

Cyber Physical Systems

Verification of Train System with Electronically-controlled brake in PRISM

Project Report

**ELECTRICAL
AND
COMPUTER
ENGINEERING
COLLEGE OF
ENGINEERING**



Submitted By:

Bhuvan Jain (UIN: 667704103)

TABLE OF CONTENTS

List of Figures.....	2
List of Tables.....	2
1. Introduction.....	3
1.1 PRISM Overview.....	4
1.2 Problem statement	4
2. Methodology.....	5
2.1. Description of Model.....	5
2.1.1 Velocity Subsystem.....	5
2.1.2 Braking Subsystem.....	6
3. Implementation in PRISM	7
3.1. Noise value Quantization.....	8
4. Discussion.....	10
5. Future Work.....	11
6. Results	12
6.1 Model Verification.....	12
6.2 Simulation.....	14
7. Reference.....	16
8. Appendix	17
5.1 MATLAB code.....	17
5.2 PRISM Code.....	18

LIST OF FIGURES

FIG 1. – Velocity Subsystem	5
FIG 2. – Braking Subsystem.....	6
FIG 3. – PDF for noise $n_1 \sim N(0,1)$	8
FIG 4. – CDF for noise $n_1 \sim N(0,1)$	8

LIST OF Tables

Table 1. – Calculation for Quantized value of $n_1 \sim N(0,1)$	9
---	---

1.INTRODUCTION

1.1 PRISM Overview:

PRISM is a probabilistic model checker, a tool for formal modelling and analysis of systems that exhibit random or probabilistic behavior. It has been used to analyze systems from many different application domains, including communication and multimedia protocols, randomized distributed algorithms, security protocols, biological systems and many others.

PRISM can build and analyze several types of probabilistic models:

- discrete-time Markov chains (DTMCs)
- continuous-time Markov chains (CTMCs)
- Markov decision processes (MDPs)
- probabilistic automata (PAs)
- probabilistic timed automata (PTAs)



Models are described using the PRISM language, a simple, state-based language. PRISM provides support for automated analysis of a wide range of quantitative properties of these models, e.g. "what is the probability of a failure causing the system to shut down within 4 hours?", "what is the worst-case probability of the protocol terminating in error, over all possible initial configurations?", "what is the expected size of the message queue after 30 minutes?", or "what is the worst-case expected time taken for the algorithm to terminate?". The property specification language incorporates the temporal logics PCTL, CSL, LTL and PCTL*, as well as extensions for quantitative specifications and costs/rewards.

PRISM incorporates state-of-the-art symbolic data structures and algorithms, based on BDDs (Binary Decision Diagrams) and MTBDDs (Multi-Terminal Binary Decision Diagrams). It also includes a discrete-event simulation engine, providing support for approximate/statistical model checking, and implementations of various analysis techniques, such as quantitative abstraction refinement and symmetry reduction.

PRISM is free and open source, released under the GNU General Public License (GPL).

1.2 Problem Statement:

Quantize the model of a train with electronically-controlled brakes as described in [1] and to verify the system using the PRISM model checker.

2. METHODOLOGY

2.1 Description of Model:

- Train system having **5 cars**. Each having their own braking system.
- Model Type- **Stochastic**.

2.1.1 Velocity subsystem (first module for PRISM):

- The train starts in the discrete state $q_v = 1$ and remains there until the velocity exceeds a threshold $V_U = 28.5$, switches to the discrete state $q_v = 2$.
- The train remains in the state $q_v = 2$ until one of the brakes engages and it switches to state $q_v = 3$.
- The velocity in states $q_v = 3$ depends on the number of brakes that have been engaged through the braking force term, where F_j is the braking force of car j .
- When all the brakes disengage, the velocity system switches back to the state $q_v = 1$. When in the state $q_v = 1$, the train accelerates to a constant velocity of $V_C = 25$ and oscillates around it with the amplitude 2.5.

