Visual Verifier Suite

Guide to Examples

Cheng Pang

Valeriy Vyatkin

Contents

Introduction:

This document provides a guide to the NCES models in the standard library, including model description, sample testing scenario, reachability graph analysis, and verification of each model, typically using both safety and liveness properties. Moreover, some NCES features are also explained in the Miscellaneous sections.

Conventions and Syntaxes:

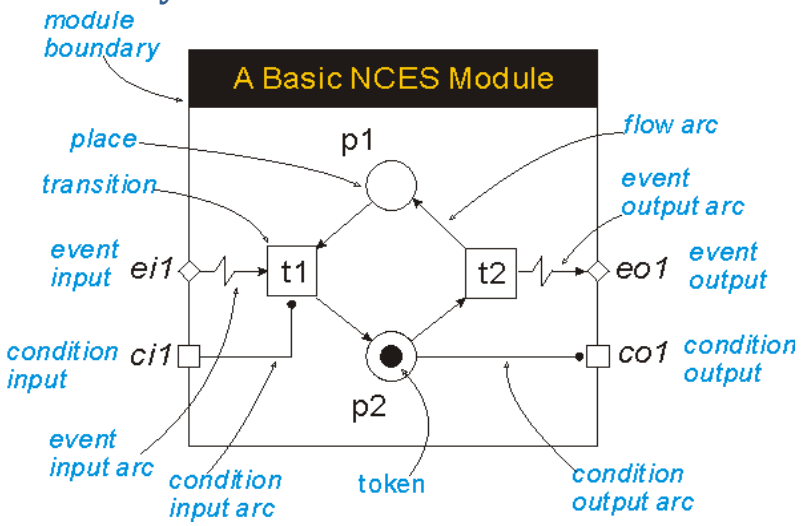

Figure 1 Graphical Notation of a Basic NCES Module

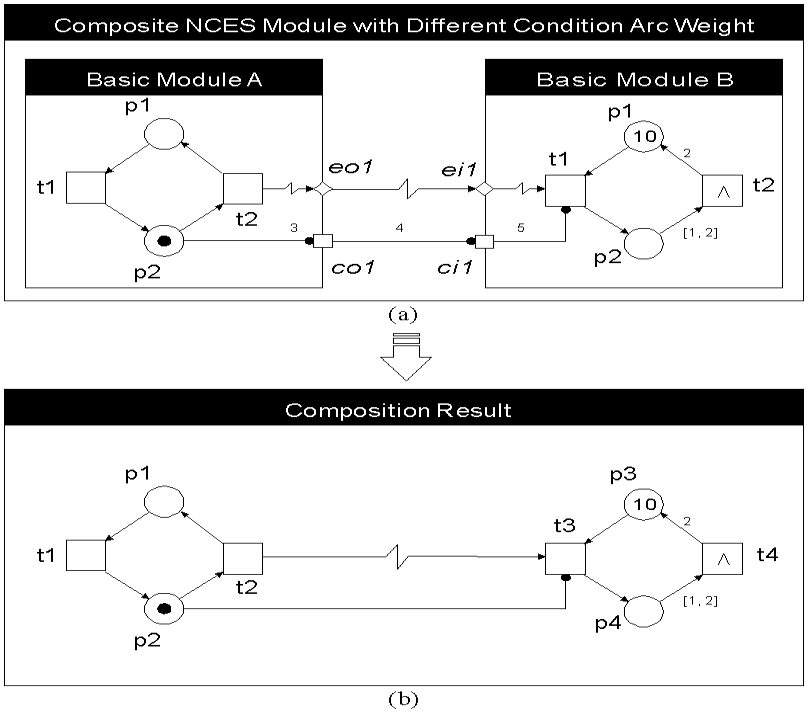

Figure 2 Composition of Composite NCES Module: (a) Original Module, and (b) Flatten Module

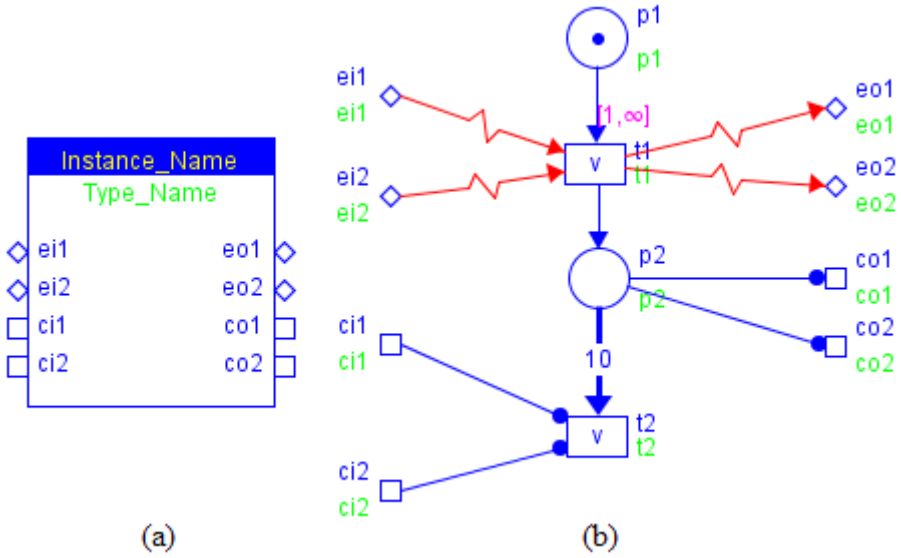

Figure 3 A Sample Basic NCES Module: (a) Module Interface, and (b) Module Content

- The module interface shows the port names and type information. The Instance Name is yellow and the Type Name is green. In basic NCES mode, type name and instance name are identical as shown in Figure 3 (a), whereas in composite NCES mode, you can define your own instance name. In this guide, normal text is used for Instance_Name and **Bolded** text is used for **Type_Name**.
- Event/condition input/output port (signal) name is in *Italic*, e.g. *ei1*. If two signals share the same name, append the instance name with a period (.) in front of the signal name to differentiate, e.g. Instance1.*eo1* and Instance2.*eo1*, which are called fully qualified name.
- The symbolic names of transition and place are blue whereas the ID's of transition and place are green if shown. You can only define your own symbolic name. Place and transition are directly referred to their names when referring to place/transition in flatten module or qualified with their instance name.
- Arcs are referred as *[Source, Destination]*, e.g. the event input arc from *ei1* to t1 is represented as *[ei1, t1]* or Instance1.*[ei1, t1]*.
- The number 10 beside flow arc [*p2*, *t2*] is the weight of the arc. Weights can be assigned to all kinds of arcs.
- The symbol [1; ∞] beside flow arc *[p1, t1]* indicates the time interval of this arc.
- Specifying token location and flow path:

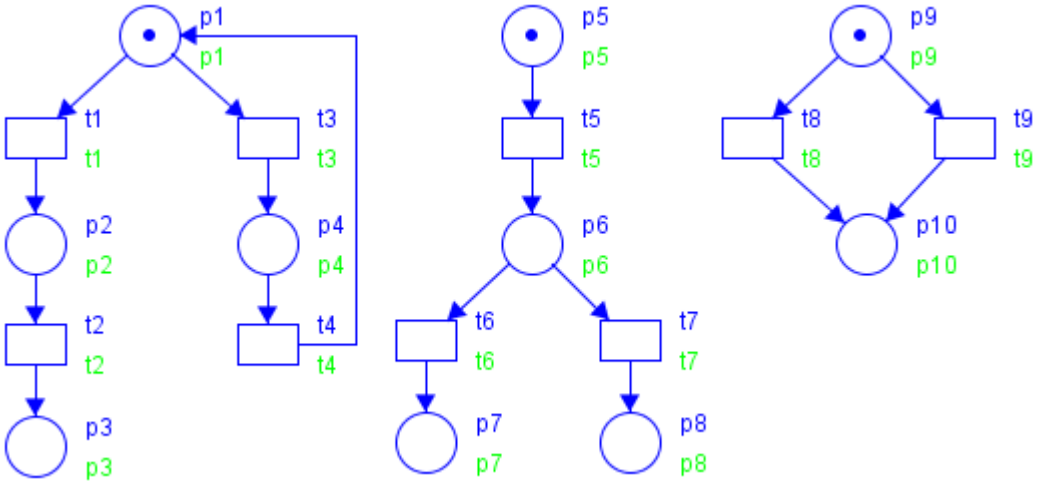

Figure 4 Token Location and Flow Path

Usually basic NCES module contains multiple tokens, but token doesn't possess any ID. Therefore we concatenate prefix TK and the ID (in green colour) of the place holding the token to specify the token, e.g. TKp1 stands for the token inside place p1 in the above diagram. The token ID can also be qualified.

Token flow path can be specified by using the following syntax:

$$\left\{ \mathrm{pi} \rightarrow \underbrace{(\,\mathrm{pb1}|\mathrm{tb1} \rightarrow \mathrm{pb2}|\mathrm{tb2} \rightarrow \cdots \rightarrow \mathrm{pbn}|\mathrm{tbn})}_{\mathrm{n}} \rightarrow \mathrm{pl} \middle| \mathrm{tl} \right\} (\mathrm{L}) \qquad (1.1)$$

where:

pi – the initial place holding the token
pb1 – the first non-initial place leading a branch, e.g. p6 in above diagram
pbn – the n^{th} non-initial place leading a branch
tbn – the n^{th} branching transition, e.g. t8, t9
pl – the last place in the flow path, e.g. p7
tl – the last transition in the flow path, e.g. t4
L – indicates the specified flow path is a loop
() – means the content inside it is optional
| - means 'OR'

For example:

$\{\mathrm{p1} \rightarrow \mathrm{p3}\}$ specifies the flow path $\mathrm{p1} \rightarrow \mathrm{t1} \rightarrow \mathrm{p2} \rightarrow \mathrm{t2} \rightarrow \mathrm{p3}$

$\{\mathrm{p1} \rightarrow \mathrm{t4}\}\mathrm{L}$ specifies the flow path $\mathrm{p1} \rightarrow \mathrm{t3} \rightarrow \mathrm{p4} \rightarrow \mathrm{t4} \rightarrow \mathrm{p1}$ which is a loop

$\{\mathrm{p5} \rightarrow \mathrm{p6} \rightarrow \mathrm{p8}\}$ specifies the flow path $\mathrm{p5} \rightarrow \mathrm{t5} \rightarrow \mathrm{p6} \rightarrow \mathrm{t7} \rightarrow \mathrm{p8}$

- Input Sequence: the sequence of inputs, including both event and condition input signals, is specified in the following syntax:

$$\left\{ \underbrace{\text{Input signals for S1; } \cdots \text{; Input signals for Sn}}_{\mathrm{n}} \right\} (\mathrm{L}) \qquad (1.2)$$

where:

Input signals – any event/condition signals. N.B. a condition signal is included only if it is true in the state.

Sn – the n^{th} state

L – indicates whether this input sequence is cyclical

For example, $\{ei1, ei2, ci1; ci3; ei1|ei2, ci2\}L$ defines the following cyclical input sequence:

S1: ei1, ei2, ci1

S2: ci3

S3: ei1 or ei2, ci2

S1: ei1, ei2, ci1

Standard Library Models

Model Name: E_AND

Model Descriptions:

Perform AND operation on the two event input signals. Event output signal *eo1* will be issued upon the occurrence of both event signal *ei1* and *ei2*.

Model Details:

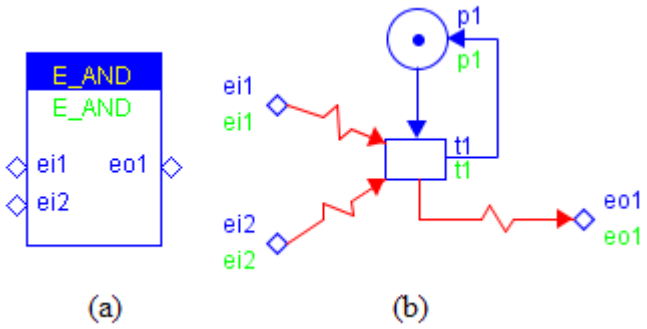

Figure 5 E_AND.xml

Simple Testing Scenario:

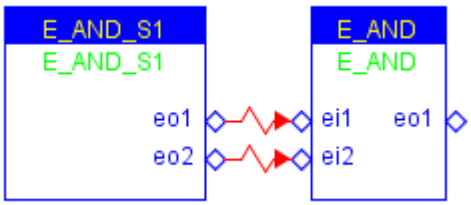

Figure 6 TEST_E_AND_S1.xml

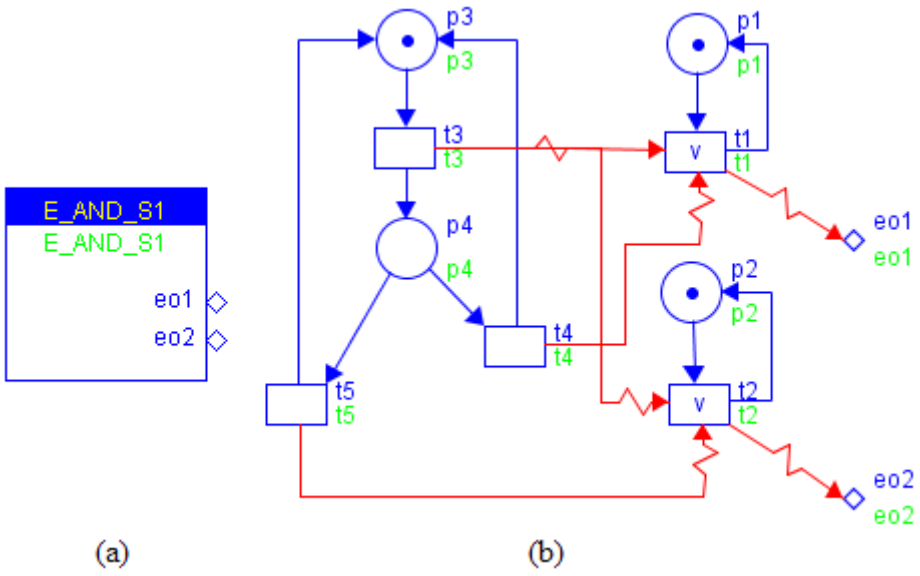

Figure 7 E_AND_S1.xml

The **E_AND_S1** model generates the following input sequence:

$$\{E\_AND\_S1.eo1, E\_AND\_S1.eo2; E\_AND\_S1.eo1 | E\_AND\_S1.eo2\}\mathrm{L}$$

ViVe Reachability Graph:

[All testing scenarios use Maximal Firing rule and other settings remain default]

Note: *ViVe assembles the individual model instances into a single NCES module, which is called flattening, and reorders and re-labels the places and transitions. Therefore, the reachability graph is generated based on the flattened module. Please refer to the assembled module when reading the* **ViVe Reachability Graph** *and* **Model Verification** *sections, unless the names are fully qualified.*

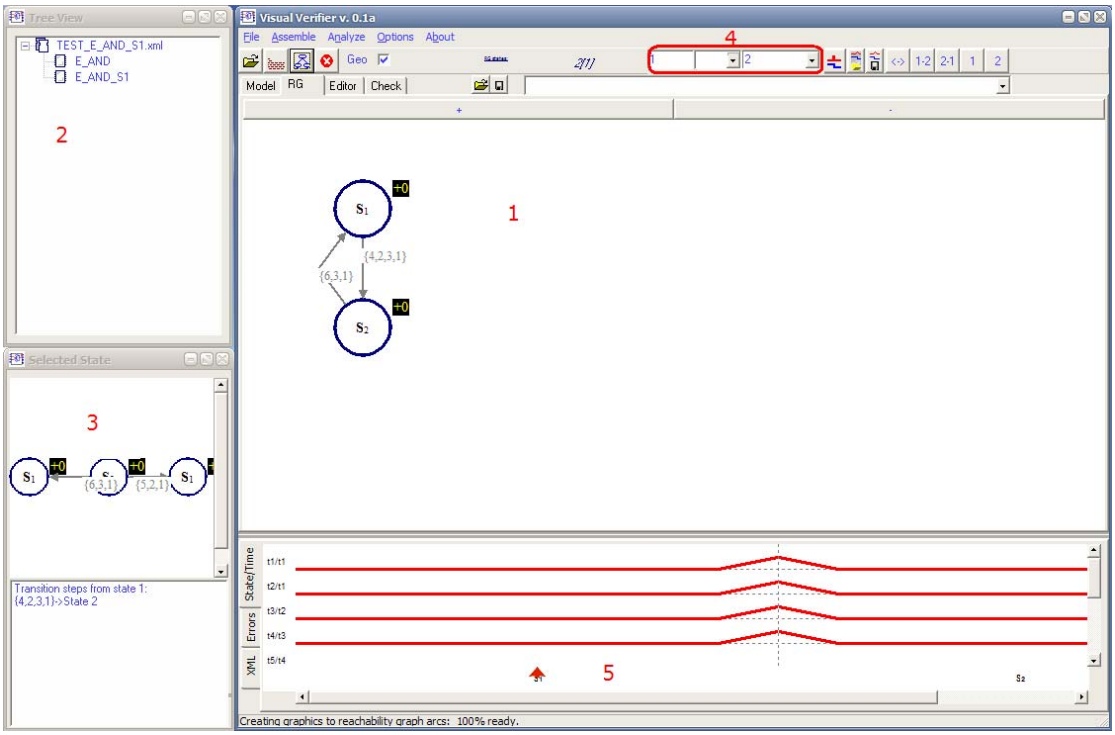

Figure 8 Screenshot of ViVe where 1 RG tab displaying the reachability graph; 2 Tree View listing all models used inside current module; 3 Selected State window showing the selected state in the reachability graph and its succeeding states; 4 Trace Toolbar where value in the text box indicates the first state number of the trace, value in the middle combo box indicates the intermediate state numbers, and value in the last combo box indicates the last state number; 5 State-Time Diagram visualises the state transactions in the selected path

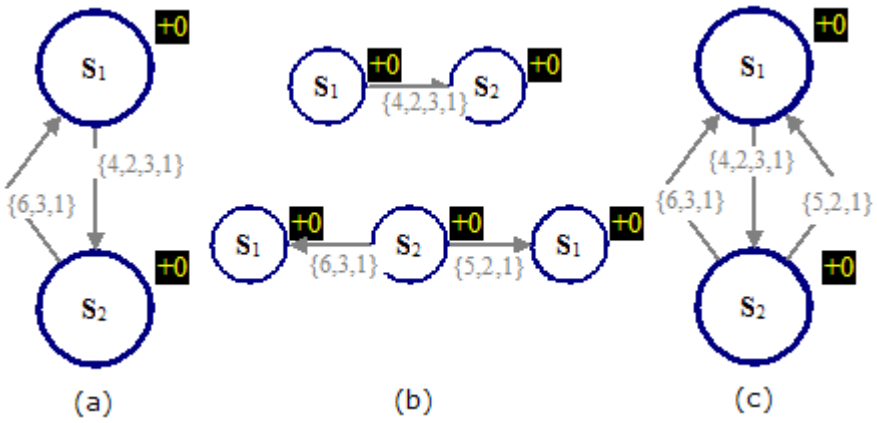

Figure 9 (a) Snapshot of ViVe RG Tab (b) Snapshot of ViVe Select State Window (c) Combined Reachability Graph

