark()==2    Primary->Mark()+PrimaryFPorFN->Mark()==2)
Function	:

Input Gate:	CheckSafingLayerErratic
Predicate	SafingLayerSilent->Mark()+SafingLayerErratic->Mark()==0 && SafingErratic->Mark()+SafingFPorFN->Mark()==2
Function	:

Input Gate:	CheckSafingLayerSilent
Predicate	SafingLayerSilent->Mark()+SafingLayerErratic->Mark()==0 && (SafingSGSilent->Mark()+SafingSilent->Mark())>=1    SafingErratic->Mark()+SafingSG->Mark()==2    Safing->Mark()+SafingFPorFN->Mark()==2)
Function	:

Input Gate:	CheckSafingMRM
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && Safing->Mark()+SafingSG->Mark()==2 && PrimaryLayerSilent->Mark()==1
Function	:

Output Gate:	CCF2of4

	<pre>int a = (Primary-&gt;Mark()) + Safing-&gt;Mark() == 2); int b = (PrimarySG-&gt;Mark() + SafingSG-&gt;Mark() == 2); int c = (PrimarySG-&gt;Mark() + Safing-&gt;Mark() == 2); int d = (Primary-&gt;Mark()) + SafingSG-&gt;Mark() == 2); int g = (Safing-&gt;Mark() + SafingSG-&gt;Mark() == 2); int f = (Primary-&gt;Mark() + PrimarySG-&gt;Mark() == 2); int n = a + b + c + d + g + f; int e = 6; if (n) {     int r = rand() % n;     if (a &amp;&amp; r-- == 0) e = 0;     else if (b &amp;&amp; r-- == 0) e = 1;     else if (c &amp;&amp; r-- == 0) e = 2;     else if (d &amp;&amp; r-- == 0) e = 3;     else if (g &amp;&amp; r-- == 0) e = 4;     else if (f &amp;&amp; r-- == 0) e = 5; } if (e==0) { Primary-&gt;Mark()=0; Safing-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; SafingFailed-&gt;Mark()=1; } else if (e==1) { PrimarySG-&gt;Mark()=0; SafingSG-&gt;Mark()=0; PrimarySGFailed-&gt;Mark()=1; SafingSGFailed-&gt;Mark()=1; } else if (e==2) { PrimarySG-&gt;Mark()=0; Safing-&gt;Mark()=0; PrimarySGFailed-&gt;Mark()=1; SafingFailed-&gt;Mark()=1; } else if (e==3) { Primary-&gt;Mark()=0; SafingSG-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; SafingSGFailed-&gt;Mark()=1; } else if (e==4) { Safing-&gt;Mark()=0; SafingSG-&gt;Mark()=0; SafingFailed-&gt;Mark()=1; SafingSGFailed-&gt;Mark()=1; } else if (e==5) { Primary-&gt;Mark()=0; PrimarySG-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; PrimarySGFailed-&gt;Mark()=1; }</pre>
Function	

Output Gate:	CCF3of4
Function	<pre>int a = (Primary-&gt;Mark()) + PrimarySG-&gt;Mark() + Safing-&gt;Mark() == 3); int b = (Primary-&gt;Mark()) + PrimarySG-&gt;Mark() + SafingSG-&gt;Mark() == 3); int c = (Primary-&gt;Mark() + Safing-&gt;Mark() + SafingSG-&gt;Mark() == 3); int d = (PrimarySG-&gt;Mark() + Safing-&gt;Mark() + SafingSG-&gt;Mark() == 3); int n = a + b + c + d; int e = 4; if (n) {     int r = rand() % n;     if (a &amp;&amp; r-- == 0) e = 0;     else if (b &amp;&amp; r-- == 0) e = 1;     else if (c &amp;&amp; r-- == 0) e = 2;     else if (d &amp;&amp; r-- == 0) e = 3; } if (e==0) { Primary-&gt;Mark()=0; PrimarySG-&gt;Mark()=0; Safing-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; PrimarySGFailed-&gt;Mark()=1; SafingFailed-&gt;Mark()=1; } else if (e==1) { PrimarySG-&gt;Mark()=0; PrimarySG-&gt;Mark()=0; PrimarySGFailed-&gt;Mark()=1; PrimarySGFailed-&gt;Mark()=1; SafingFailed-&gt;Mark()=1; } else if (e==2) { Primary-&gt;Mark()=0; SafingSG-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; SafingSGFailed-&gt;Mark()=1; } else if (e==3) { Safing-&gt;Mark()=0; SafingSG-&gt;Mark()=0; SafingFailed-&gt;Mark()=1; SafingSGFailed-&gt;Mark()=1; } else if (e==4) { Primary-&gt;Mark()=0; PrimarySG-&gt;Mark()=0; PrimaryFailed-&gt;Mark()=1; PrimarySGFailed-&gt;Mark()=1; SafingFailed-&gt;Mark()=1; }</pre>