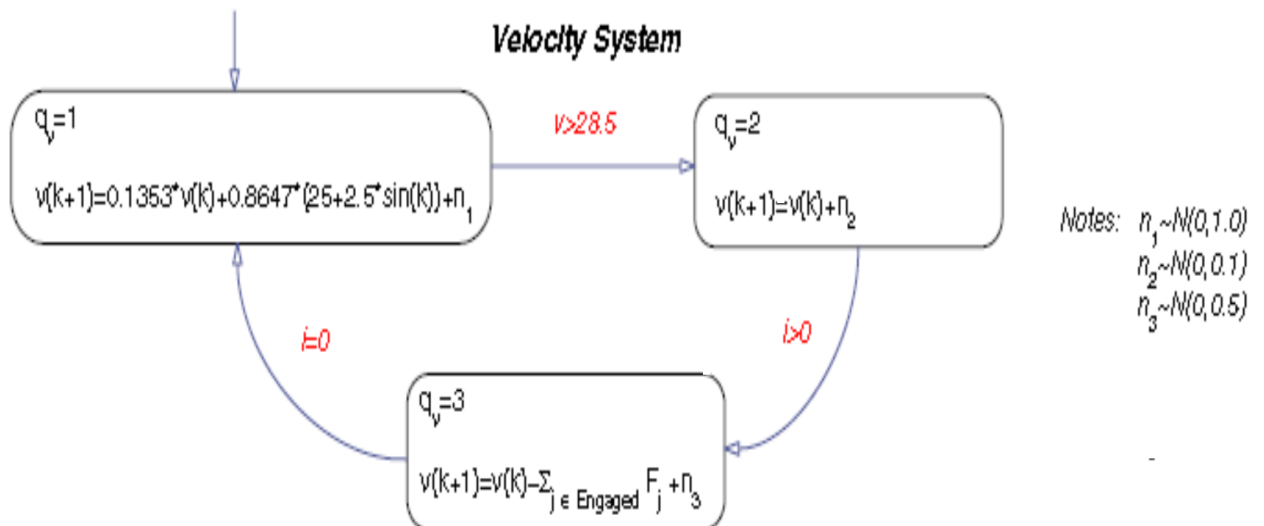


Fig 1. Velocity Subsystem

2.1.2 Braking subsystem for each car (second module for PRISM):

- The train starts in the discrete state $q_b = 1$ and remains there until the velocity exceeds a threshold $V_U = 28.5$, switches to $q_b = 2$.
- Initial value of the timer c_1 in the state $q_b = 2$ is not deterministic, so the duration of time the system remains in $q_b = 2$ is a random variable (until $c_1 > 1$).
- Braking system can fail with a probability $p = 0.1$ and then permanently switch to $q_b = 3$. With the probability $p = 0.9$ it either returns to state $q_b = 1$ if velocity fall below the threshold $V_L = 20$, or switches to $q_b = 4$ and engages in braking sequence otherwise.
- When the brake engages the variable i is increased by 1, thereby affecting the velocity of the train as per velocity module.
- When the velocity falls below $V_L = 20$, the brake disengages after a random amount of time (modeled by the timer c_2 in the state $q_b = 5$), when it switches to the state $q_b = 1$.

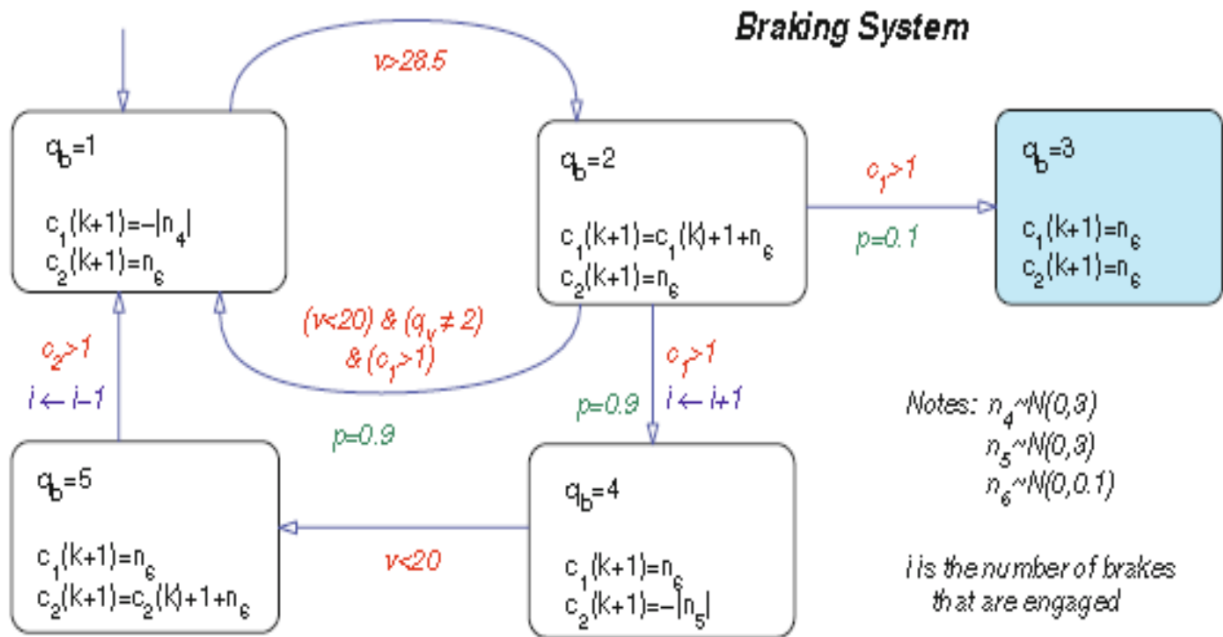


Fig 2. Braking Subsystem

3. IMPLEMENTATION IN PRISM

As can be seen that in the velocity module for the state $q_v = 1$ the velocity equation is given as follows $v(k+1) = 0.1353 * v(k) + 0.8647 * (25 + 2.5 * \sin(k)) + n_1$, where the velocity is discretized at every point k and n_1 is the noise which is the normal gaussian noise represented as $n_1 \sim N(0,1)$. For quantization of velocity value kept the quantization constant $\delta=1$ and define a global variable k in the PRISM from 1 to 500 (also act as global timestamp / number of iteration).

global k : [0..500] init 0;

For the function of sin used in the velocity formula which is generated as a lookup function using ? operator as PRISM doesn't allow to define a procedural function and is defined as: $(condition)?(value_returned_if_true):(value_returned_if_false)$; so it returns -0.76 whenever $k=4$ for the line written as $(k=4)?(-0.76:4)$. The system will pick the lowest value and never the value 4 as the actual response 4 is out of bound.

**formula sin =min [(k=0?0:4),
 (k=1?0.84:4),
 (k=2?0.91:4),
 :
 (k=500?-0.47:4)];**

Now state $q_v = 1$ is written in PRISM is written as:

[] $q_v=1 \rightarrow (v_k1' = \text{ceil}(0.1353 * v_k1 + (0.8647 * (25 + 2.5 * \sin)) + n_1)) \& (k' = \min(k+1, 500));$



Transition from $q_v=1$ to $q_v=2$ for $(V > V_U = 28.5)$ written as:

[] $q_v=1 \& v_k1 \geq 27 \rightarrow 1: (q_v'=2);$

Similarly, all the states in the velocity and braking subsystem is written in the PRISM as what is represented above using the guard and update as per required by the different states and their transitions.

3.1 Noise Value Quantization:

As the model contains noise at every state which is the Normal Gaussian noise with a zero mean and some variance, this affect the sensor readings for velocity and brake and need to be quantized at every point before adding it in our PRISM.

This can be done by approximating a continuous normal random variable w (noise here) with the density $N(0, \sigma)$ and the cumulative distribution $F(\tau)$ by using a discrete random variable $\hat{w}=k\delta$ (discrete noise that is added finally) so that

$$P[\hat{w} = k\delta] = \begin{cases} F((k + \frac{1}{2})\delta) - F((k - \frac{1}{2})\delta) & -M < k < M \\ F((k + \frac{1}{2})\delta) & k = -M \\ 1 - F((k - \frac{1}{2})\delta) & k = M \\ 0 & \text{otherwise} \end{cases}$$

Where $M \in \mathbb{N}$ defines the interval of interest for the random variable and calculated as $M\delta = 3\sigma$. Here k is an integer ranging from $-M$ to M .