Note: *The current version of ViVe has some limitations in displaying the reachability graph in the RG tab and the enabled transitions in the Model tab. For the RG tab, when the destination states of multiple transition steps are identical, the RG tab will only display one transition step as shown in* Figure 9 *(a).* Figure 9 *(b) shows that there are two transition steps from S2 to S1: Transition Step {6, 3, 1} and {5, 2, 1}. However, the RG tab only displays one transition step. Moreover, since there is no intermediate state it is impossible to select the other transition step by using the Trace Toolbar. The workaround for this is to select the source state, e.g. S2, and examine all possible transition steps in the Selected State window. In this guide the ViVe reachability graphs are manually processed to include all transition steps as illustrated in* Figure 9 *(c) for your convenience.*

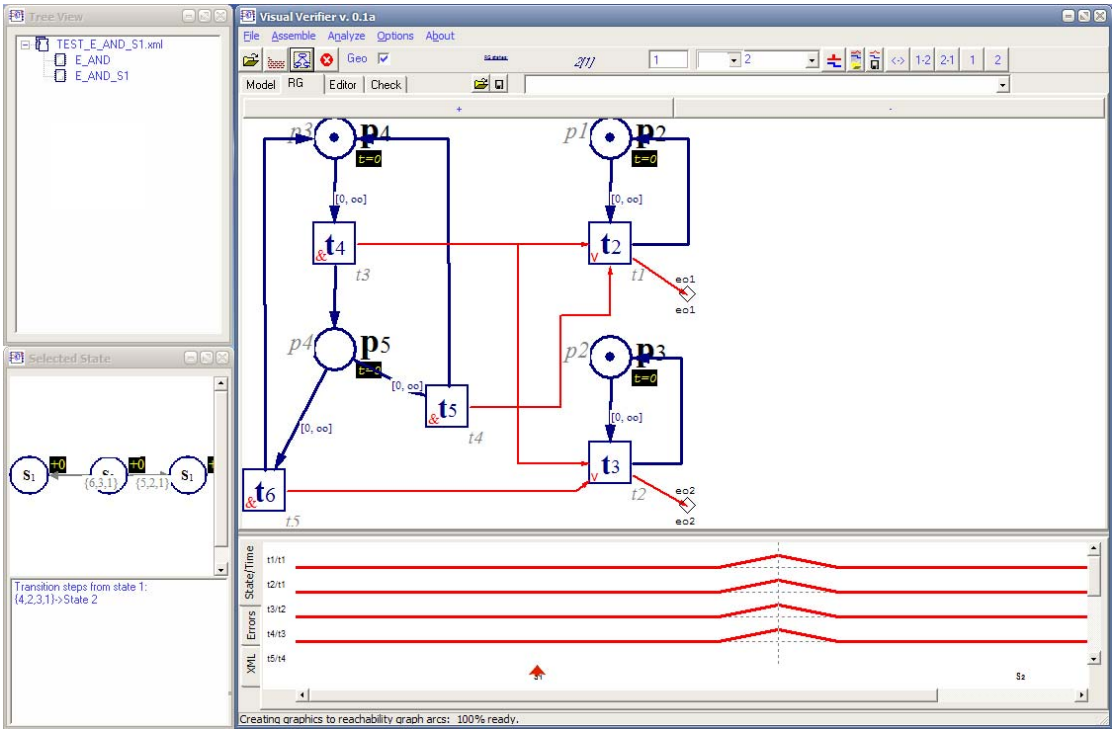

Figure 10 Limitation of the ViVe Model Tab

Normally the Model tab of ViVe highlights the enabled transitions in current state. However, if the states in the reachability graph form a loop and the last state is selected, the Model tab will not highlight the enabled transitions. For example, Figure 10 *shows the Model tab displaying the content of model* **E_AND_S1** *in State S2. In according to the* Figure 9 *(a), t6 is enabled and must be highlight in the Model tab. The workaround for this limitation is also to refer to the Selected State window and set the steps in the Trace Toolbar.*

Model **E_AND** is designed to model Boolean AND operation, therefore t1 will only be enabled when both E_AND_S1.*eo1* and E_AND_S1.*eo2* present simultaneously. This behaviour is verified in according to the reachability graph shown in Figure 9 (c).

Model Verification:

Note: *Sometimes it is impossible to manually examine the entire reachability graph; therefore it is easier and better to use model checking facility come with ViVe to verify the designed system's specifications.*

Properties to be checked:

1. Transition E_AND_S1.*t5* is only enabled when event signal E_AND_S1.*eo1* and E_AND_S1.*eo2* present simultaneously.

CTL formulae: *[All CTL formulae are represented in the SESA syntax]*

$$\mathrm{EX\ E}\left(\left(\left(\mathrm{t3\ AND\ NOT\ t2}\right)\ \mathrm{OR}\ \left(\mathrm{t2\ AND\ NOT\ t3}\right)\right)\mathrm{AND\ t1}\right)\mathrm{X\ m(p1)} = 1$$

Note: *This model is trivial and can be directly verified by physically examining the reachability graph. Moreover, due to the special case of p1, which always holds a token, the eCTL formula is more emphasis on the transition steps. Therefore, the sample eCTL formula is demonstrative and not generic.*

Miscellaneous:

Conflicts and non-determinism in NCES: as shown in Figure 7, when TKp3 flows to *p4* both *t4* and *t5* will be enabled simultaneously, as a result TKp4 can flow back to *p3* either via *t4* or *t5*. This ambiguous or non-deterministic token flow situation is called *conflict* in NCES. ViVe handles conflicts by including combinations of all possible flow paths in the reachability graph. This feature facilitates the design of testing scenarios by reducing the number of place and token required to generate all possible combinations of event/condition signals.

Model Name: C_E_Convertor

Model Description:

The **C_E_Convertor** model converts condition signal to event signal. Upon the occurrence of event input signal *ei1*, if the condition input signal *ci1* is enabled, the event output signal *eo1* will be issued; otherwise no event output signal will be generated.

Note: E_Permit_ALG, E_Trigger are functionally identical to **C_E_Convertor** .

Model Details:

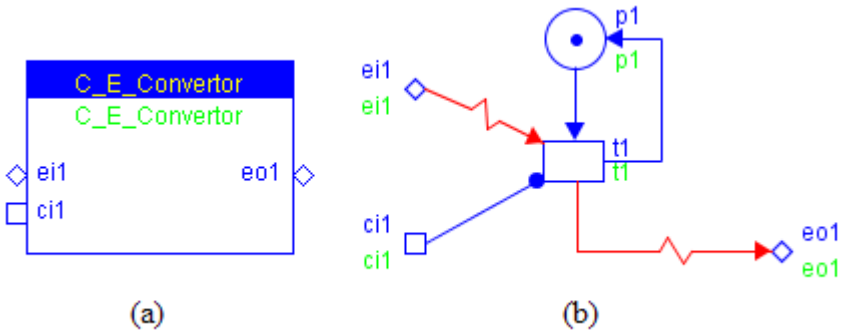

Figure 11 C_E_Convertor.xml

Simple Testing Scenario:

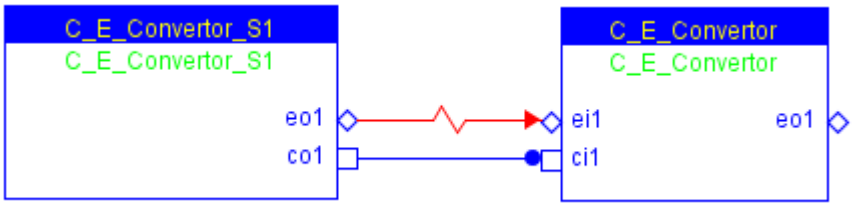

Figure 12 TEST_C_E_Convertor_S1.xml

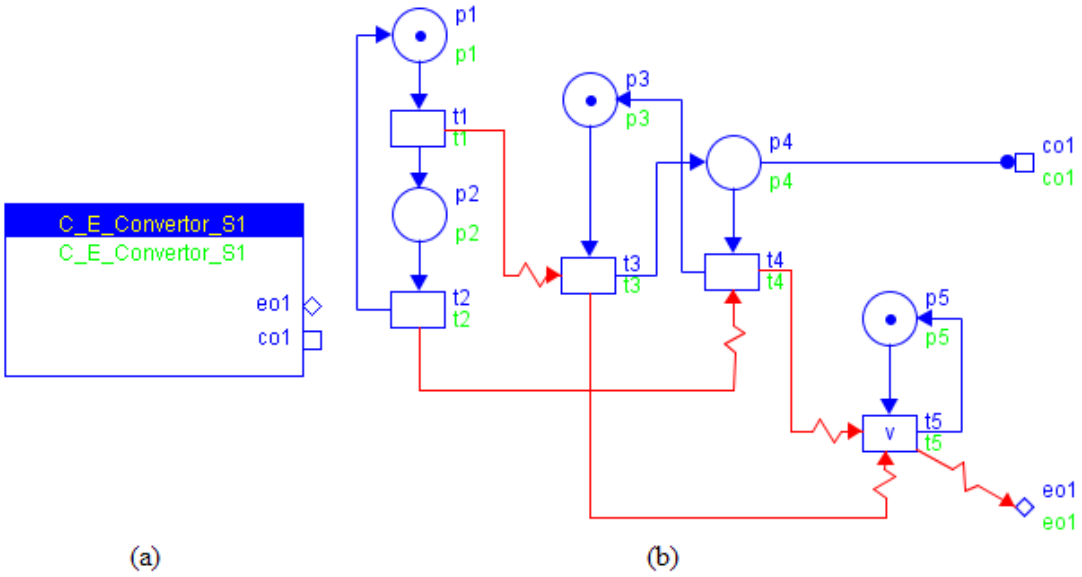

Figure 13 C_E_Convertor_S1.xml

The **C_E_Convertor_S1** model generates the following input sequence:

$$\{\text{E\_Convertor\_S1.eo1; E\_Convertor\_S1.eo1, E\_Convertor\_S1.co1}\}\text{L}$$

ViVe Reachability Graph:

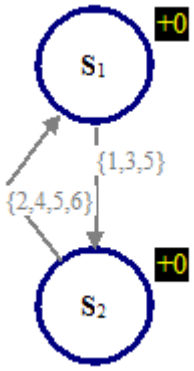

Figure 14 RG of TEST_C_E_Convertor_S1

Model Verification:

Properties to be checked:

1. Transition C_E_Convertor.*t1* is only enabled when C_E_Convertor.*ci1* is enabled and C_E_Convertor.*ei1* presents.

CTL formulae:

1. $\text{EX E (t6)X m(p3)} = 1$
   This formula specifies in S2 C_E_Convertor.*ci1* is enabled and C_E_Convertor.*t6* will fire in the following transition step.

Miscellaneous:

Model Name: E_DELAY

Model Descriptions

Models a delay of 100 time units. When event input signal *START* arrives after 100 time unit delay, the event output signal *eo1* will be issued. Moreover, when *STOP* arrives the timing will be interrupted and the model will be reset.

Model Details:

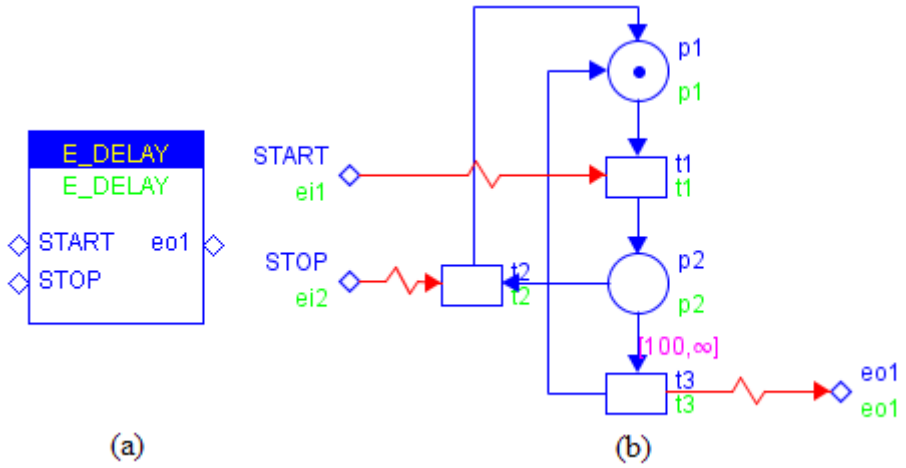

Figure 15 E_DELAY.xml

Simple Testing Scenario:

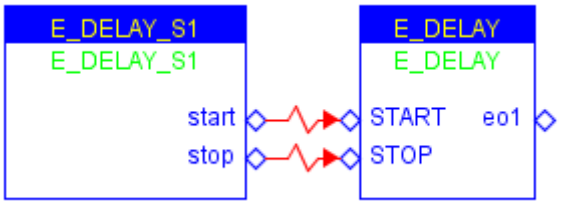

Figure 16 TEST_E_DELAY_S1.xml

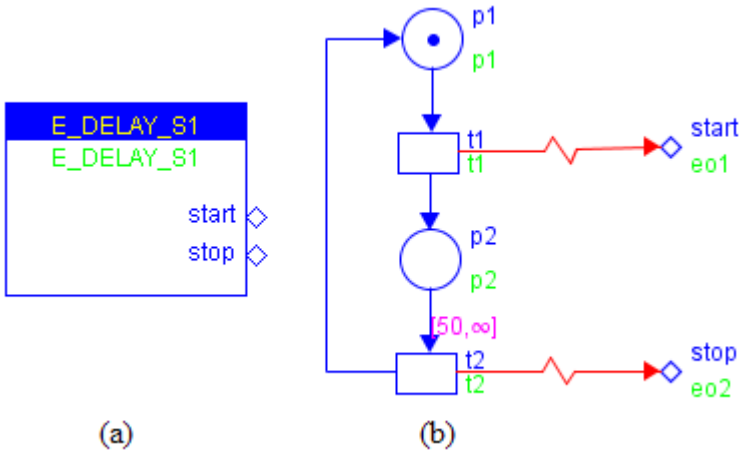

Figure 17 E_DELAY_S1.xml

ViVe Reachability Graph:

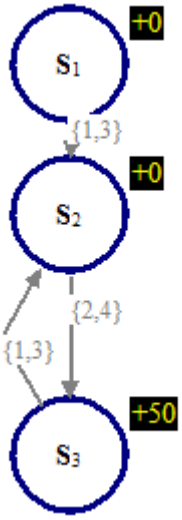

Figure 18 RG of TEST_E_DELAY_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

When use this model attention should be paid to the timed flow arc *[p2, t3]* and other time flow arc in the entire module to avoid dead lock.

Model Name: E_MERGE

Model Descriptions:

Merging two event input signals. Whenever *ei1* or *ei2* occurs, **E_MERGE** will issue event output signal *eo1*.

Note: E_OR2, **E_OR3**, **E_OR4**, **E_OR7**, **E_DEMUX_OR**, etc have the same structure as **E_MERGE** but with different number of event inputs.

Model Details:

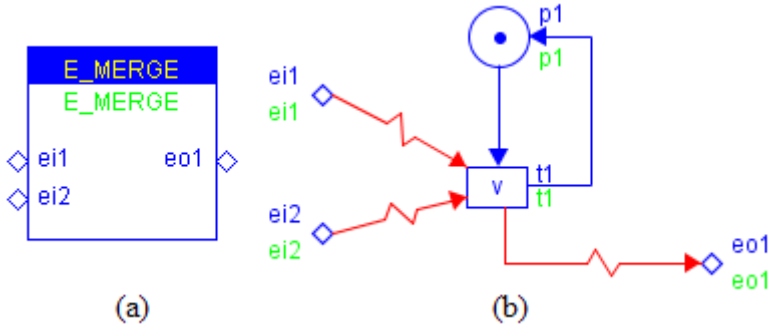

Figure 19 E_MERGE.xml

Simple Testing Scenario:

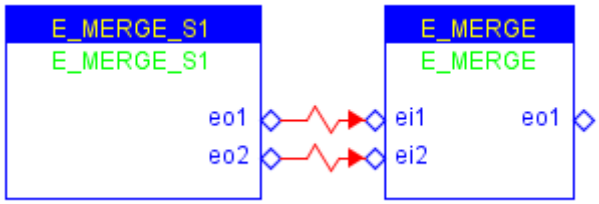

Figure 20 TEST_E_MERGE_S1.xml

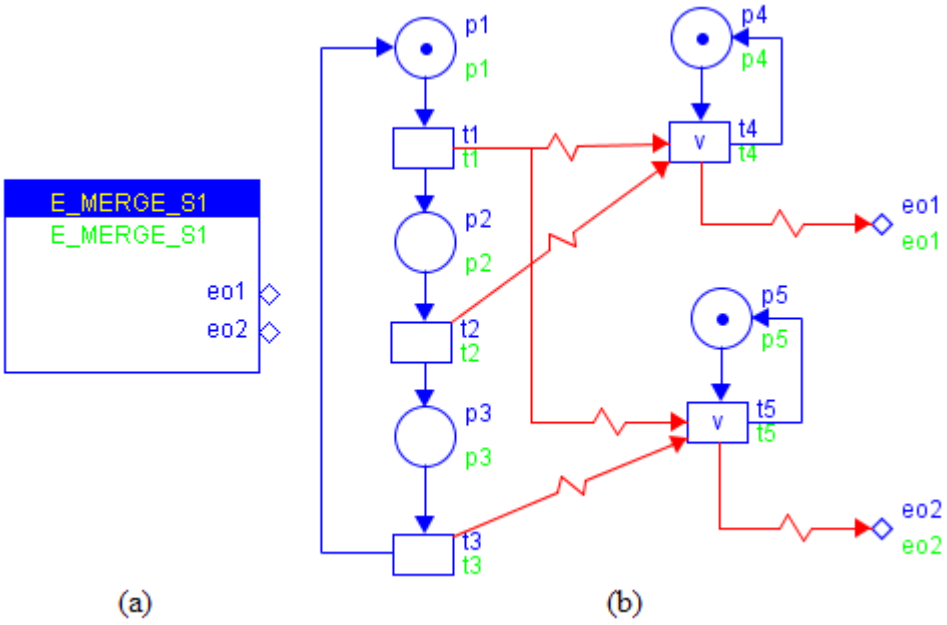

Figure 21 E_MERGE_S1.xml

The **E_MERGE_S1** model generates the following input sequence:

$$\{E\_Convertor\_S1.\,eo1;\ E\_Convertor\_S1.\,eo1,\ E\_Convertor\_S1.\,co1\}L$$

ViVe Reachability Graph:

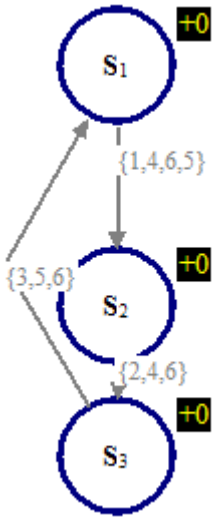

Figure 22 RG of TEST_E_MERGE_S1

Model Verification:

Properties to be checked:

1. E_MERGE.*t1*

CTL formulae:

Miscellaneous:

Model Name: E_SPLIT

Model Descriptions:

Splitting one event input signal to two sequential event output signals.

