

Place Attributes:

Place Names	Initial Markings
Channel1	1
Channel1Erratic	0
Channel1Failed	0
Channel1Silent	0
Channel2a	1
Channel2aFailed	0
Channel2aSilent	0
Channel2b	1
Channel2bErratic	0
Channel2bFailed	0
Channel2bSilent	0
ErraticEnvelope	0
SafeState	0
Selectors	2
UnsafeState	0

Timed Activity:	CCF
	Rate
Distribution Parameters	$fr\_complex * (1-p\_individual)$
Activation Predicate	(none)
Reactivation Predicate	(none)
Case Distributions	case 1 $(1-p\_individual-p\_ccf3of3)/(1-p\_individual)$ case 2 $p\_ccf3of3/(1-p\_individual)$

Timed Activity:	Channel1Failure
	Rate
Distribution Parameters	$fr\_complex * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	Channel1MRM
	Rate
Distribution Parameters	$r\_MRM$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	Channel2aFailure
	Rate
Distribution Parameters	$fr\_complex * p\_individual$
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	Channel2bFailure
	Rate
Distribution Parameters	$fr\_complex * p\_individual$

Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	Channel2bMRM
Distribution Parameters	Rate r_MRM
Activation Predicate	(none)
Reactivation Predicate	(none)

Timed Activity:	SelectorsFailure
Distribution Parameters	Rate fr_simple * Selectors->Mark()
Activation Predicate	(none)
Reactivation Predicate	(none)

Instantaneous Activity:	Channel1FailureType
Case Distributions	case 1 1-p_erratic case 2 p_erratic

Instantaneous Activity:	Channel2aFailureType
Case Distributions	case 1 1-p_erratic case 2 p_erratic

Instantaneous Activity:	Channel2bFailureType
Case Distributions	case 1 1-p_erratic case 2 p_erratic

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

Instantaneous Activities Without Cases:
CatastrophicFailure

Input Gate:	CheckCatastrophicFailure
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && (Channel1Erratic->Mark()+ErraticEnvelope->Mark())==2    (Channel2bErratic->Mark())==1 && (Channel1Silent->Mark()+Channel2a->Mark())==2    Channel1Erratic->Mark()+Channel2a->Mark()==2    Channel1->Mark()+ErraticEnvelope->Mark()==2))
Function	;

Input Gate:	CheckChannel1MRM
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && Channel1->Mark()+Channel2a->Mark()+Channel2bSilent->Mark()==3
Function	;

Input Gate:	CheckChannel2bMRM
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && Channel2b->Mark()==1 && (Channel1Silent->Mark()+Channel2a->Mark())==2    Channel1->Mark()+ErraticEnvelope->Mark()==2    Channel1Erratic->Mark()+Channel2a->Mark()==2)
Function	;

Input Gate:	CheckNonCatastrophicFailure
Predicate	SafeState->Mark()+UnsafeState->Mark()==0 && (Selectors->Mark())==0    Channel2aSilent->Mark()==1    (Channel2bSilent->Mark())==1 && (Channel1->Mark()+ErraticEnvelope->Mark())==2    Channel1Silent->Mark()+Channel2a->Mark()==2    Channel1Erratic->Mark()+Channel2a->Mark()==2))
Function	;

Output Gate:	CCF2of3
Function	int a = (Channel1->Mark()) + Channel2a->Mark() == 2); int b = (Channel1->Mark()) + Channel2b->Mark() == 2); int c = (Channel2a->Mark()) + Channel2b->Mark() == 2); int n = a + b + c; int e = 3; if (n) { int r = rand() % n; if (a && r-- == 0) e = 0; else if (b && r-- == 0) e = 1; else if (c && r-- == 0) e = 2; } if (e==0) { Channel1->Mark(); Channel2a->Mark(); Channel1Failed->Mark(); Channel2aFailed->Mark(); } else if (e==1) { Channel1->Mark(); Channel2b->Mark(); Channel1Failed->Mark(); Channel2bFailed->Mark(); } else if (e==2) { Channel2a->Mark(); Channel2b->Mark(); Channel2aFailed->Mark(); Channel2bFailed->Mark(); };

Output Gate:	CCF3of3

Function	<pre> if (Channel1-&gt;Mark()+Channel2a-&gt;Mark()+Channel2b-&gt;Mark())==3) { Channel1-&gt;Mark()=0; Channel2a-&gt;Mark()=0; Channel2b-&gt;Mark()=0; Channel1Failed-&gt;Mark()=1; Channel2aFailed-&gt;Mark()=1; Channel2bFailed-&gt;Mark()=1; } </pre>
----------	---

Output Gate:	Channel1NonSilent
Function	<pre> if (Channel1Silent-&gt;Mark())==1) { Channel2a-&gt;Mark()=1; } else { ErraticEnvelope-&gt;Mark()=1; } </pre>

Range Study Variable Assignments for Study **ADEYE**Parameter in Project **ADEYE** :

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_ccf3of3	double	Fixed	0.025	-	-	-	-
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: ADEYEReward		
Top Level Model Information	Child Model Name	ADEYEModel
	Model Type	SAN Model

Performance Variable : p_safestate		
Affecting Models	ADEYEModel	
Impulse Functions		
Reward Function	<i>(Reward is over all Available Models)</i> if (ADEYEModel->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	Confidence Level 0.95
		Confidence Interval 0.1

Performance Variable : p_unsafestate		
Affecting Models	ADEYEModel	
Impulse Functions		
Reward Function	<i>(Reward is over all Available Models)</i> if (ADEYEModel->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	Confidence Level 0.95
		Confidence Interval 0.1

Performance Variable : p_safestate_steadystate		
Affecting Models	ADEYEModel	
Impulse Functions		
Reward Function	<i>(Reward is over all Available Models)</i> if (ADEYEModel->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
	Parameters	Initial Transient 0.0
		Batch Size 1.0
Confidence	Confidence Level	0.95
	Confidence Interval	0.1

Performance Variable : p_unsafestate_steadystate		
Affecting Models	ADEYEModel	
Impulse Functions		
Reward Function	<i>(Reward is over all Available Models)</i> if (ADEYEModel->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 0.0
		Batch Size 1.0
	Confidence Level	0.95

	Confidence	
	Confidence Interval	0.1