The above equation is implemented in the MTLAB to get the values of k and M for a given σ and assuming δ accordingly. The code to generate that is given in the appendix section of this report **[see appendix for the code]**. The probability density function and the cumulative density functions are also obtained for the noise value $n_1 \sim N(0,1)$.

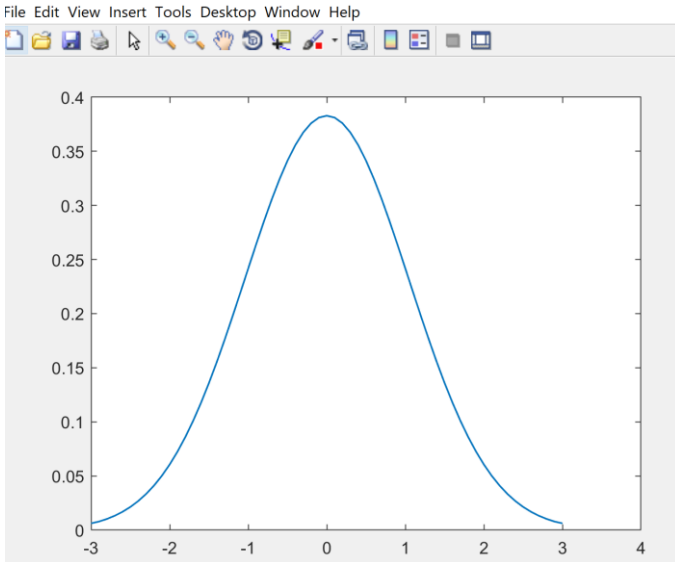


Fig 3. PDF for noise $n_1 \sim N(0,1)$

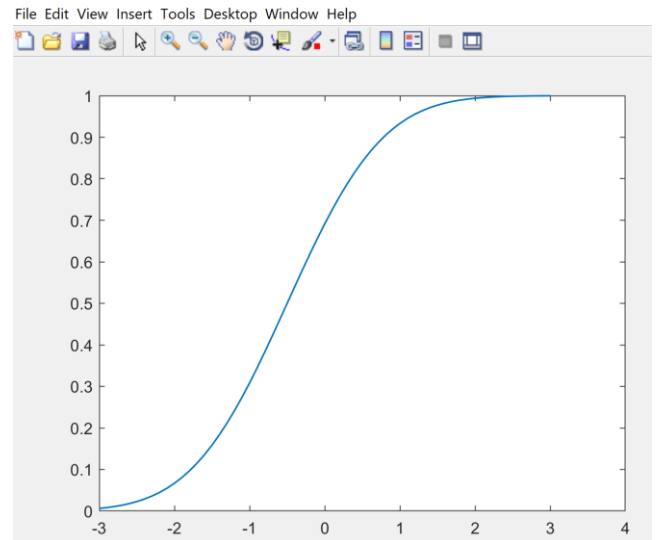


Fig 4. CDF for noise $n_1 \sim N(0,1)$

For quantization of $n_1 \sim N(0,1)$, having $\mu=0$, $\sigma=1$ and by assuming $\delta=1$.

We get $M=3$ and $-3 < k < 3$. The probability distribution and corresponding discrete value for this noise are:

Sigma:	1	Delta:	1	M:	3
k Ranges from -M to M	Probability	W Value Discrete			
-3	0.00621	-3			
-2	0.060598	-2			
-1	0.24173	-1			
0	0.382925	0			
1	0.24173	1			
2	0.060598	2			
3	0.00621	3			
Prob Sum	1				

Table 1. Calculation for Quantized value of $n_1 \sim N(0,1)$

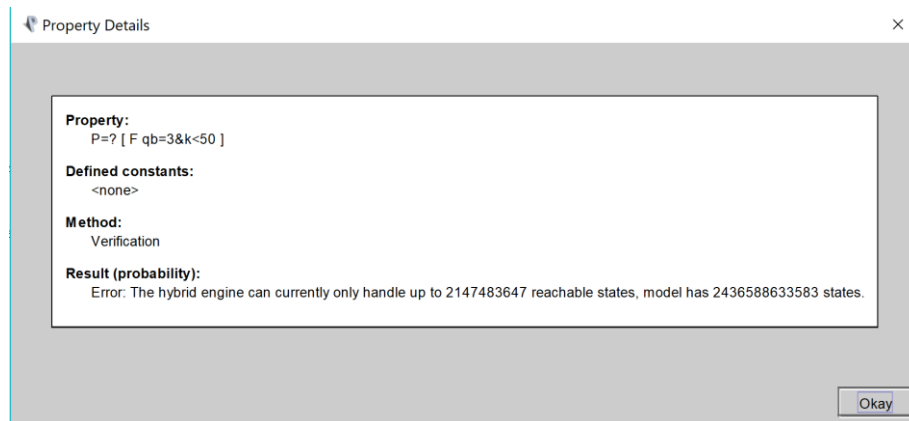
Now as noise has the discrete value the velocity equation can be written as:

```
[ ] qv=1 -> 0.00621:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-3))& (k'= min(k+1,500))
+ 0.060598:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-2))& (k'= min(k+1,500))
+ 0.24173:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-1))& (k'= min(k+1,500))
+ 0.382925:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-0))& (k'= min(k+1,500))
+ 0.00621:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+3))& (k'= min(k+1,500))
+ 0.060598:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+2))& (k'= min(k+1,500))
+ 0.24173:(v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+1))& (k'= min(k+1,500));
```

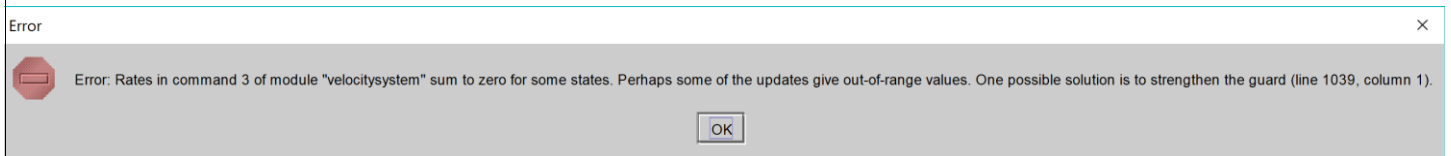
Similarly for all noises $n_2 \sim N(0,0.1)$, $n_3 \sim N(0,0.5)$, $n_4 \sim N(0,3)$, $n_5 \sim N(0,3)$, $n_6 \sim N(0,0.1)$ quantized values are calculated and then added in the PRISM velocity, braking module.