Model Details:

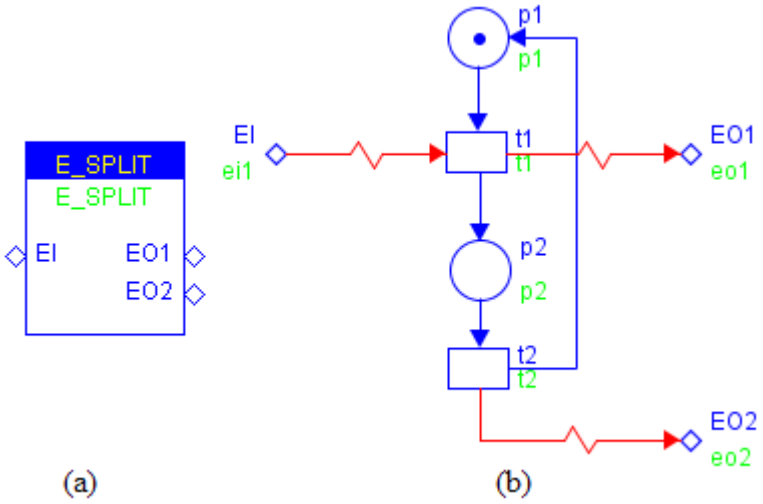

Figure 23 E_SPLIT.xml

Simple Testing Scenario:

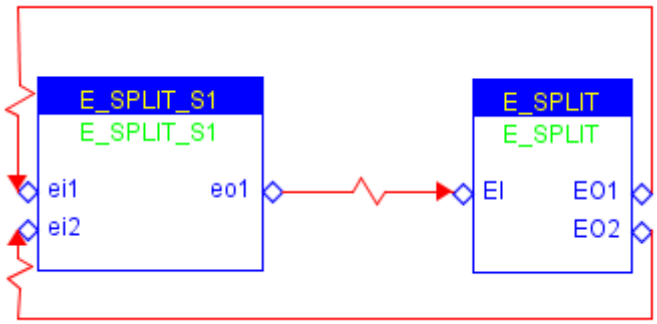

Figure 24 TEST_E_SPLIT_S1.xml

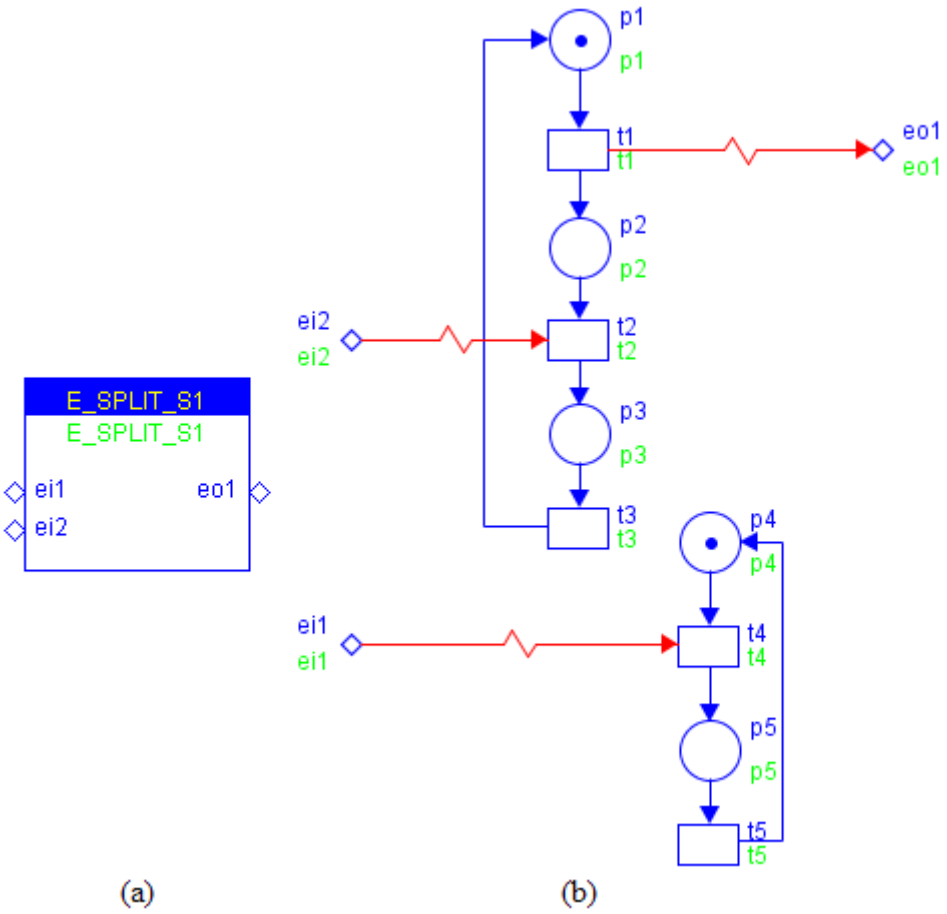

Figure 25 E_SPLIT_S1.xml

ViVe Reachability Graph:

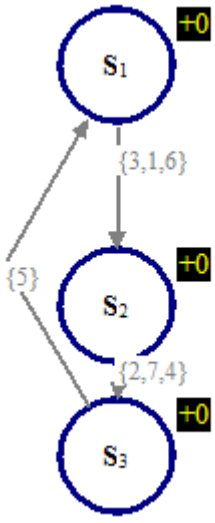

Figure 26 RG of TEST_E_SPLIT_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_Bistable

Model Descriptions:

A bistable used to represent one of two possible states. E_Bistable can be set or reset according to the input signal and issues an event output signal *Update* to inform other connected modules.

Model Details:

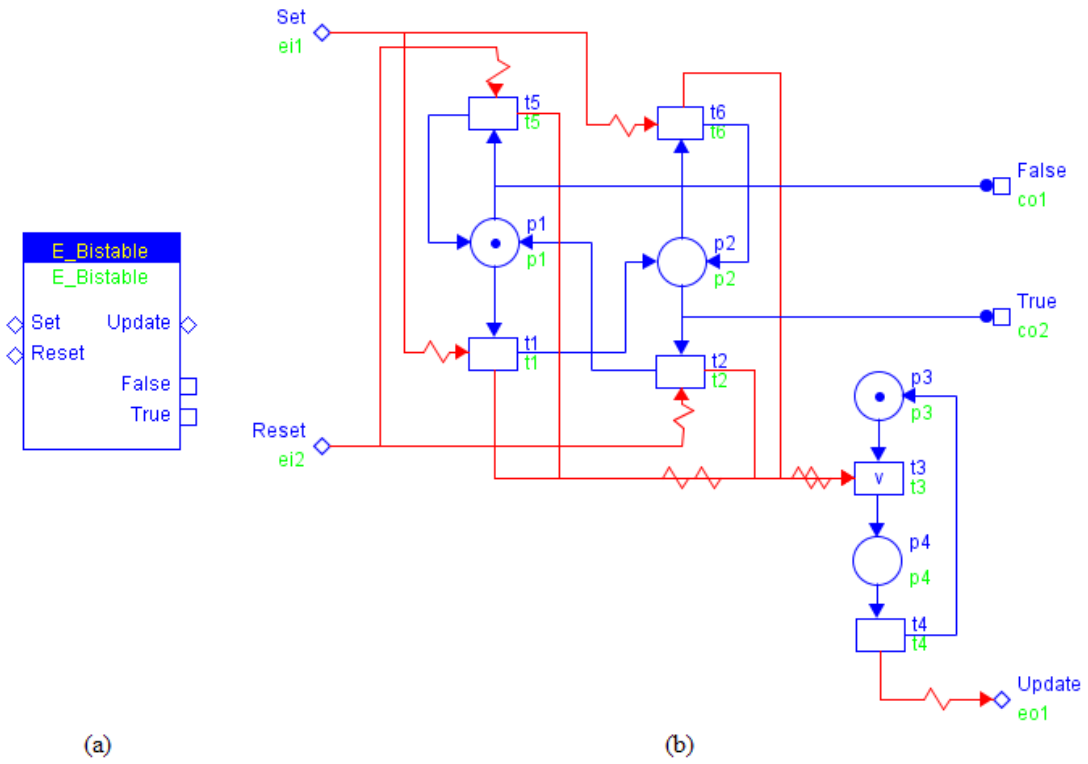

Figure 27 E_Bistable.xml

Simple Testing Scenario:

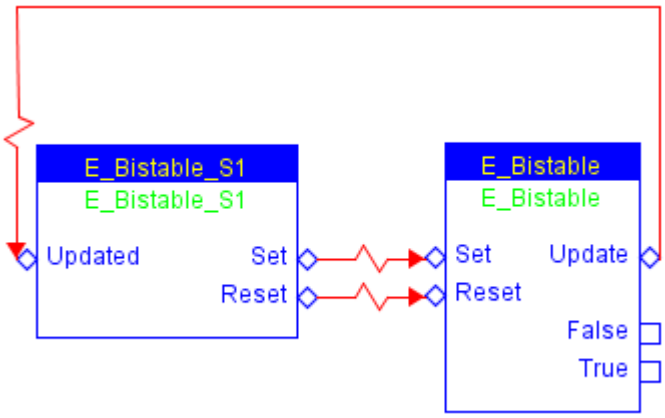

Figure 28 TEST_E_Bistable_S1.xml

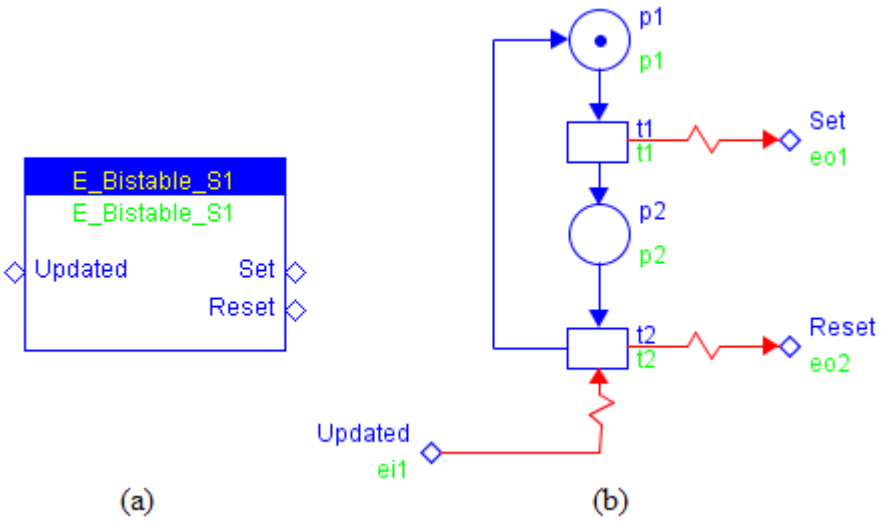

Figure 29 E_Bistable_S1.xml

ViVe Reachability Graph:

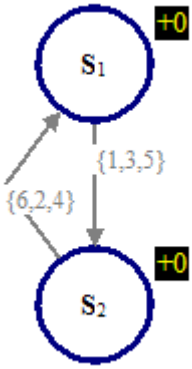

Figure 30 RG of TEST_E_Bistable_S1.xml

Model Verification:

Properties to be checked:

CTL formulae:

Initially the bistable is in the False state, i.e. p1 holds a token, then the bistable is set to the True state. When the event input signal *Reset* arrives, the bistable is reset to the default state again.

Miscellaneous:

Model Name: Domin_EI1_ALG

Model Descriptions:

The Domin_EI1_ALG introduces a priority mechanism to NCES. Event output signal *EI1_Out* is responsible to event input signal *EI1*, whereas *EI1_Out* is responsible to *EI2*. When both *EI1* and *EI2* present simultaneously, only *EI1_Out* is issued.

Model Details:

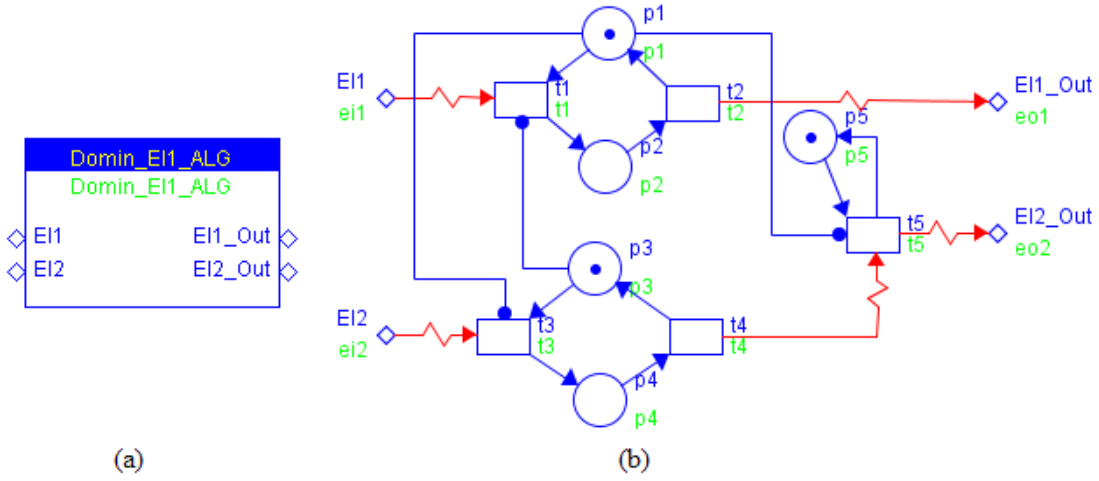

Figure 31 Domin_EI1_ALG.xml

Simple Testing Scenario:

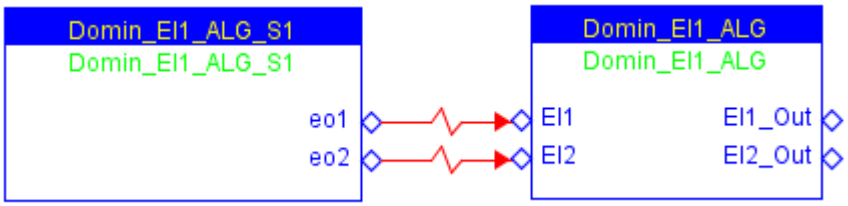

Figure 32 TEST_Domin_EI1_ALG_S1.xml

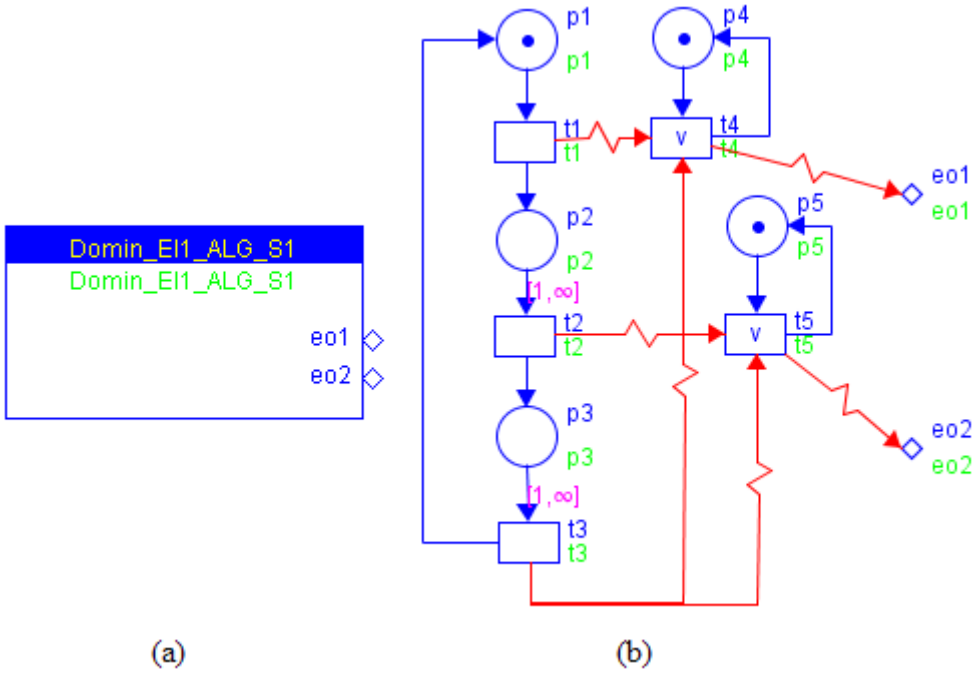

Figure 33 Domin_EI1_ALG_S1.xml

ViVe Reachability Graph:

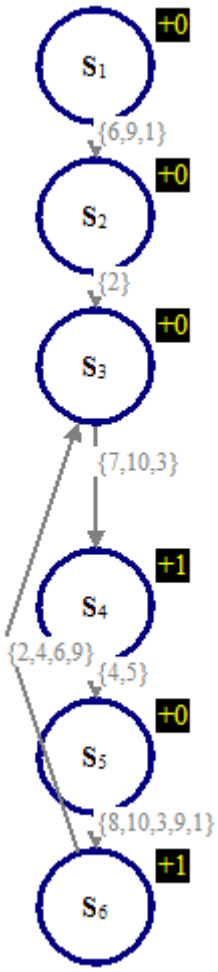

Figure 34 RG of TEST_Domin_EI1_ALG_S1

The testing scenario module Domin_EI1_ALG_S1 cyclically generates the event sequence: eo1->eo2->[eo1 and eo2]. When both event signal *eo1* and *eo2* are sent to Domin_EI1_ALG, only *EI1_Out* is issued (indicated in S6). The reachability graph also shows a state loop, S3->S4->S5->S6->S3.

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

The flow arc [p2, t2] and [p3, t3] are timed, which fire only when no more transition is enabled in the entire module (in this case the module containing Domin_EI1_ALG_S1 and Domin_EI1_ALG). **Timed flow arcs are extremely useful to detect the completeness of certain operations.**

Model Name: E_D_FF_ALG

Model Descriptions:

This model models the D Flip Flop. Based on the condition input signals, E_D_FF_ALG adjusts its internal bistable and issues the corresponding event output signal *EO* and condition output signals.