Output Gate:	CCF4of4
Function	<pre>if (PrimarySG-&gt;Mark()+SafingSG-&gt;Mark()+Primary-&gt;Mark()+Safing-&gt;Mark()==4) {     PrimarySG-&gt;Mark()=0;     SafingSG-&gt;Mark()=0;     Primary-&gt;Mark()=0;     Safing-&gt;Mark()=0;     PrimarySGFailed-&gt;Mark()=1;     SafingSGFailed-&gt;Mark()=1;     PrimaryFailed-&gt;Mark()=1;     SafingFailed-&gt;Mark()=1; }</pre>

Output Gate:	PrimaryNonSilent
Function	<pre>if (PrimarySilent-&gt;Mark()==1) {     PrimarySG-&gt;Mark()=1; } else {     PrimaryFPorFN-&gt;Mark()=1; }</pre>

Output Gate:	SafingNonSilent
Function	<pre>if (SafingSilent-&gt;Mark()==1) {     SafingSG-&gt;Mark()=1; } else {     SafingFPorFN-&gt;Mark()=1; }</pre>

Range Study Variable Assignments for Study LDCFParameter in Project LDCF :							
Variable	Type	Range Type	Range	Increment	Increment Type	Function	n
fr_complex	double	Fixed	1.0E-5	-	-	-	-
fr_simple	double	Fixed	1.0E-6	-	-	-	-
p_MRM	double	Manual	[0.75, 0.85, 0.95]	-	-	-	-
p_ccf3of4	double	Fixed	0.015	-	-	-	-
p_ccf4of4	double	Fixed	0.01	-	-	-	-
p_erratic	double	Manual	[0.1, 0.3, 0.5]	-	-	-	-
p_individual	double	Manual	[0.8, 0.875, 0.95]	-	-	-	-
r_MRM	double	Fixed	6.0	-	-	-	-

Performance Variable Model: LDCFReward	
Top Level Model Information	Child Model Name  LDCFModel  Model Type  SAN Model

Performance Variable : p_safestate		
Affecting Models	LDCFModel	
Impulse Functions		
Reward Function	(Reward is over all Available Models) if (LDCFModel->SafeState->Mark()==1) return 1;	
Simulator Statistics	Type	Instant of Time
	Options	Estimate Mean
		Include Lower Bound on Interval Estimate
		Include Upper Bound on Interval Estimate
		Estimate out of Range Probabilities
	Confidence Level is Relative	
	Parameters	Start Time 5000.0,15000.0,25000.0,35000.0, Confidence Level 0.95 Confidence Interval 0.1

Performance Variable : p_unsafestate		
Affecting Models	LDCFModel	
Impulse Functions		
Reward Function	(Reward is over all Available Models) if (LDCFModel->UnsafeState->Mark()==1) return 1;	
Simulator Statistics	Type	Instant of Time
	Options	Estimate Mean
		Include Lower Bound on Interval Estimate
		Include Upper Bound on Interval Estimate
		Estimate out of Range Probabilities
	Confidence Level is Relative	
	Parameters	Start Time 5000.0,15000.0,25000.0,35000.0, Confidence Level 0.95 Confidence Interval 0.1

Performance Variable : p_safestate_steadystate		
Affecting Models	LDCFModel	
Impulse Functions		
Reward Function	(Reward is over all Available Models) if (LDCFModel->SafeState->Mark()==1) return 1;	
Simulator Statistics	Type	Steady State
	Options	Estimate Mean
		Include Lower Bound on Interval Estimate
		Include Upper Bound on Interval Estimate
		Estimate out of Range Probabilities
	Confidence Level is Relative	
	Initial Transient	0.0

	Parameters		
	Batch Size		1.0
	Confidence	Confidence Level	0.95
		Confidence Interval	0.1

Performance Variable : p_unsafestate_steadystate			
Affecting Models	LDCFModel		
Impulse Functions			
Reward Function	(Reward is over all Available Models)		
	if (LDCFModel->UnsafeState->Mark())==1) return 1;		
Simulator Statistics	Type	Steady State	
	Options	Estimate Mean	
		Include Lower Bound on Interval Estimate	
		Include Upper Bound on Interval Estimate	
		Estimate out of Range Probabilities	
		Confidence Level is Relative	
	Parameters	Initial Transient	
		Batch Size	
	Confidence	Confidence Level	
		Confidence Interval	