4. DISCUSSION (Problems Faced):

- 1) When tried to verify the property $P=?[F(qb=3 \ \& \ k<50)]$ on the 5 car system PRISM has generated total of $>10^{12}$ reachable states for my model and gives an error as its hybrid engine can only handle up to 10^9 states



- 2) Another problem arise which is shown as follows:



- Both problems are resolved as I have merged my velocity and braking model together such that the number of states and their transitions could be reduced.
 - I have strengthened my guard (the left-hand side states in the code) by keeping both the velocity (q_v) and braking (q_b) states together whenever there is a possibility of simultaneous transitions and upgradation for both states. It also implicitly removed the error for “updates give out of range value”.
- 3) PRISM doesn't have inbuilt function such as $\sin(x)$ so need to implement a look up function for 500 values in a trivial manner as discussed earlier in the Implementation in PRISM part.
 - 4) Also, PRISM during verification of properties takes couple of hours to do just 50 iterations, which was very time consuming.

5. FUTURE WORK:

- 1) Instead of taking only the brakes into account also take the sensor readings for stress on each car and try to implement that model too.
- 2) Compute the different value of quantization constant δ , M and study what effect it has on the model.

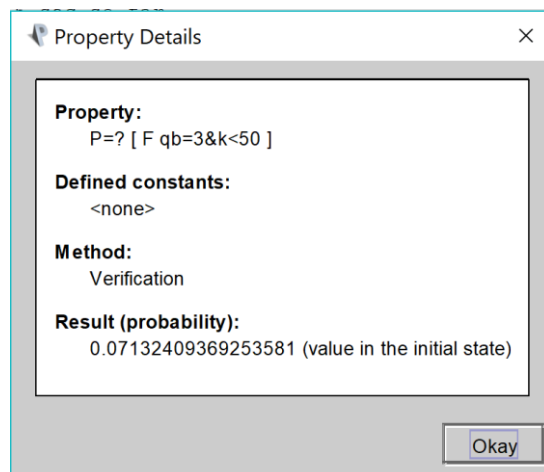
6. RESULTS:

6.1 Model Verification:

1). Initially a simple model with only 3 cars was verified using a simple property:

$P=?[F(qb=3 \ \& \ k<50)]$ where F is the eventually (in near future) operator

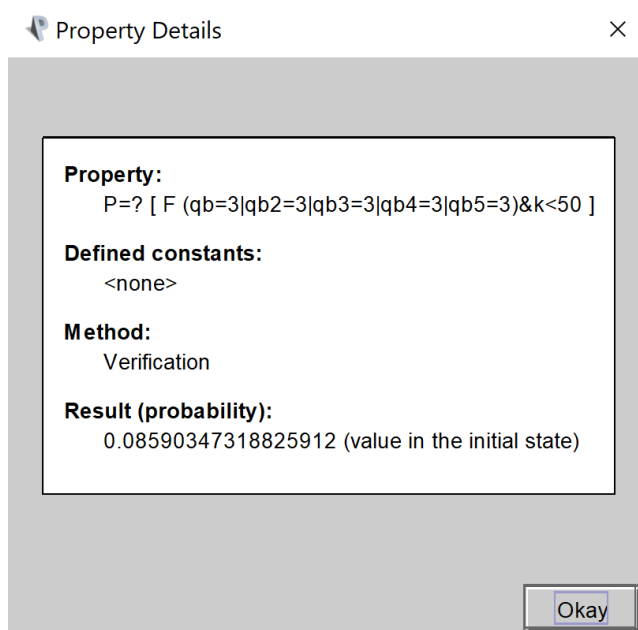
(What is the probability that car 1 brake will fail in less than 50 iterations)



2). Full implementation of 5 cars was verified using property:

$P=? [F(qb=3 \ | \ qb2=3 \ | \ qb3=3 \ | \ qb4=3 \ | \ qb5=3) \ \& \ k<50]$

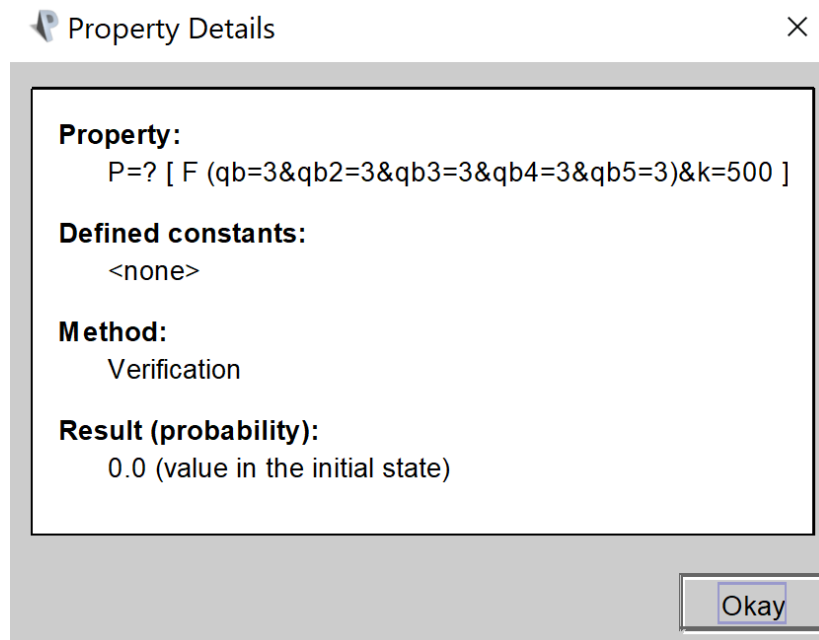
(What is the probability that either car 1 or 2 or 3 or 4 or 5 brake will fail in less than 50 iterations)