Model Details:

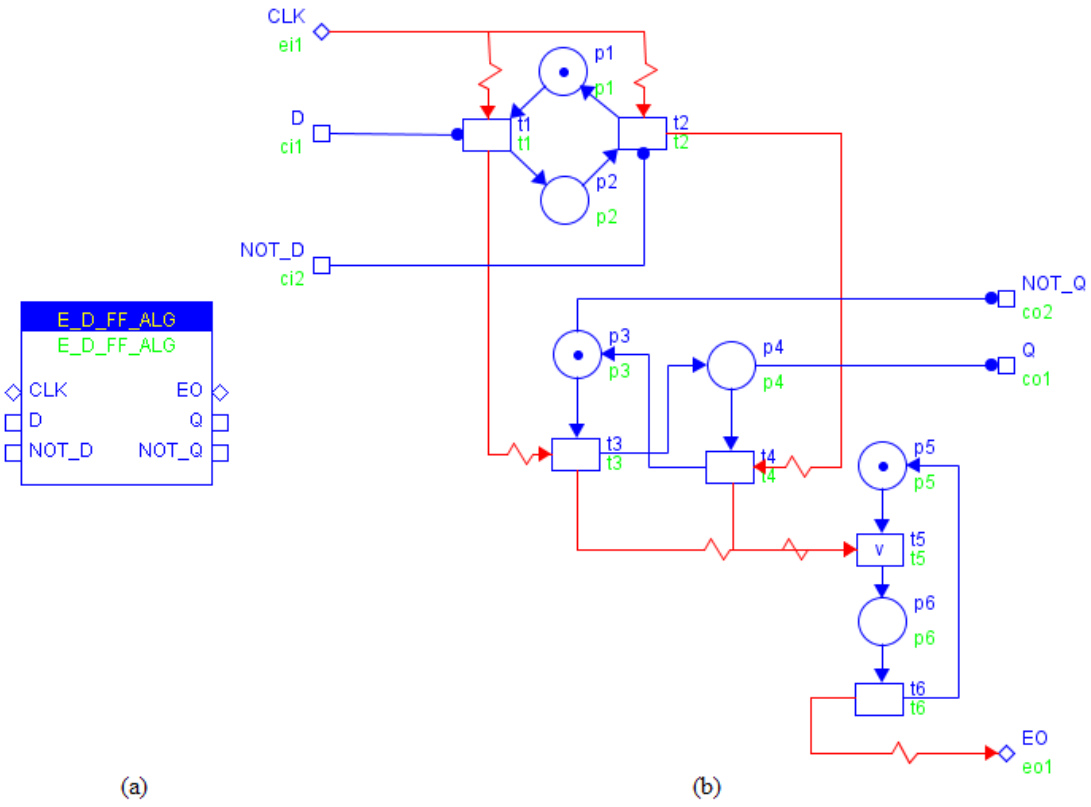

Figure 35 E_D_FF_ALG.xml

Simple Testing Scenario:

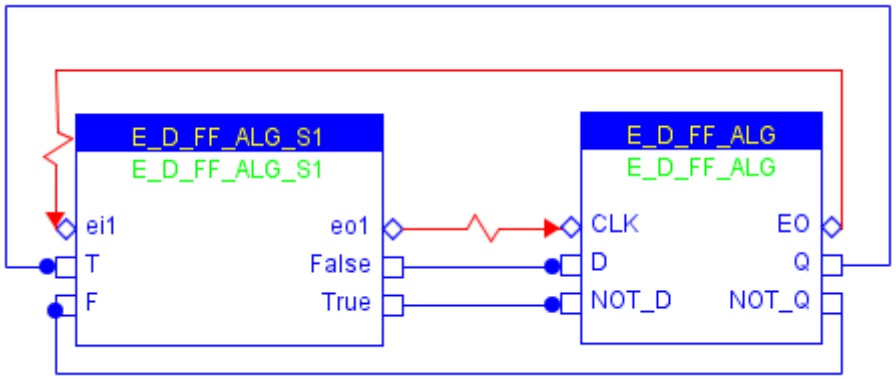

Figure 36 TEST_E_D_FF_ALG_S1.xml

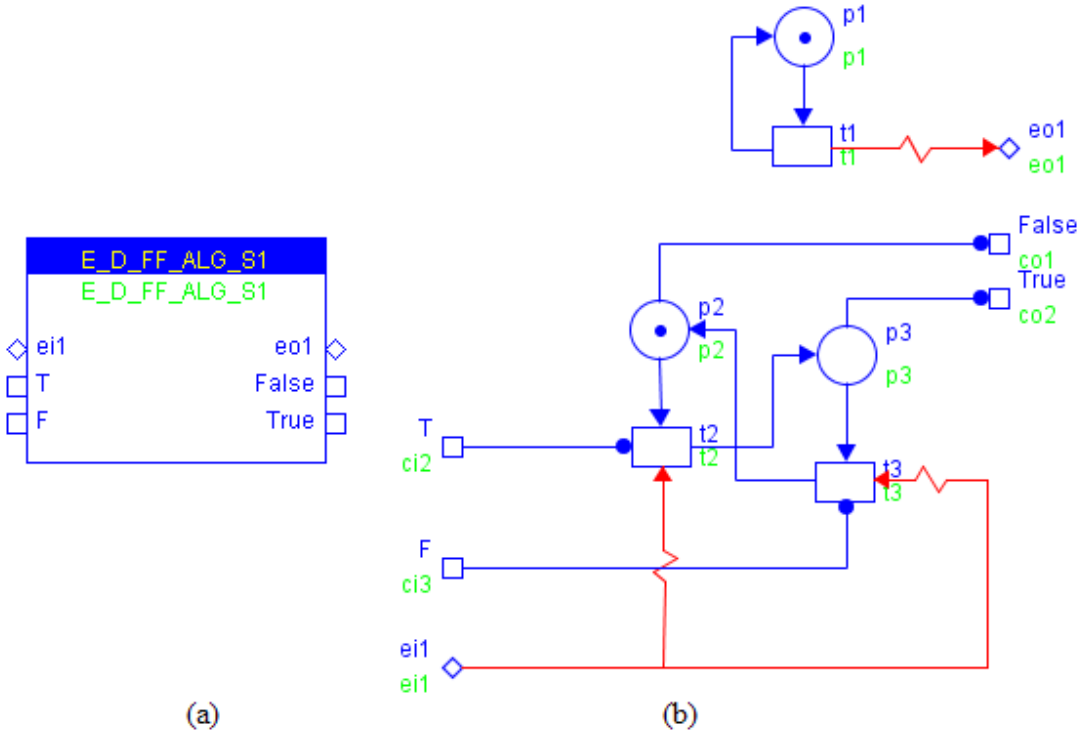

Figure 37 E_D_FF_ALG_S1.xml

ViVe Reachability Graph:

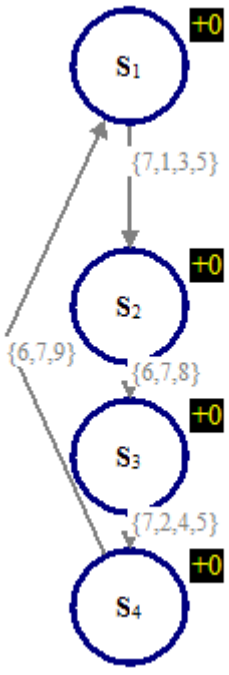

Figure 38 RG of TEST_E_D_FF_ALG_S1

p1 and t1 form a timer issuing event output signal *eo1*. The two bistables inside E_DFF_ALG_S1 and E_D_FF_ALG are interconnected and update each other.

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_Compare

Model Descriptions:

Based on the event input signal, **E_Compare** issues the corresponding conditon output signal and the *Sampled* event output signal. The following state diagram illustrates the state transition relationship.

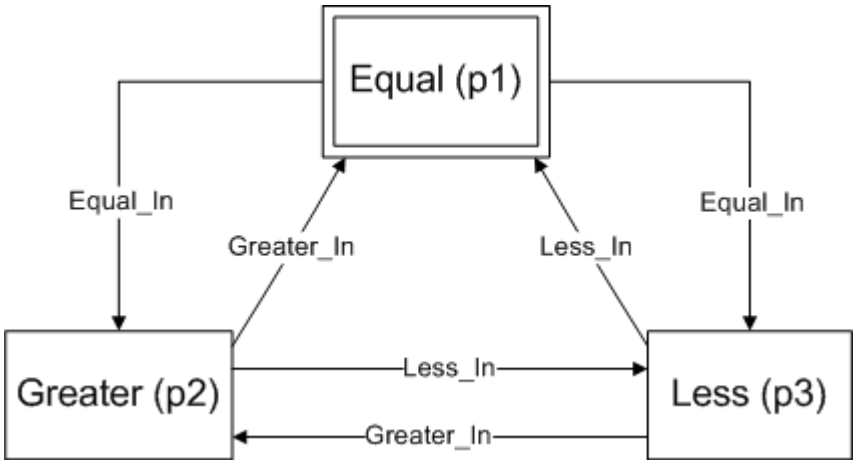

Figure 39 State Diagram of E_Compare

Model Details:

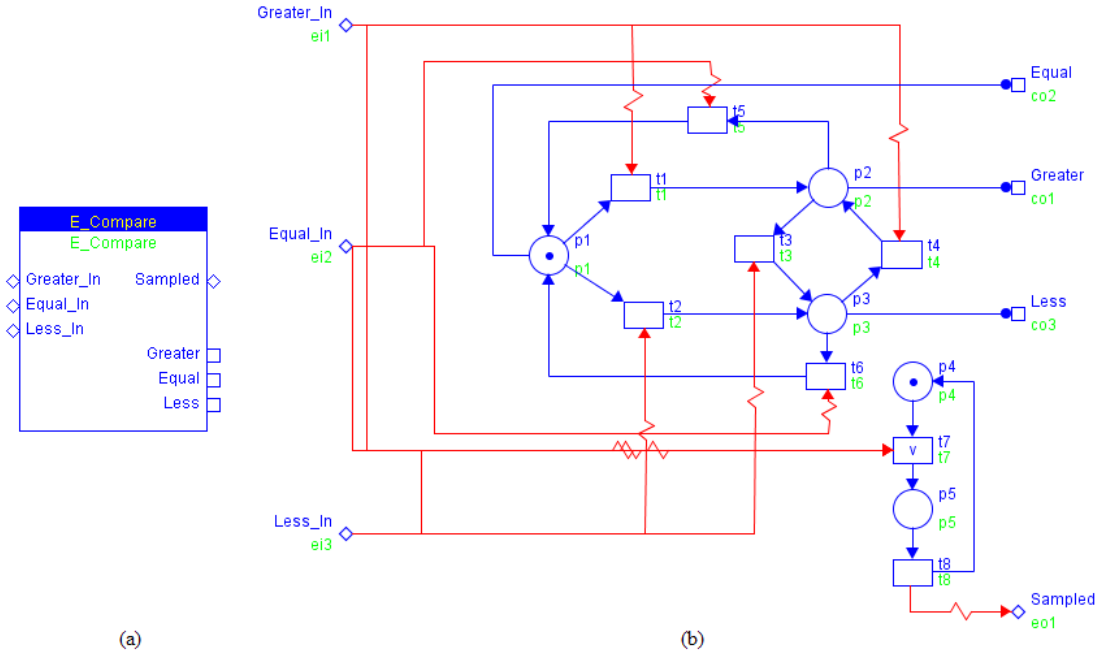

Figure 40 E_Compare.xml

Simple Testing Scenario:

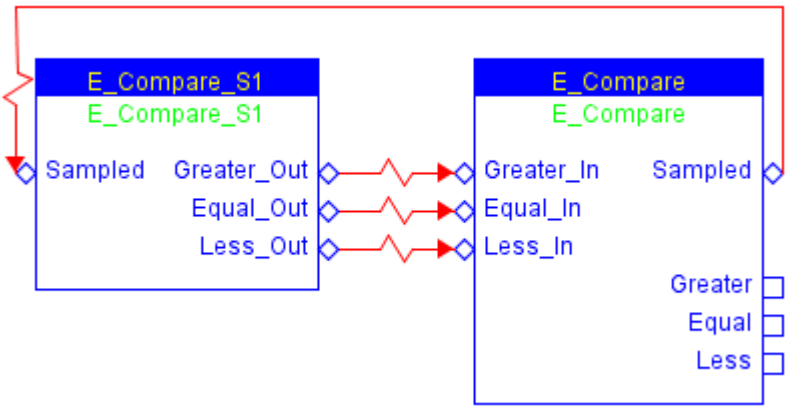

Figure 41 TEST_E_Compare_S1.xml

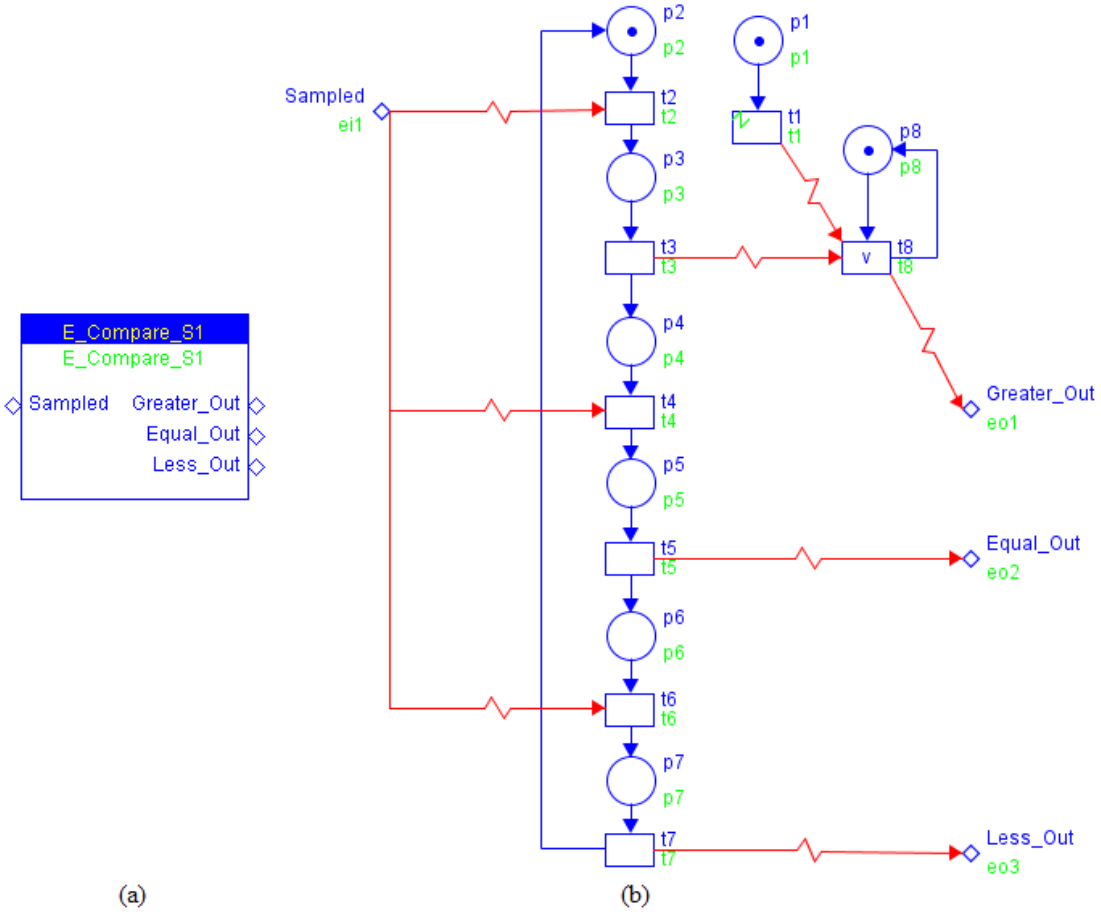

Figure 42 E_Compare_S1.xml

ViVe Reachability Graph:

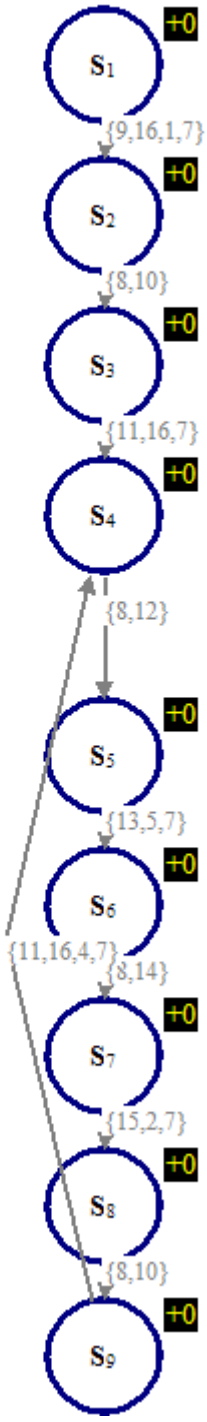

Figure 43 RG of TEST_E_Compare_S1.xml

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_Compare_I

Model Descriptions:

An enhanced version of **E_Compare** which introduces an extra state *Initial* and an event input signal *Reset* to reset the model back to its initial state as shown below:

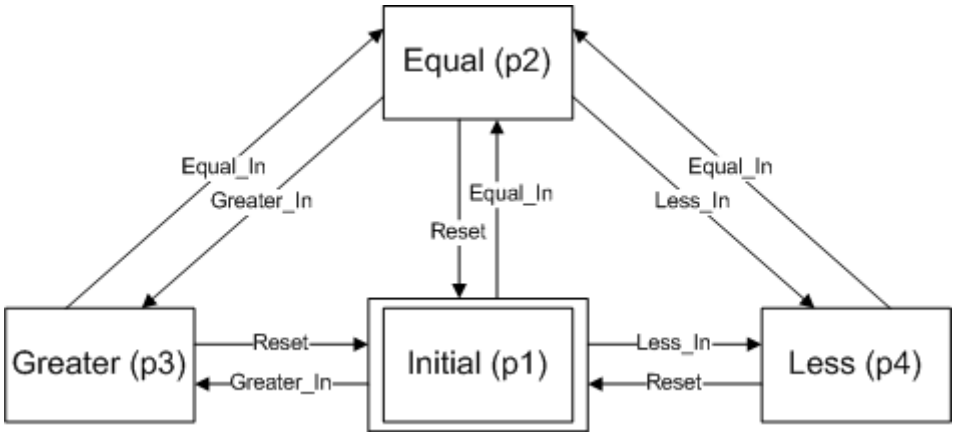

Figure 44 State Diagram of E_Compare_I

Model Details:

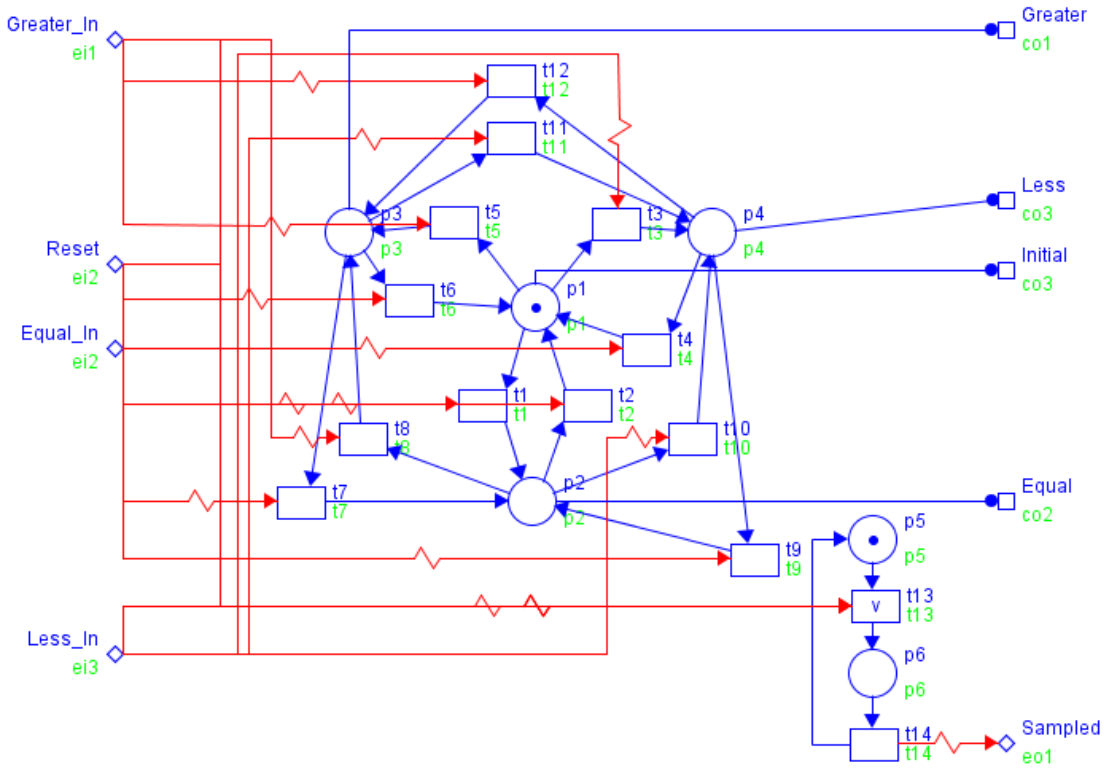

Figure 45 E_Compare_I.xml

Simple Testing Scenario:

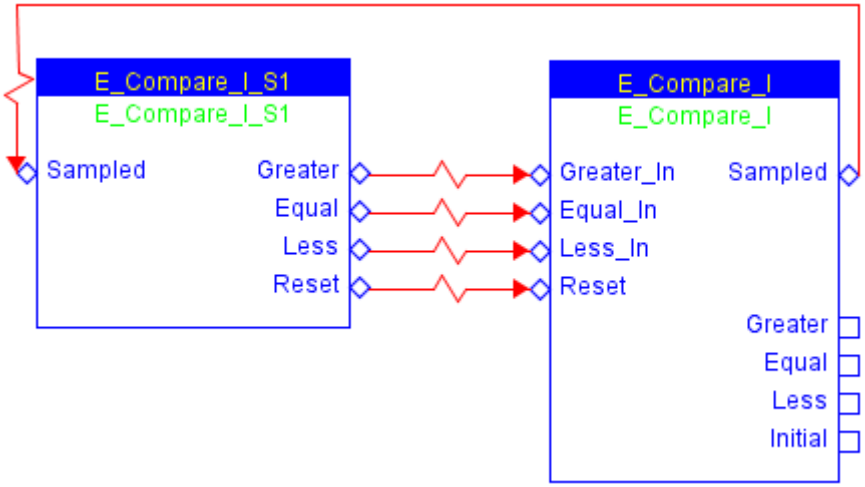

Figure 46 TEST_E_Compare_I_S1.xml

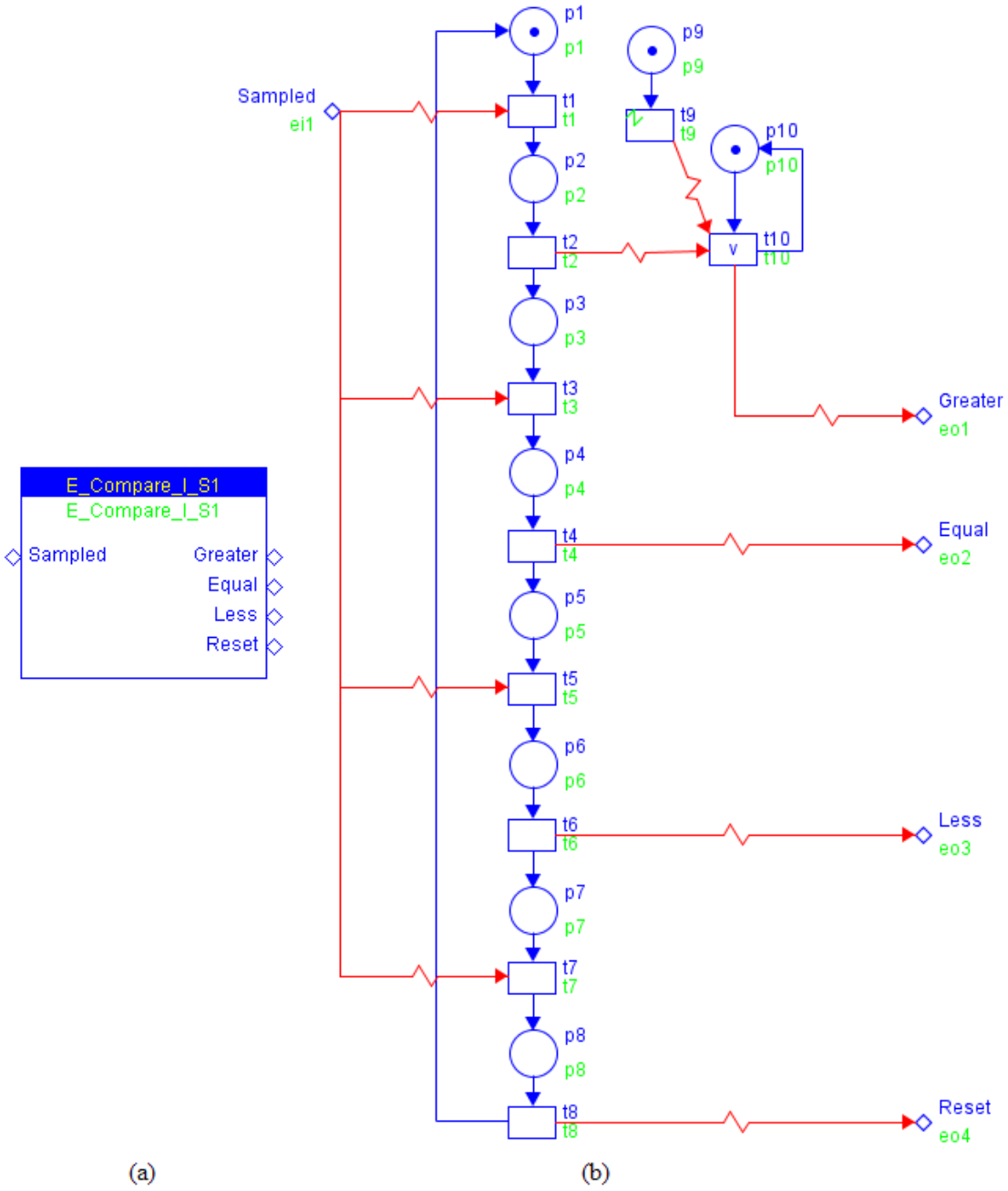

Figure 47 E_Compare_I_S1.xml

ViVe Reachability Graph:

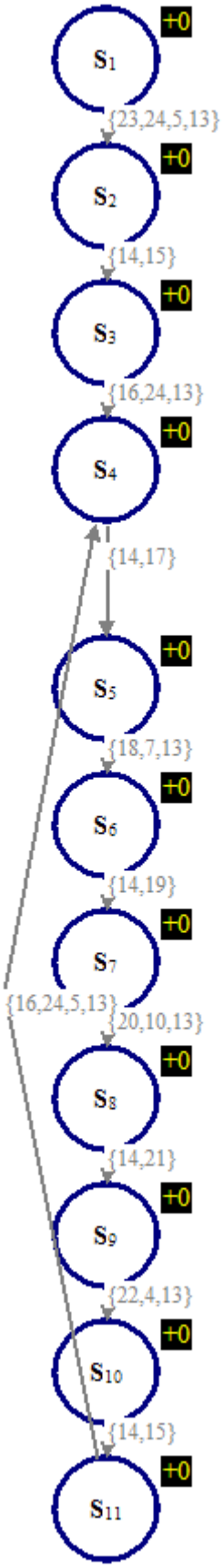

Figure 48 RG of TEST_E_Compare_I_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_DEMUX_ALG

Model Descriptions:

A demultiplexer issues event output signal in according to the condition input signal and is triggered by the event input signal *EI*.

Model Details:

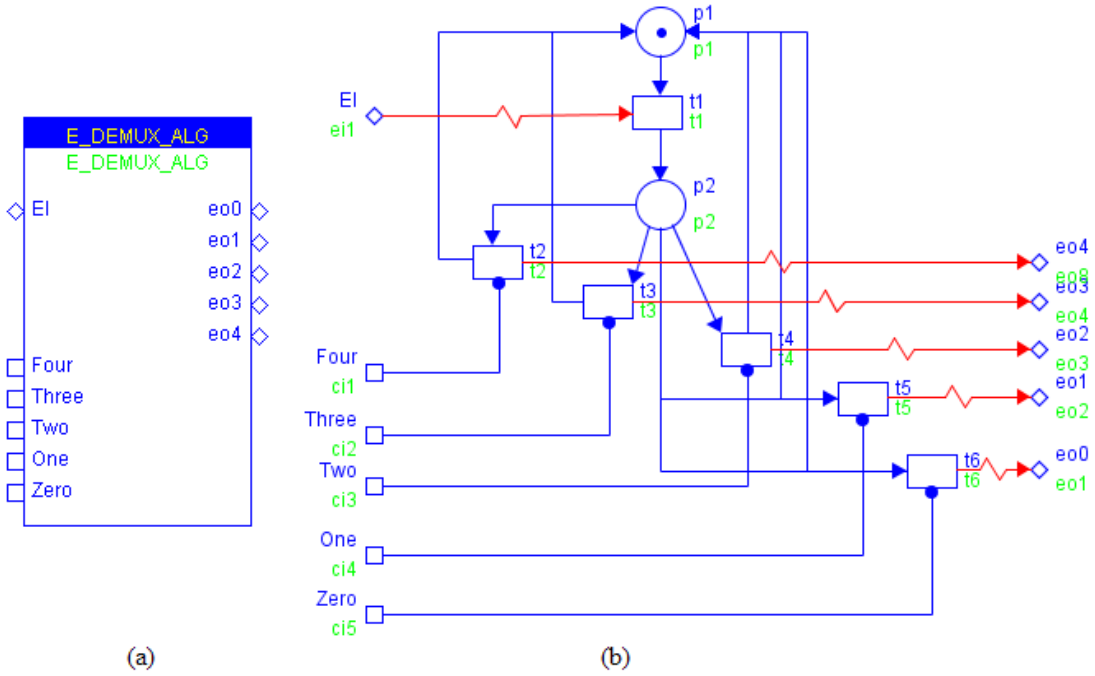

Figure 49 E_DEMUX_ALG.xml

Simple Testing Scenario:

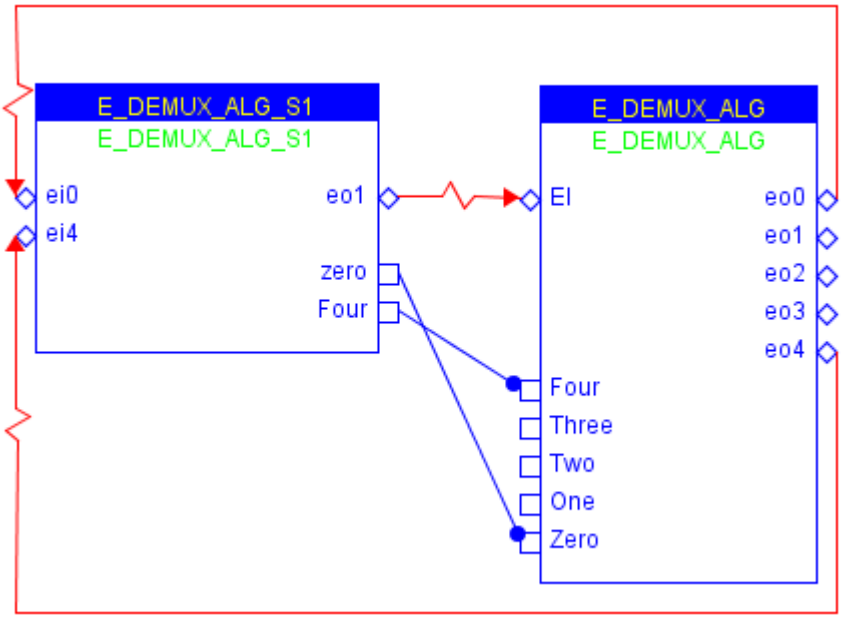

Figure 50 TEST_E_DEMUX_ALG_S1.xml

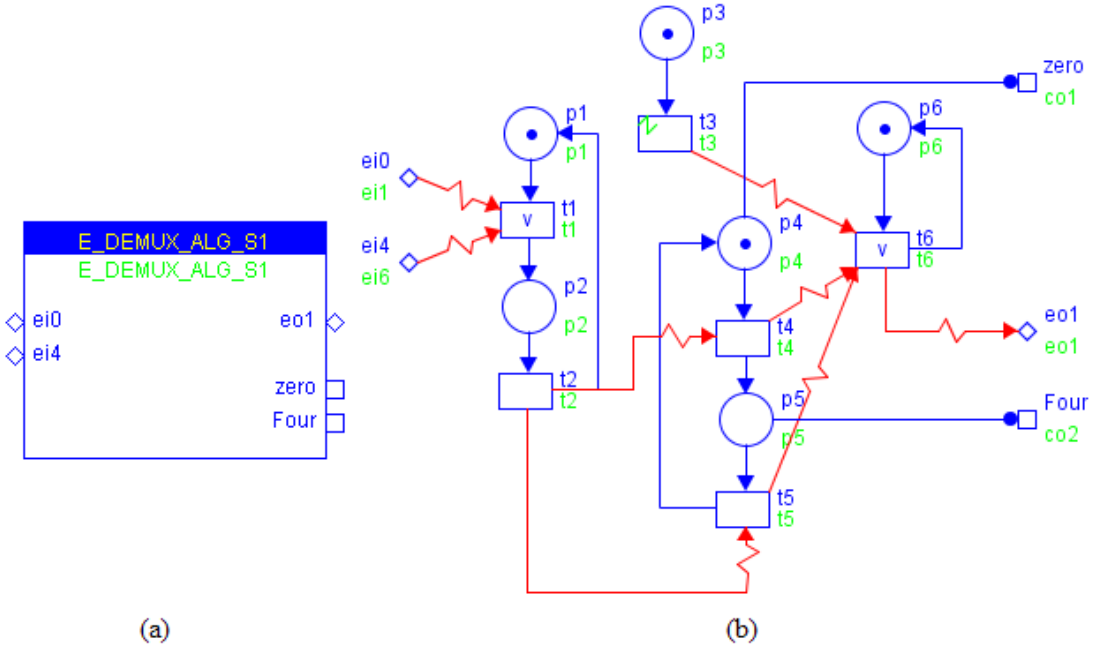

Figure 51 E_DEMUX_ALG_S1.xml

ViVe Reachability Graph:

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_REND

Model Descriptions:

Rendezvous of two event input signals.

Model Details:

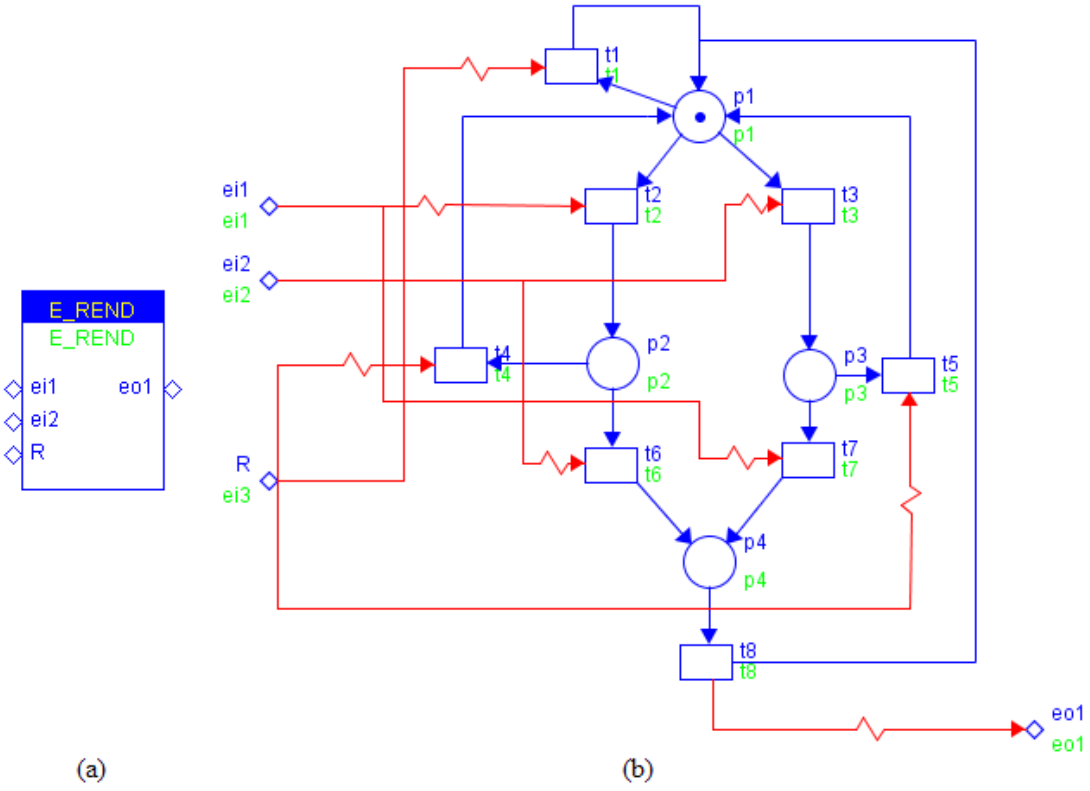

Figure 52 E_REND.xml

Simple Testing Scenario:

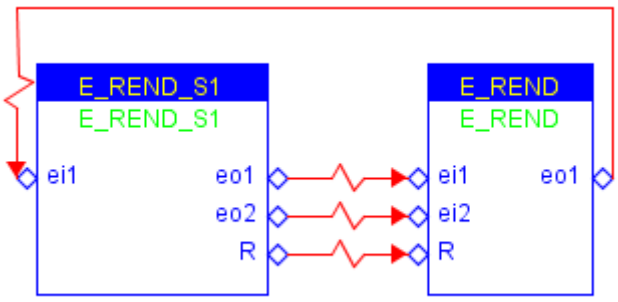

Figure 53 TEST_E_REND_S1.xml

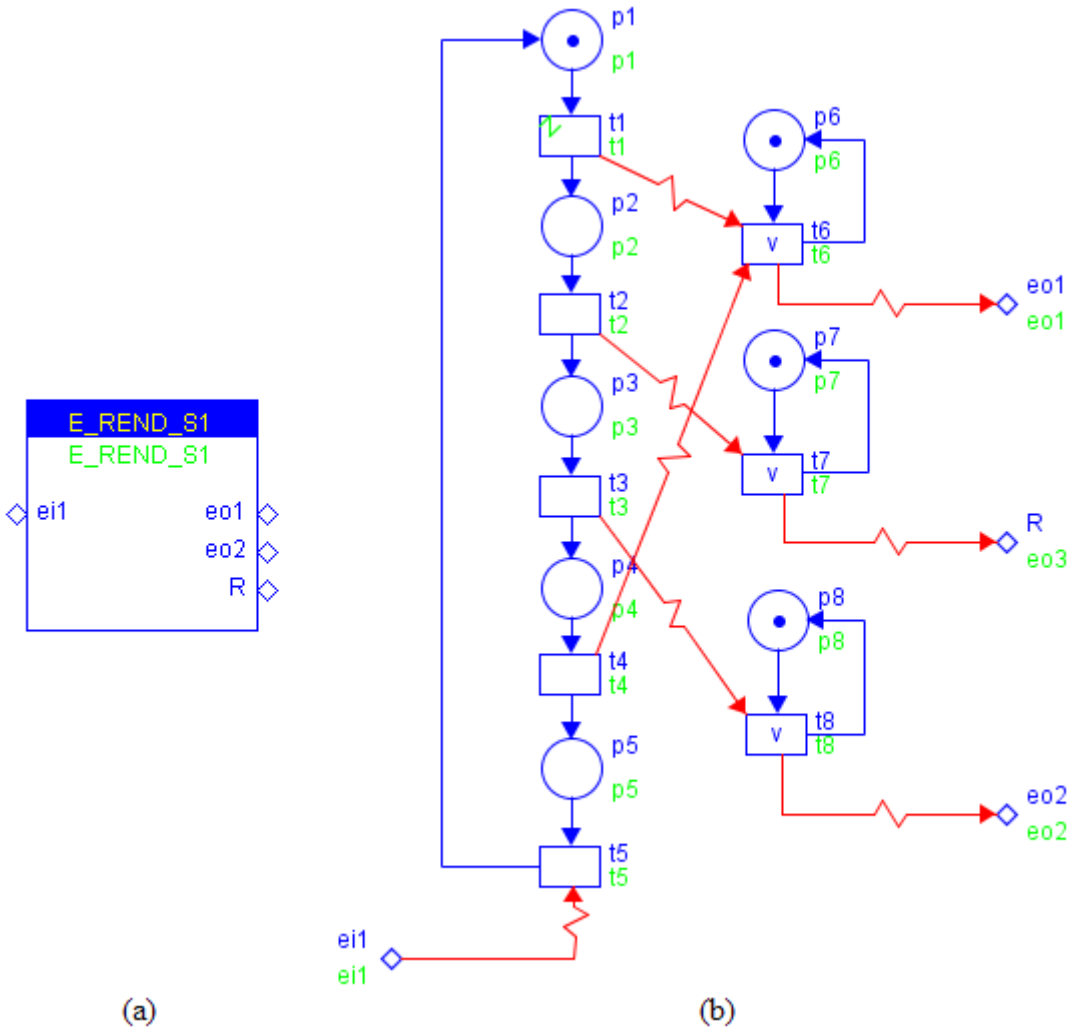

Figure 54 E_REND_S1.xml

ViVe Reachability Graph:

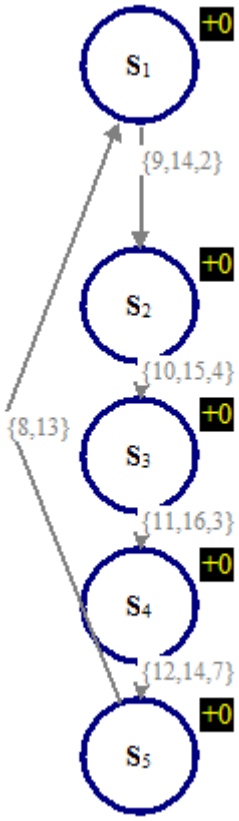

Figure 55 RG of TEST_E_REND_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_SWITCH_ALG

Model Descriptions:

Based on the condition input signal, the switcher issues one of two possible event output signals.

Model Details:

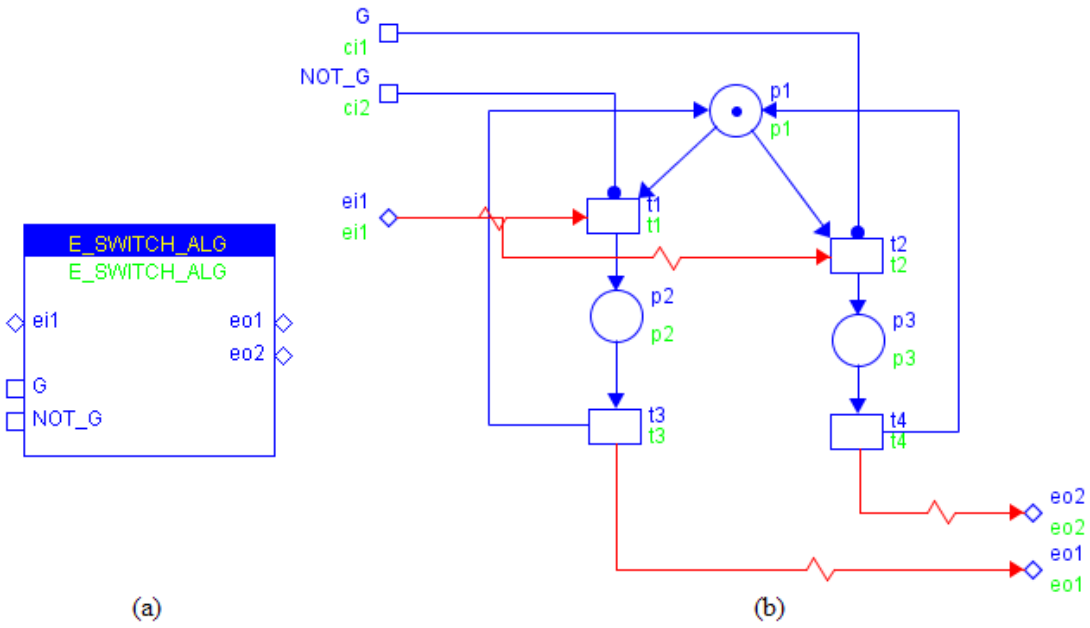

Figure 56 E_SWITCH_ALG.xml

Simple Testing Scenario:

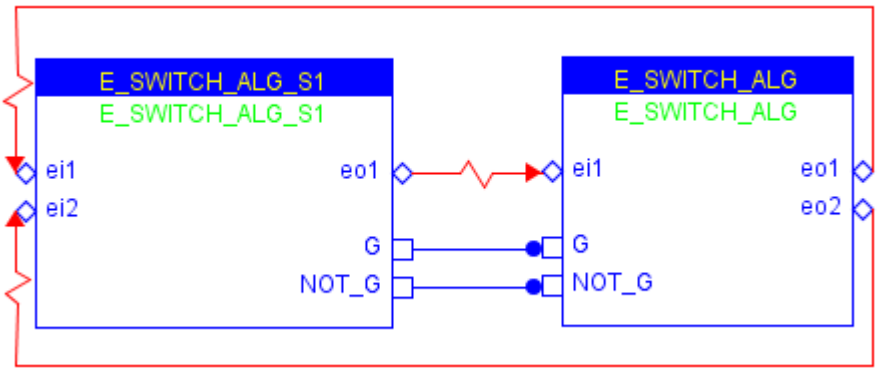

Figure 57 TEST_E_SWITCH_ALG_S1.xml

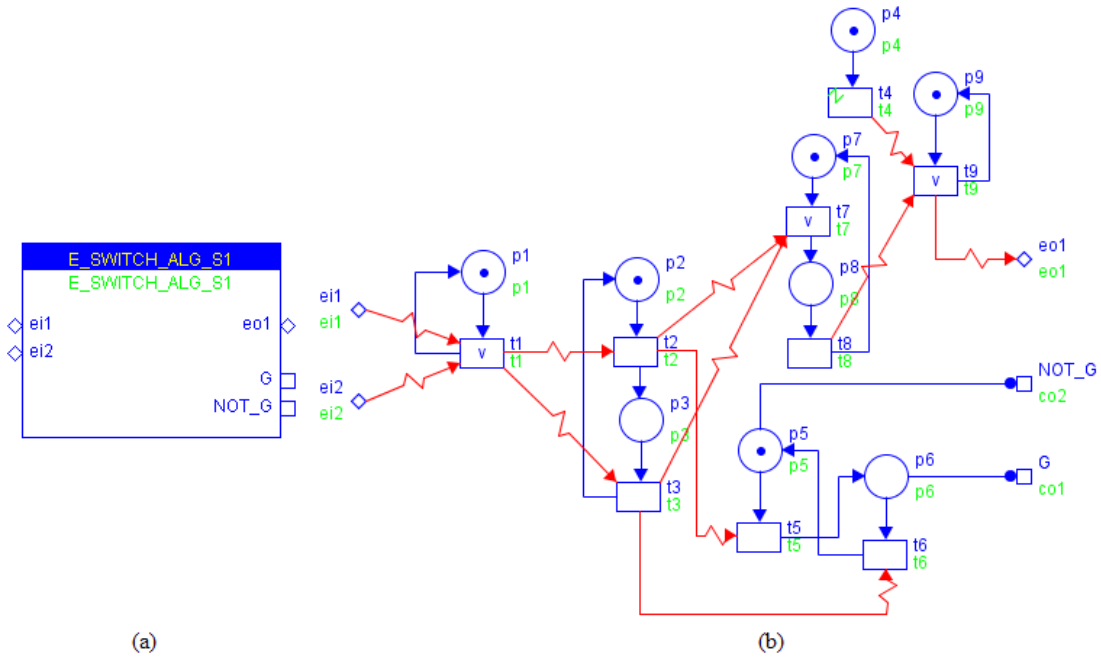

Figure 58 E_SWITCH_ALG_S1.xml

ViVe Reachability Graph:

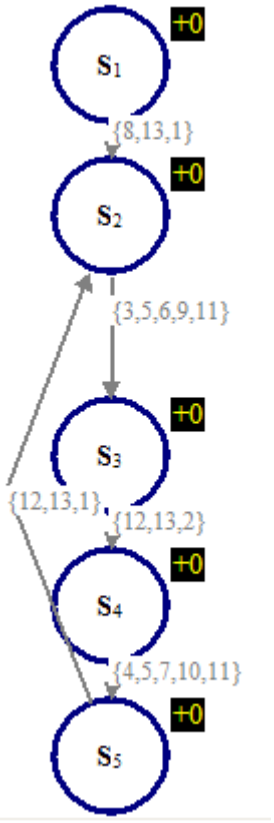

Figure 59 RG of TEST_E_SWITCH_ALG_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: Internal_Boolean

Model Descriptions:

This model models an internal Boolean variable whose value can be set or reset by the event input signals.

Model Details:

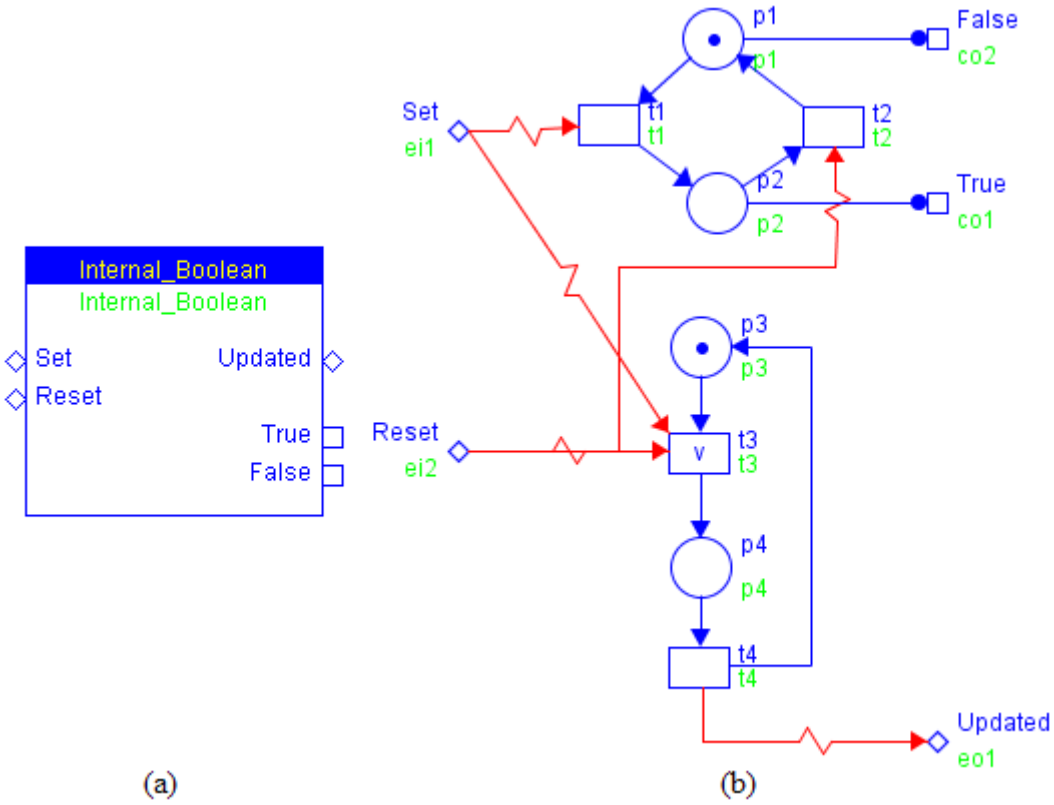

Figure 60 Internal_Bollean.xml

Simple Testing Scenario:

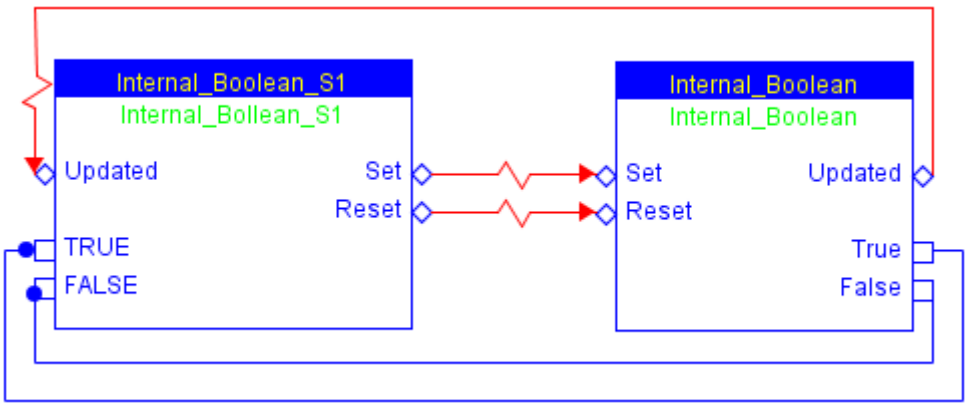

Figure 61 .xml

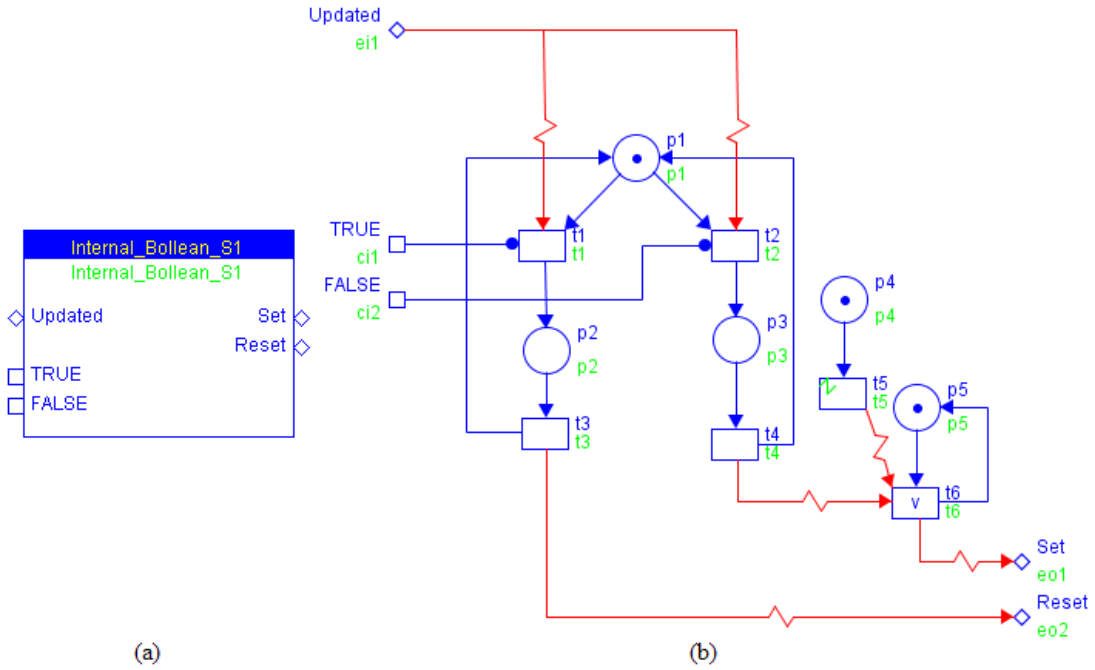

Figure 62 Internal_Boolean_S1.xml

ViVe Reachability Graph:

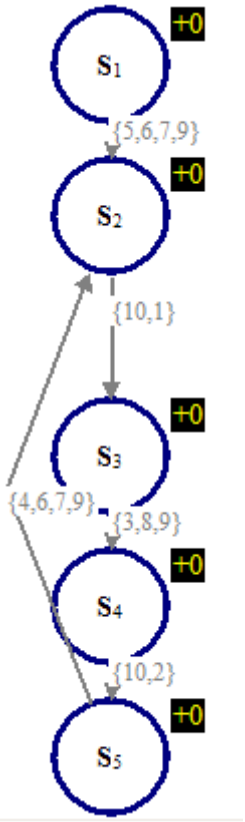

Figure 63 RG of TEST_Internal_Boolean_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_SELECT_ALG

Model Descriptions:

E_SELECT_ALG models a selection between two events. The state diagram of **E_SELECT_ALG** is illustrated below:

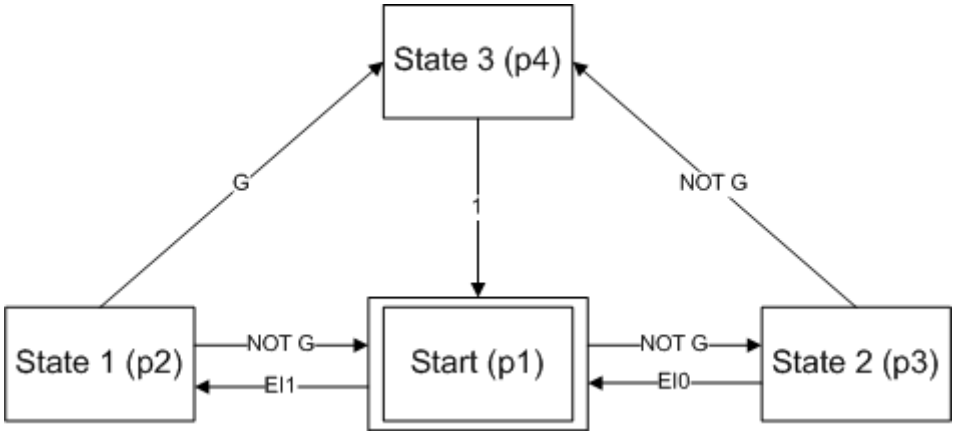

Figure 64 State Diagram of E_SELECT_ALG

Model Details:

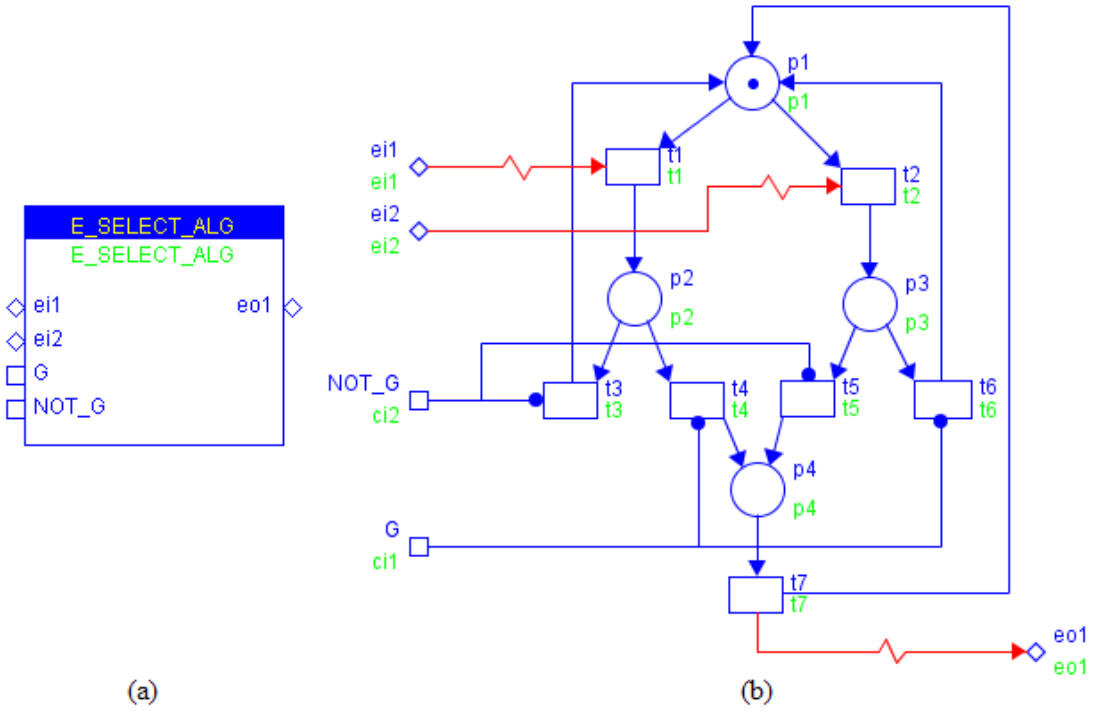

Figure 65 E_SELECT_ALG.xml

Simple Testing Scenario:

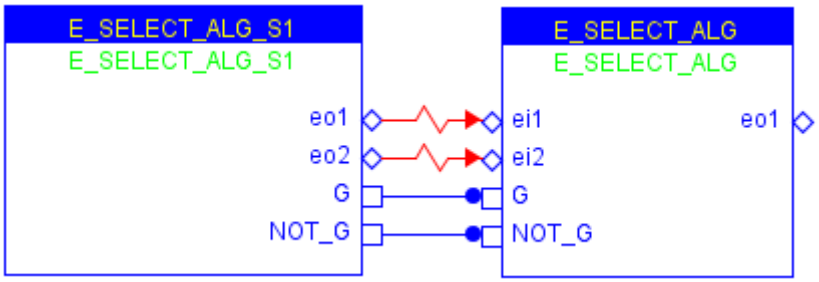

Figure 66 TEST_E_SELECT_ALG_S1.xml

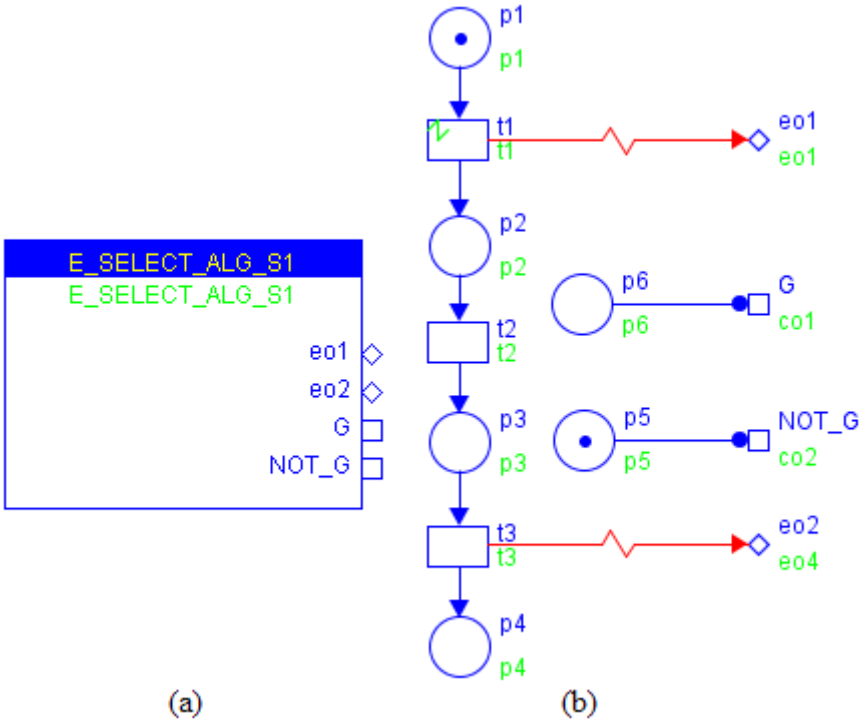

Figure 67 E_SELECT_ALG_S1.xml

ViVe Reachability Graph:

Event output signal *eo1* is issued when *ei1* arrives and *Not_G* is true or when *ei2* arrives and *G* is true.

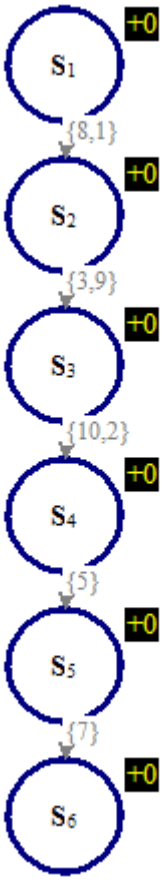

Figure 68 RG of TEST_E_SELECT_ALG_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: TFSwitcher

Model Descriptions:

TFSwitcher updates its internal bistable according to the condition input signals.

Model Details:

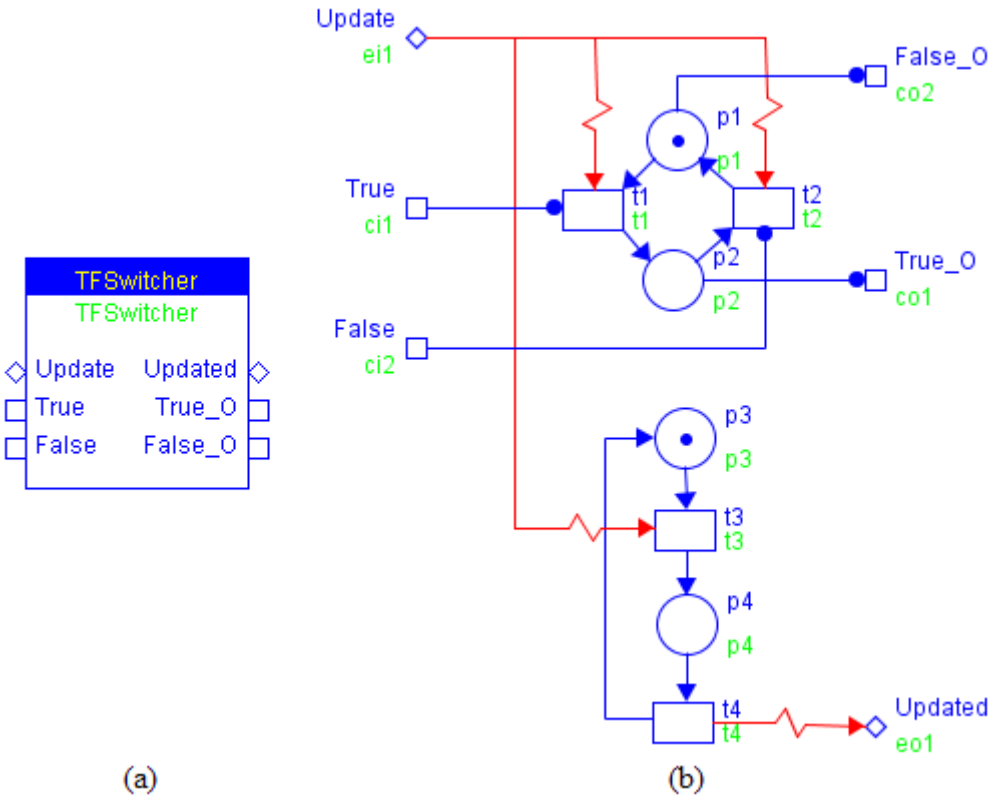

Figure 69 TFSwitcher.xml

Simple Testing Scenario:

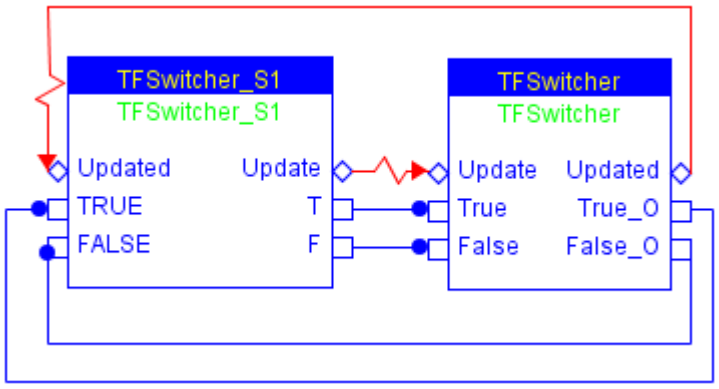

Figure 70 TEST_TFSwitcher_S1.xml

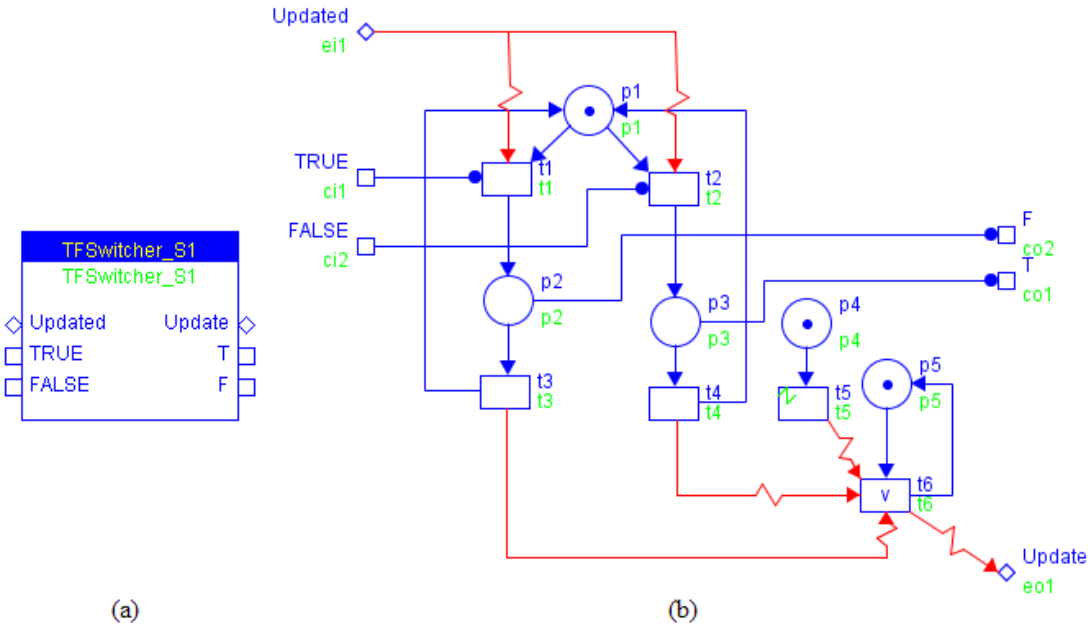

Figure 71 TWSwitcher_S1.xml

ViVe Reachability Graph:

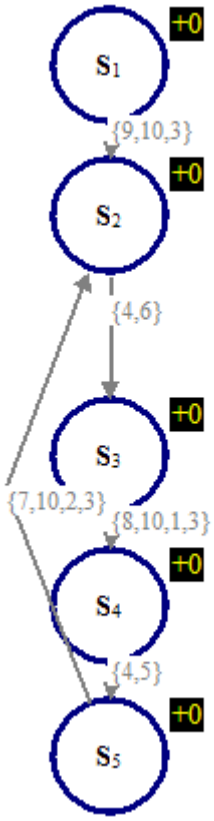

Figure 72 RG of TEST_TFSwitcher

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: Boolean_Input

Model Descriptions:

Modelling of Boolean Input.

Note: Boolean_Input and **Boolean_Output** are identical except the model type name.

Model Details:

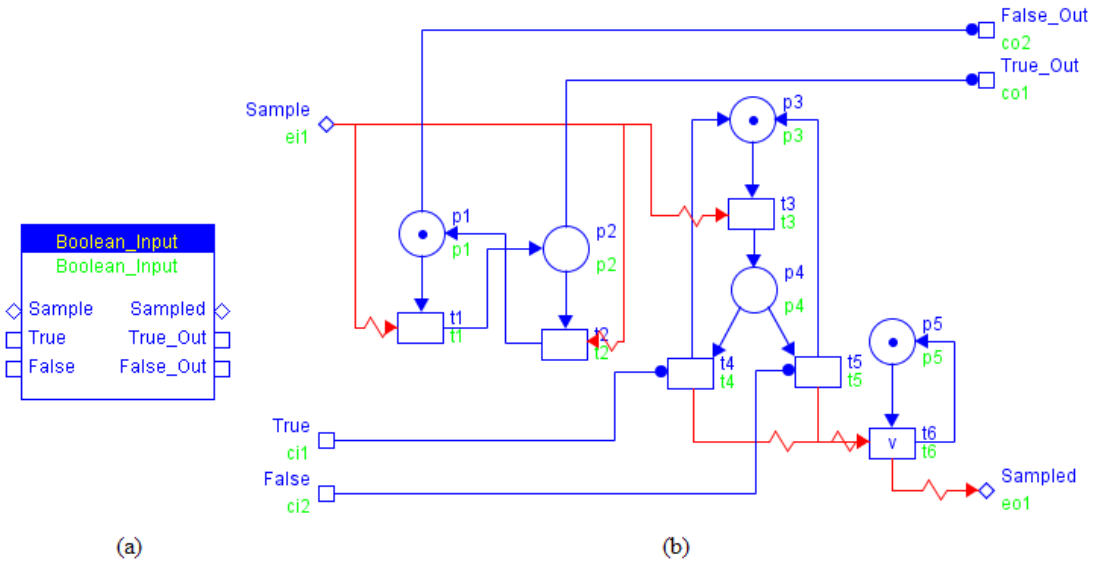

Figure 73 Boolean_Input.xml

Simple Testing Scenario:

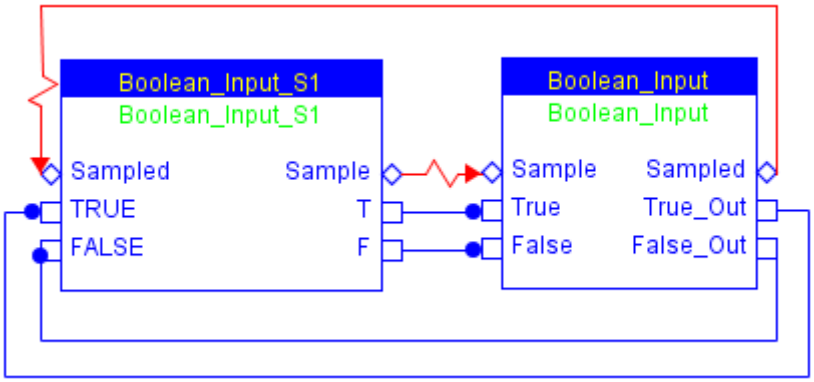

Figure 74 TEST_Boolean_Input_S1.xml

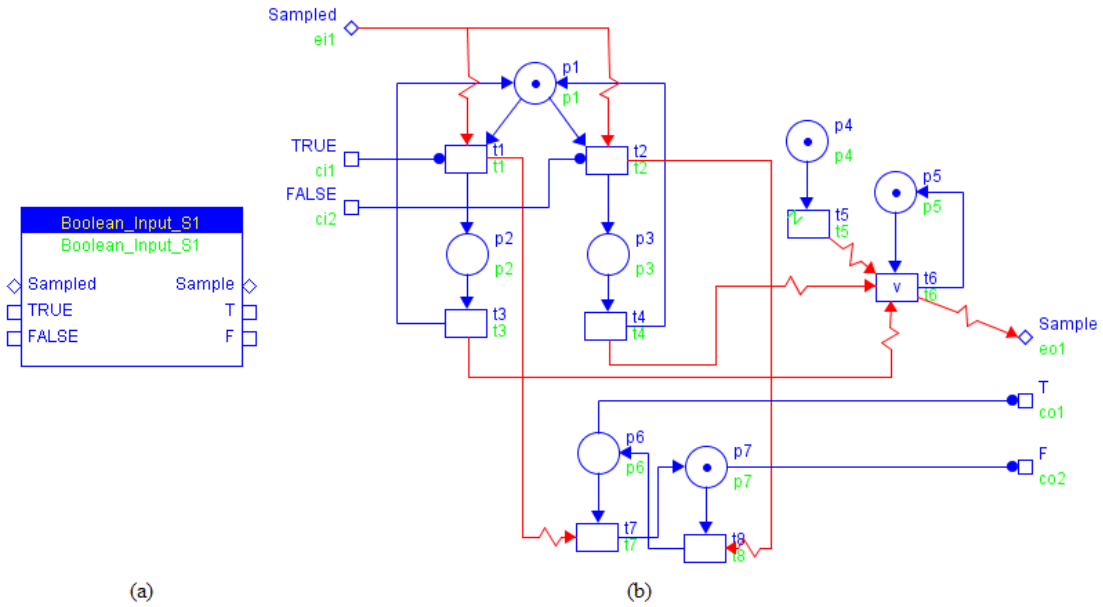

Figure 75 Boolean_Input_S1.xml

ViVe Reachability Graph:

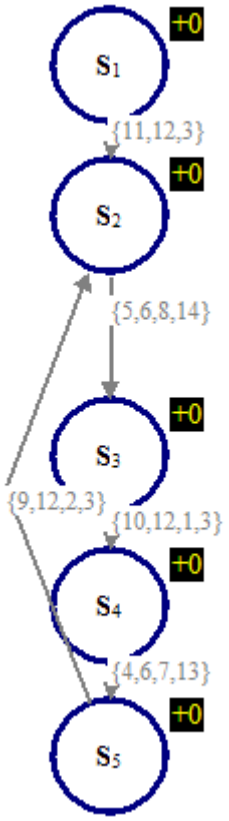

Figure 76 RG of TEST_Boolean_Input_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: BNP

Model Descriptions:

Modelling not present state of token inside a place.