3). **Full implementation of 5 cars** was verified using property:

P=? [F (qb=3 &qb2=3 &qb3=3 &qb4=3 &qb5=3) & k=500]

(What is the probability that all 5 cars brake will fail in 500 iterations)



6.2 Simulation:

PRISM 4.4

File Edit Model Properties Simulator Log Options

Automatic exploration: Simulate Steps: 100 Backtracking: Backtrack Steps: 1

Manual exploration:

Module/action	Rate	Update
velocitysystem	0.00621	v_k1=28, k=79, c1_j
velocitysystem	0.060598	v_k1=28, k=79, c1_j
velocitysystem	0.24173	v_k1=28, k=79, c1_j
velocitysystem	0.382925	v_k1=28, k=79, c1_j
velocitysystem	0.00621	v_k1=29, k=79, c1_j

State labels: init, deadlock

Path:

Step	Time	i	k	v_k1	qv	qb	c1_k1	c2_k1	qb2	c1_k2	c2_k2	qb3	c1_k3	c2_k3	qb4	c1_k4	c2_k4	qb5	c1_k5	c2_k5
19	1.25014	0	14	28	2	2	2	0	2	2	1	2	0	0	2	-2	0	2	-9	0
20	1.4135	0	14	28	2	2	2	0	3	2	1	2	0	0	2	-2	0	2	-9	0
21	1.49785	0	14	28	2	2	2	0	3	2	1	2	0	0	2	-2	0	2	-9	0
22	1.56139	1	14	28	2	4	2	0	3	2	1	2	0	0	2	-2	0	2	-9	0
23	1.69927	1	15	28	2	4	2	0	3	0	0	2	0	0	2	-2	0	2	-9	0
24	1.88647	1	16	28	2	4	2	0	3	0	0	2	0	0	2	-1	0	2	-9	0
25	1.94905	1	17	28	2	4	2	0	3	0	0	2	0	0	2	-1	0	2	-9	0
26	1.95286	1	18	28	2	4	2	0	3	0	0	2	1	0	2	-1	0	2	-9	0
27	1.95628	1	19	28	2	4	2	0	3	0	0	2	2	0	2	-1	0	2	-9	0
28	2.13558	2	19	28	2	4	2	0	3	0	0	4	2	0	2	-1	0	2	-9	0
29	2.75193	2	20	29	2	4	2	0	3	0	0	4	2	0	2	-1	0	2	-7	1
30	3.03807	2	21	29	2	4	2	0	3	0	0	4	2	0	2	-1	0	2	-6	0
31	4.32518	2	22	30	2	4	2	0	3	0	0	4	2	0	2	-1	0	2	-4	1
32	4.37738	2	23	30	2	4	2	0	3	0	0	4	2	0	2	0	0	2	-4	1
33	4.58432	2	24	30	2	4	2	0	3	0	0	4	2	0	2	1	0	2	-4	1
34	4.83032	2	25	30	2	4	2	0	3	0	0	4	2	0	2	1	0	2	-4	1
35	5.50354	2	26	31	2	4	2	0	3	0	0	4	2	0	2	3	1	2	-4	1
36	5.7214	2	27	31	2	4	2	0	3	0	0	4	2	0	2	3	1	2	-4	1
37	5.73406	3	27	31	2	4	2	0	3	0	0	4	2	0	4	3	1	2	-4	1
38	6.61212	3	28	31	2	4	2	0	3	1	1	4	2	0	4	3	1	2	-4	1
39	6.96309	3	29	31	2	4	2	0	3	1	1	4	2	0	4	3	1	2	-3	0

Model Properties Simulator Log

Saving model... done

Type here to search

ENG 05:01 PM 10-05-2018

PRISM 4.4

File Edit Model Properties Simulator Log Options

Automatic exploration: Simulate Steps: 100 Backtracking: Backtrack Steps: 1

Manual exploration:

Module/action	Rate	Update
velocitysystem	0.00621	v_k1=28, k=79, c1_j
velocitysystem	0.060598	v_k1=28, k=79, c1_j
velocitysystem	0.24173	v_k1=28, k=79, c1_j
velocitysystem	0.382925	v_k1=28, k=79, c1_j
velocitysystem	0.00621	v_k1=29, k=79, c1_j

State labels: init, deadlock

Path:

Step	Time	i	k	v_k1	qv	qb	c1_k1	c2_k1	qb2	c1_k2	c2_k2	qb3	c1_k3	c2_k3	qb4	c1_k4	c2_k4	qb5	c1_k5	c2_k5
48	12.2326	3	38	33	2	4	2	0	3	1	1	4	2	0	4	3	1	2	3	1
49	12.9081	3	39	33	2	4	2	0	3	0	0	4	2	0	4	3	1	2	3	1
50	13.3585	4	39	33	2	4	2	0	3	0	0	4	2	0	4	3	1	4	3	1
51	13.7206	4	40	15	3	4	2	0	3	0	0	4	2	0	4	3	1	4	3	1
52	13.742	4	41	15	3	4	2	0	3	0	0	4	0	0	4	3	1	4	3	1
53	13.8838	4	41	15	3	5	2	0	3	0	0	4	0	0	4	3	1	4	3	1
54	14.0388	4	42	15	3	5	1	2	3	0	0	4	0	0	4	3	1	4	3	1
55	14.0834	3	42	15	3	1	1	2	3	0	0	4	0	0	4	3	1	4	3	1
56	14.0958	3	42	15	3	1	1	2	3	0	0	5	0	0	4	3	1	4	3	1
57	14.1316	3	43	15	3	1	1	2	3	1	1	5	0	0	4	3	1	4	3	1
58	14.5563	3	43	15	3	1	1	2	3	1	1	5	0	0	5	3	1	4	3	1
59	14.5773	3	44	15	3	1	1	2	3	1	1	5	0	1	5	3	1	4	3	1
60	15.3121	3	45	15	3	1	1	2	3	1	1	5	0	1	5	1	3	4	3	1
61	15.4782	3	46	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
62	15.6357	3	47	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
63	15.7644	3	48	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
64	15.9489	3	48	15	3	1	1	2	3	1	1	5	0	2	5	1	3	5	3	1
65	16.452	2	48	15	3	1	1	2	3	1	1	1	0	2	5	1	3	5	3	1
66	16.4632	2	49	15	3	1	-3	0	3	1	1	1	0	2	5	1	3	5	3	1
67	16.4793	1	49	15	3	1	-3	0	3	1	1	1	0	2	1	1	3	5	3	1

Model Properties Simulator Log

Saving model... done

Type here to search

ENG 05:01 PM 10-05-2018

PRISM 4.4

File Edit Model Properties Simulator Log Options

Automatic exploration: Simulate Steps 100 Backtracking Backtrack Steps 1

Manual exploration: Module/action Rate Update

Module/action	Rate	Update
velocitysystem	0.00621	v_k1=28, k=79, c1_j
velocitysystem	0.060598	v_k1=28, k=79, c1_j
velocitysystem	0.24173	v_k1=28, k=79, c1_j
velocitysystem	0.382925	v_k1=28, k=79, c1_j
velocitysystem	0.00621	v_k1=28, k=79, c1_j

Generate time automatically

State labels: init, deadlock

Path

Step	Time	Action	#	l	k	v_k1	qv	qb	c1_k1	c2_k1	qb2	c1_k2	c2_k2	qb3	c1_k3	c2_k3	qb4	c1_k4	c2_k4	qb5	c1_k5	c2_k5
0	0			0	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
1	0.167244	velocitysystem	1	0	1	23	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
2	0.171489	velocitysystem	2	0	2	28	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
3	0.278785	velocitysystem	3	0	3	28	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
4	0.317868	velocitysystem	4	0	4	25	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
5	0.383127	velocitysystem	5	0	5	25	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
6	0.51343	velocitysystem	6	0	6	25	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
7	0.541718	velocitysystem	7	0	7	27	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
8	0.549599	velocitysystem	8	0	8	27	1	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
9	0.550597	velocitysystem	9	0	8	27	2	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
10	0.565289	velocitysystem	10	0	8	27	2	1	0	0	1	0	0	2	0	0	1	0	0	1	0	0
11	0.598945	velocitysystem	11	0	9	27	2	1	0	0	1	0	0	2	0	0	1	0	0	1	0	0
12	0.621438	velocitysystem	12	0	9	27	2	1	0	0	1	0	0	2	0	0	2	0	0	1	0	0
13	0.696994	velocitysystem	13	0	9	27	2	2	0	0	1	0	0	2	0	0	2	0	0	1	0	0
14	0.755907	velocitysystem	14	0	10	27	2	2	0	0	1	0	0	2	0	0	2	0	0	1	0	0
15	0.822732	velocitysystem	15	0	11	27	2	2	1	0	1	0	0	2	0	0	2	0	0	1	0	0
16	0.825292	velocitysystem	16	0	12	27	2	2	1	0	1	0	0	2	0	0	2	0	0	1	0	0
17	0.965225	velocitysystem	17	0	12	27	2	2	1	0	2	0	0	2	0	0	2	0	0	1	0	0
18	1.12235	velocitysystem	18	0	13	27	2	2	2	0	2	0	0	2	0	0	2	0	0	1	0	0
19	1.23014	velocitysystem	19	0	14	28	2	2	2	0	2	2	1	2	0	0	2	0	0	1	0	0

Model Properties Simulator Log

Saving model... done

Type here to search

05:00 PM 10-05-2018

PRISM 4.4

File Edit Model Properties Simulator Log Options

Automatic exploration: Simulate Steps 100 Backtracking Backtrack Steps 1

Manual exploration: Module/action Rate Update

Module/action	Rate	Update
velocitysystem	0.00621	v_k1=28, k=79, c1_j
velocitysystem	0.060598	v_k1=28, k=79, c1_j
velocitysystem	0.24173	v_k1=28, k=79, c1_j
velocitysystem	0.382925	v_k1=28, k=79, c1_j
velocitysystem	0.00621	v_k1=28, k=79, c1_j

Generate time automatically

State labels: init, deadlock

Path

Step	Time	Action	#	l	k	v_k1	qv	qb	c1_k1	c2_k1	qb2	c1_k2	c2_k2	qb3	c1_k3	c2_k3	qb4	c1_k4	c2_k4	qb5	c1_k5	c2_k5
55	14.0834	velocitysystem	55	3	42	15	3	1	1	2	3	0	0	5	0	0	4	3	1	4	3	1
56	14.0958	velocitysystem	56	3	42	15	3	1	1	2	3	0	0	5	0	0	4	3	1	4	3	1
57	14.1316	velocitysystem	57	3	43	15	3	1	1	2	3	1	1	5	0	0	4	3	1	4	3	1
58	14.5563	velocitysystem	58	3	43	15	3	1	1	2	3	1	1	5	0	0	5	3	1	4	3	1
59	14.5773	velocitysystem	59	3	44	15	3	1	1	2	3	1	1	5	0	1	5	3	1	4	3	1
60	15.3121	velocitysystem	60	3	45	15	3	1	1	2	3	1	1	5	0	1	5	1	3	4	3	1
61	15.4782	velocitysystem	61	3	46	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
62	15.6357	velocitysystem	62	3	47	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
63	15.7644	velocitysystem	63	3	48	15	3	1	1	2	3	1	1	5	0	2	5	1	3	4	3	1
64	15.9469	velocitysystem	64	3	48	15	3	1	1	2	3	1	1	5	0	2	5	1	3	5	3	1
65	16.452	velocitysystem	65	2	48	15	3	1	1	2	3	1	1	1	0	2	5	1	3	5	3	1
66	16.4632	velocitysystem	66	2	49	15	3	1	1	0	3	1	1	1	0	2	5	1	3	5	3	1
67	16.4793	velocitysystem	67	1	49	15	3	1	1	0	3	1	1	1	0	2	1	1	3	5	3	1
68	16.7398	velocitysystem	68	1	50	15	3	1	1	0	3	1	1	1	0	2	1	1	3	5	1	3
69	17.5357	velocitysystem	69	1	51	15	3	1	1	0	3	1	1	1	0	0	1	1	3	5	1	3
70	17.5528	velocitysystem	70	0	51	15	3	1	1	0	3	1	1	1	0	0	1	1	3	1	1	3
71	17.567	velocitysystem	71	0	52	15	3	1	1	0	3	1	1	1	0	0	1	1	3	1	1	3
72	17.6831	velocitysystem	72	0	52	15	3	1	1	0	3	1	1	1	0	0	1	1	3	1	1	3
73	17.7042	velocitysystem	73	0	53	28	1	1	1	0	3	1	1	1	0	0	1	1	3	1	1	3
74	17.7292	velocitysystem	74	0	54	25	1	1	1	0	3	1	1	1	0	0	1	1	3	1	1	3
75	17.7956	velocitysystem	75	0	55	25	1	1	1	0	3	1	1	1	0	0	1	0	0	1	1	3

Model Properties Simulator Log

Saving model... done

Type here to search

05:01 PM 10-05-2018

7. REFERENCES:

- [1] P. Sistla, M. Zefran, and Y. Feng, “Runtime Monitoring of Stochastic Cyber-Physical Systems with Hybrid State,” in Runtime Verification, vol. 7186 of Lecture Notes in Computer Science, pp. 276–293, Sept. 2011.
- [2] <http://www.prismmodelchecker.org/manual/ThePRISMLanguage/Introduction>

8. APPENDIX:

MATLAB Code:

For quantization of $n_1 \sim N(0,1)$, having $\mu=0$, $\sigma=1$; Let $\delta=1$

We get $M=3$. So $-3 < k < 3$

```
1- mu = 0;
2- sigma = 0.1;
3- delta= .1;
4- M= (3*sigma)/delta;
5- K = [-M:.1:M];
6- T1= (K+.5)*delta;
7- T2= (K-.5)*delta;
8- cdf1 = normcdf(T1,mu,sigma);
9- cdf2 = normcdf(T2,mu,sigma);
10 % -M <K < M
11- p1=cdf1-cdf2;
12 % K = -M
13- p2=normcdf((-M+.5)*delta,mu,sigma);
14 % K = M
15- p3=1-(normcdf((M-.5)*delta,mu,sigma));
16
17- x=[p2 p1(2:(end-1)) p3];
18- figure,
19- plot(K,cdf1,'LineWidth',1)
20- figure,
21- plot(K,x,'LineWidth',1)
```

PRISM Code:

```

1063 // Module 1
1064 module velocitysystem
1065 // qv=1-----qv=1
1066 [] qv=1 -> 0.00621: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-3)) & (k'= min(k+1,500))
1067 + 0.060598: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-2)) & (k'= min(k+1,500))
1068 + 0.24173: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-1)) & (k'= min(k+1,500))
1069 + 0.382925: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))-0)) & (k'= min(k+1,500))
1070 + 0.00621: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+3)) & (k'= min(k+1,500))
1071 + 0.060598: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+2)) & (k'= min(k+1,500))
1072 + 0.24173: (v_k1'=ceil (0.1353*v_k1+(0.8647*(25+2.5*sin))+1)) & (k'= min(k+1,500));
1073 // qb=1 ----qb=1
1074 [] qb=1 -> 0.00621: (c1_k1'=ceil(0-9)) & (c2_k1'=ceil(0-0.3)) & (k'= min(k+1,500))
1075 + 0.060598: (c1_k1'=ceil(0-6)) & (c2_k1'=ceil(0-0.2)) & (k'= min(k+1,500))
1076 + 0.24173: (c1_k1'=ceil(0-3)) & (c2_k1'=ceil(0-0.1)) & (k'= min(k+1,500))
1077 + 0.382925: (c1_k1'=ceil(0)) & (c2_k1'=ceil(0)) & (k'= min(k+1,500))
1078 + 0.00621: (c1_k1'=ceil(0-9)) & (c2_k1'=ceil(0+0.3)) & (k'= min(k+1,500))
1079 + 0.060598: (c1_k1'=ceil(0-6)) & (c2_k1'=ceil(0+0.2)) & (k'= min(k+1,500))
1080 + 0.24173: (c1_k1'=ceil(0-3)) & (c2_k1'=ceil(0+0.1)) & (k'= min(k+1,500));
1081
1082 // qv=1-----qv=2 When Velocity > Threshold Velocity
1083 // qb=1-----qb=2
1084 [] qv=1 & v_k1>=27 & qb=1 -> (qv'=2) & (qb'=2);
1085 // qv=2-----qv=2
1086 [] qv=2 & qb=2 & !brake & (c1_k1'<=1) -> 0.00621: (v_k1'=ceil(v_k1-0.3)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1-0.3)) & (c2_k1'=ceil(0-0.3))
1087 + 0.060598: (v_k1'=ceil(v_k1-0.2)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1-0.2)) & (c2_k1'=ceil(0-0.2))
1088 + 0.24173: (v_k1'=ceil(v_k1-0.1)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1-0.1)) & (c2_k1'=ceil(0-0.1))
1089 + 0.382925: (v_k1'=ceil(v_k1-0)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1-0)) & (c2_k1'=ceil(0-0))
1090 + 0.00621: (v_k1'=ceil(v_k1+0.3)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1+0.3)) & (c2_k1'=ceil(0+0.3))
1091 + 0.060598: (v_k1'=ceil(v_k1+0.2)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1+0.2)) & (c2_k1'=ceil(0+0.2))
1092 + 0.24173: (v_k1'=ceil(v_k1+0.1)) & (k'= min(k+1,500)) & (c1_k1'=ceil(c1_k1+1+0.1)) & (c2_k1'=ceil(0+0.1));
1093 // qb=2-----qb=3 (Failure), qb=4 (Engage Braking)
1094 [] qb=2 & (c1_k1'>1) -> 0.1: (qb'=3) + 0.9: (qb'=4) & (i'=i+1);
1095 // qb=3-----qb=3 (Error state-System fail)
1096 [] qb=3 -> 0.00621: (c2_k1'=ceil(0-0.3)) & (c1_k1'=ceil(0-0.3)) & (k'= min(k+1,500))
1097 + 0.060598: (c2_k1'=ceil(0-0.2)) & (c1_k1'=ceil(0-0.2)) & (k'= min(k+1,500))
1098 + 0.24173: (c2_k1'=ceil(0-0.1)) & (c1_k1'=ceil(0-0.1)) & (k'= min(k+1,500))
1099 + 0.382925: (c2_k1'=ceil(0)) & (c1_k1'=ceil(0)) & (k'= min(k+1,500))
1100 + 0.00621: (c2_k1'=ceil(0+0.3)) & (c1_k1'=ceil(0+0.3)) & (k'= min(k+1,500))

```

```

1101 + 0.060598: (c2_k1'=ceil(0+0.2)) & (c1_k1'=ceil(0+0.2)) & (k'= min(k+1,500))
1102 + 0.24173: (c2_k1'=ceil(0+0.1)) & (c1_k1'=ceil(0+0.1)) & (k'= min(k+1,500));
1103 // qb=2-----qb=1 (if velocity fall below threshold)
1104 [] qb=2 & (v_k1<23) & (qv!=2) & (c1_k1'>1) -> 0.9: (qb'=1);
1105
1106 // qv=2-----qv=3 Brakes Engaged(Braking Force )
1107 // qb=4-----qb=4
1108 [] qv=2 & qb=4 & brake & (qb>2 & qb2>2 & qb3>2 & qb4>2 & qb5>2) & (v_k1>=23) -> 0.00621: (qv'=3) & (v_k1'=ceil(v_k1-force-1.5)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-9)) & (c1_k1'=ceil(0-0.3))
1109 + 0.060598: (qv'=3) & (v_k1'=ceil(v_k1-force-1)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-6)) & (c1_k1'=ceil(0-0.2))
1110 + 0.24173: (qv'=3) & (v_k1'=ceil(v_k1-force-0.5)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-3)) & (c1_k1'=ceil(0-0.1))
1111 + 0.382925: (qv'=3) & (v_k1'=ceil(v_k1-force-0)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0)) & (c1_k1'=ceil(0))
1112 + 0.00621: (qv'=3) & (v_k1'=ceil(v_k1-force+1.5)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-9)) & (c1_k1'=ceil(0+0.3))
1113 + 0.060598: (qv'=3) & (v_k1'=ceil(v_k1-force+1)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-6)) & (c1_k1'=ceil(0+0.2))
1114 + 0.24173: (qv'=3) & (v_k1'=ceil(v_k1-force+0.5)) & (k'= min(k+1,500)) & (c2_k1'=ceil(0-3)) & (c1_k1'=ceil(0+0.1));
1115
1116 // qb=4---qb=5 (velocity reaches below threshold)
1117 [] qb=4 & (v_k1<23) -> (qb'=5);
1118 // qb=5-----qb=5 (waiting time to disengage the brakes- until counter c2>1)
1119 [] qb=5 & (c2_k1'<=1) -> 0.00621: (c2_k1'=ceil(c2_k1+1-0.3)) & (c1_k1'=ceil(0-0.3)) & (k'= min(k+1,500))
1120 + 0.060598: (c2_k1'=ceil(c2_k1+1-0.2)) & (c1_k1'=ceil(0-0.2)) & (k'= min(k+1,500))
1121 + 0.24173: (c2_k1'=ceil(c2_k1+1-0.1)) & (c1_k1'=ceil(0-0.1)) & (k'= min(k+1,500))
1122 + 0.382925: (c2_k1'=ceil(c2_k1+1-0)) & (c1_k1'=ceil(0)) & (k'= min(k+1,500))
1123 + 0.00621: (c2_k1'=ceil(c2_k1+1+0.3)) & (c1_k1'=ceil(0+0.3)) & (k'= min(k+1,500))
1124 + 0.060598: (c2_k1'=ceil(c2_k1+1+0.2)) & (c1_k1'=ceil(0+0.2)) & (k'= min(k+1,500))
1125 + 0.24173: (c2_k1'=ceil(c2_k1+1+0.1)) & (c1_k1'=ceil(0+0.1)) & (k'= min(k+1,500));
1126 // qb=5---qb=1 (brakes are disengaged)
1127 [] qb=5 & (c2_k1'>1) & (i>0) -> (qb'=1) & (i'=i-1);
1128 // qv=3-----qv=1 Brakes Disengaged
1129 [] qv=3 & disengage -> (qv'=1);
1130 endmodule
1131
1132 // Module for car 2
1133 module velocitysystem2=velocitysystem
1134 [qb=qb2,c1_k1=c1_k2,c2_k1=c2_k2]
1135 endmodule
1136 // Module for car 3
1137 module velocitysystem3=velocitysystem
1138 [qb=qb3,c1_k1=c1_k3,c2_k1=c2_k3]

```

```

1140 endmodule
1141 // Module for car 4
1142 module velocitysystem4=velocitysystem
1143 [qb=qb4,c1_k1=c1_k4,c2_k1=c2_k4]
1144 endmodule
1145 // Module for car 5
1146 module velocitysystem5=velocitysystem
1147 [qb=qb5,c1_k1=c1_k5,c2_k1=c2_k5]
1148 endmodule

```