Model Details:

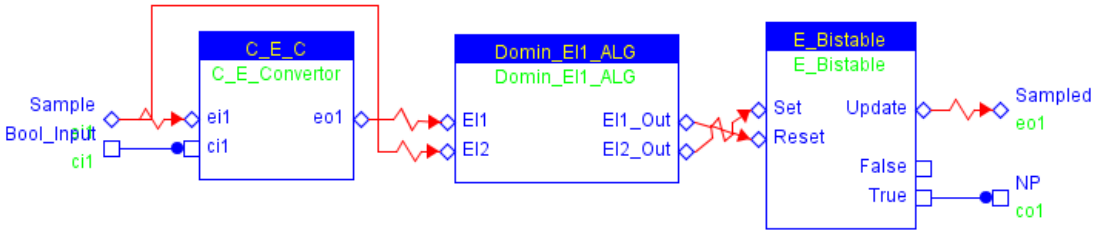

Figure 77 BNP.xml

Simple Testing Scenario:

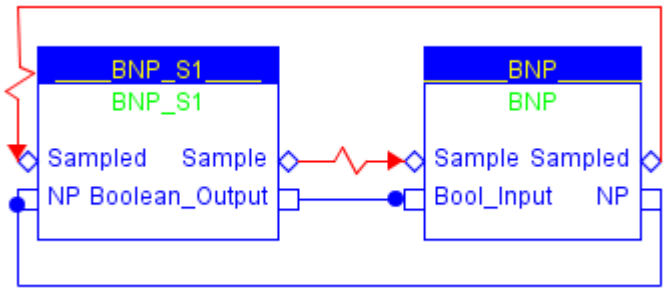

Figure 78 TEST_BNP_S1.xml

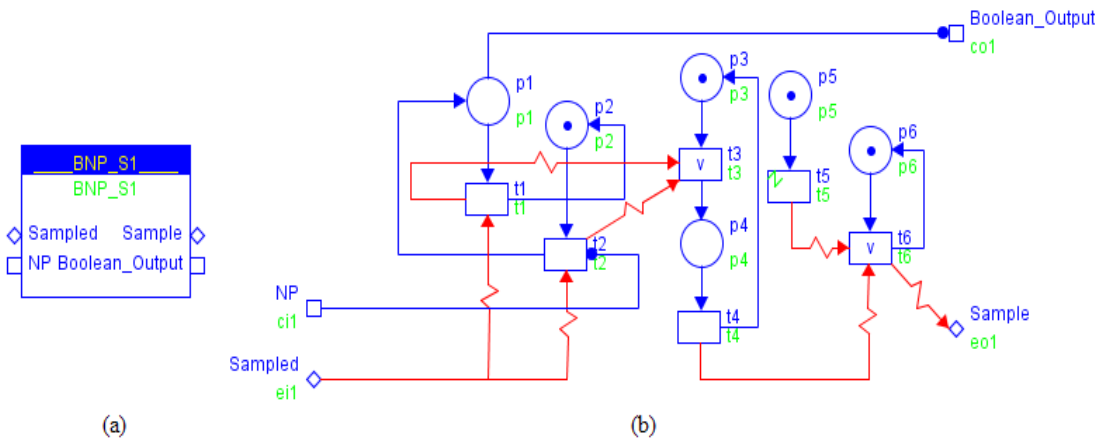

Figure 79 BNP_S1.xml

ViVe Reachability Graph:

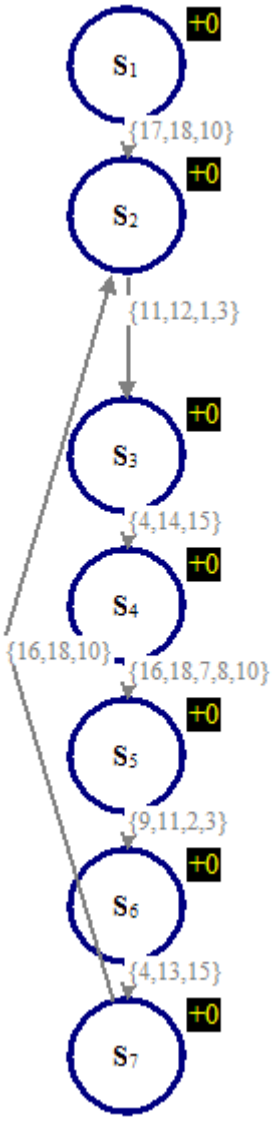

Figure 80 RG of TEST_BNP_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: Uint_Input

Model Descriptions:

Uint_Input models an Unsigned Integer Input without the value of zero.

Model Details:

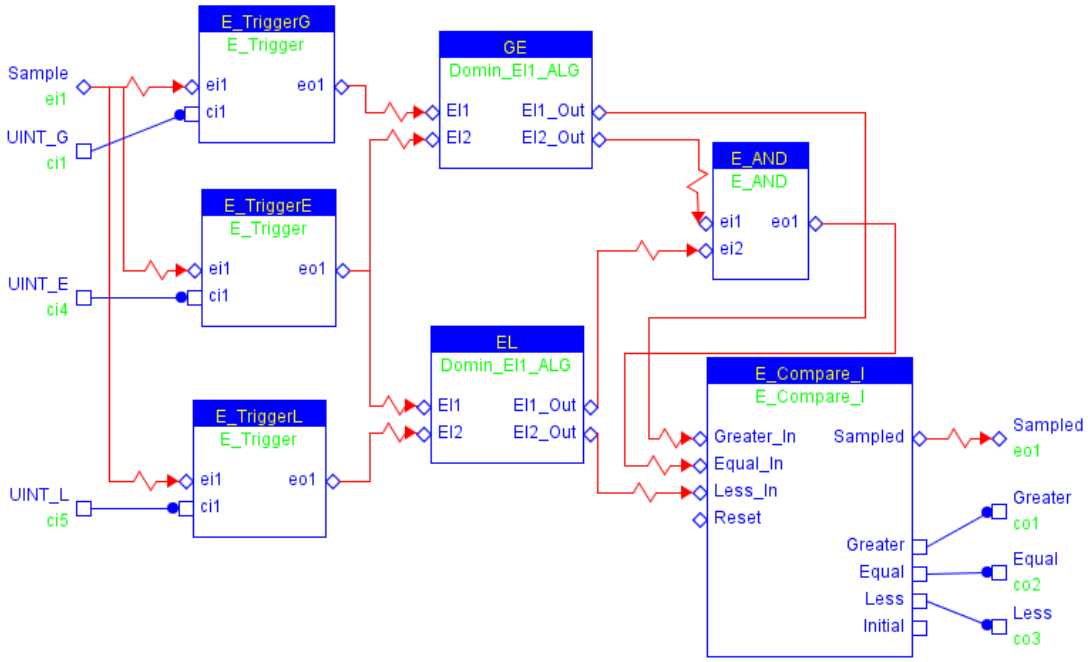

Figure 81 Uint_Input.xml

Simple Testing Scenario:

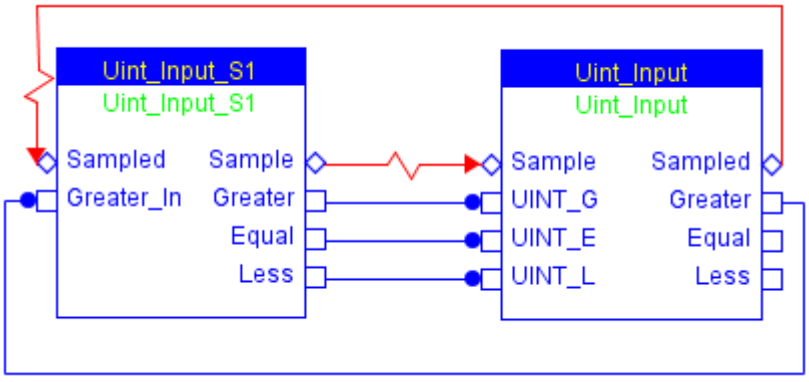

Figure 82 TEST_Uint_Input_S1.xml

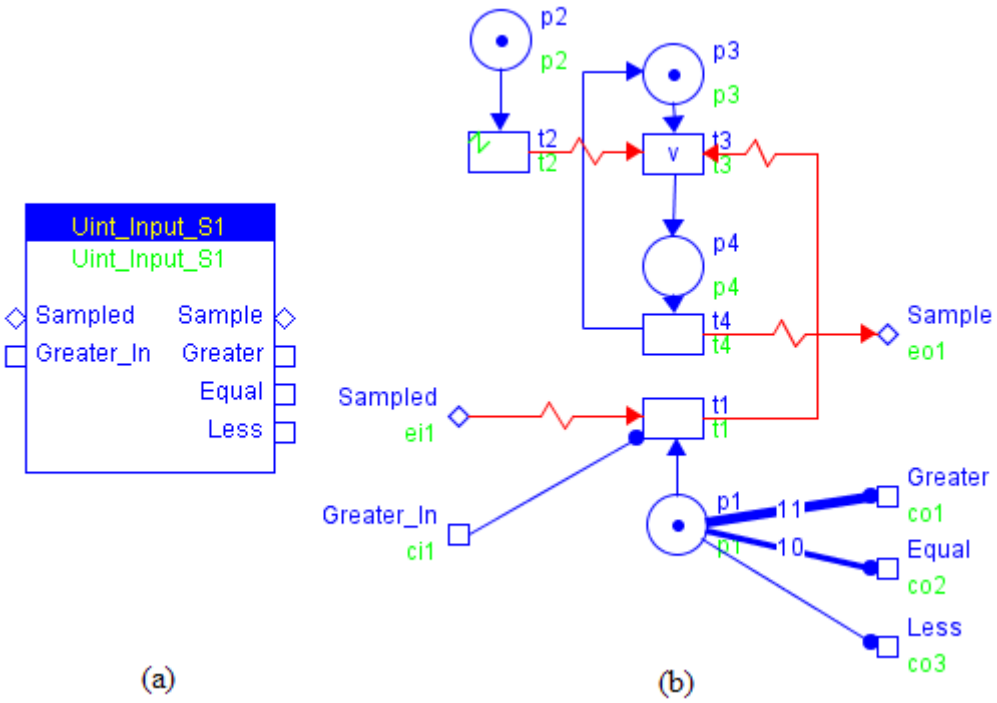

Figure 83 Uint_Input_S1.xml

ViVe Reachability Graph:

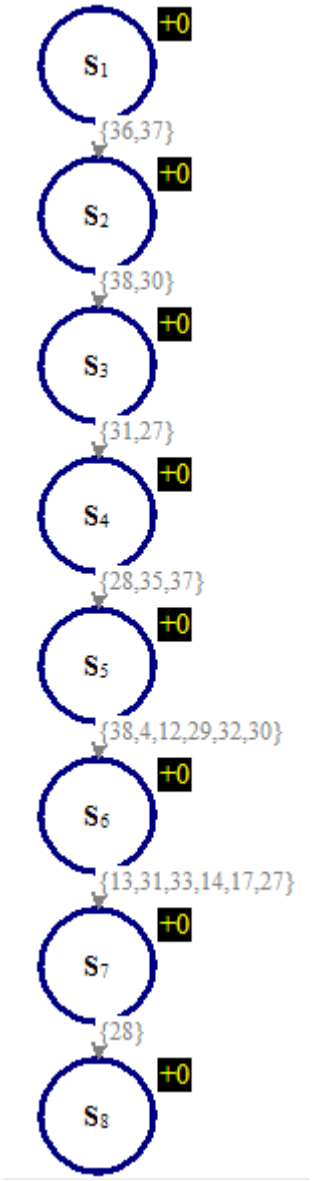

Figure 84 RG of TEST_Uint_Input_S1

Uint_Input_S1.p1 adjusts its token number according to the feedback signal *Uint_Input_S1.Greater_In connected to Uint_Input.Greater.*

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: Uint_Input_Z

Model Descriptions:

Modelling of Unsigned Integer Input with zero.

Model Details:

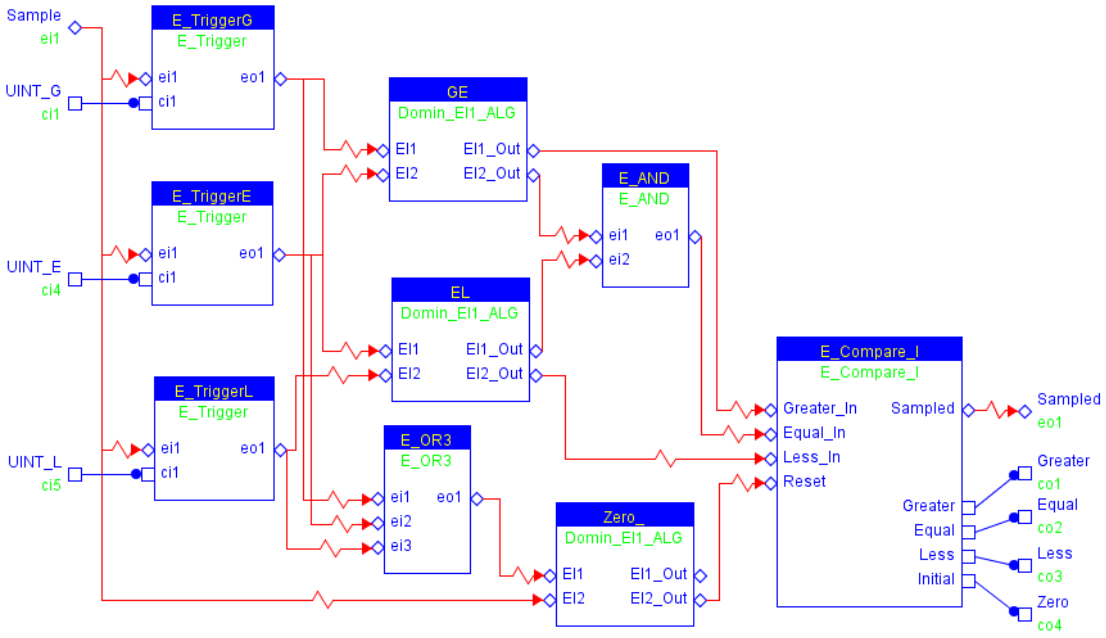

Figure 85 Uint_Input_Z.xml

Simple Testing Scenario:

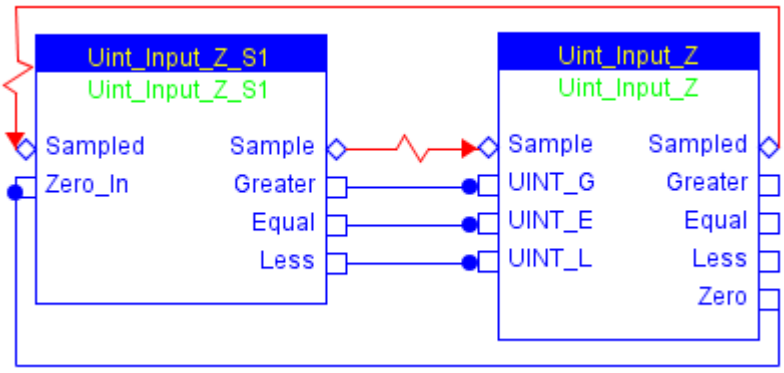

Figure 86 TEST_Uint_Input_Z_S1

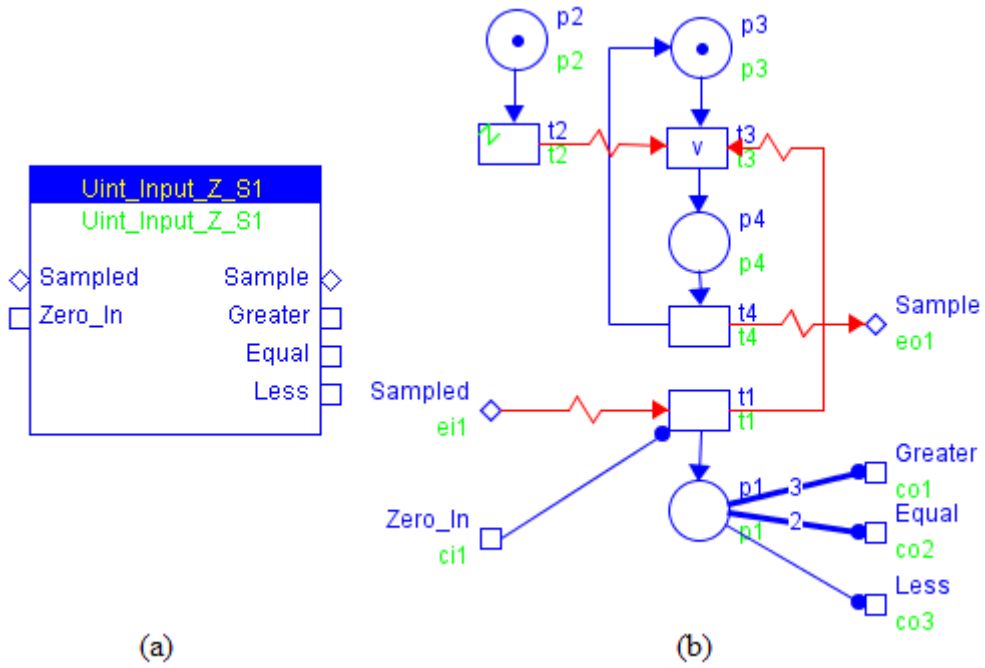

Figure 87 Uint_Input_Z_S1.xml

ViVe Reachability Graph:

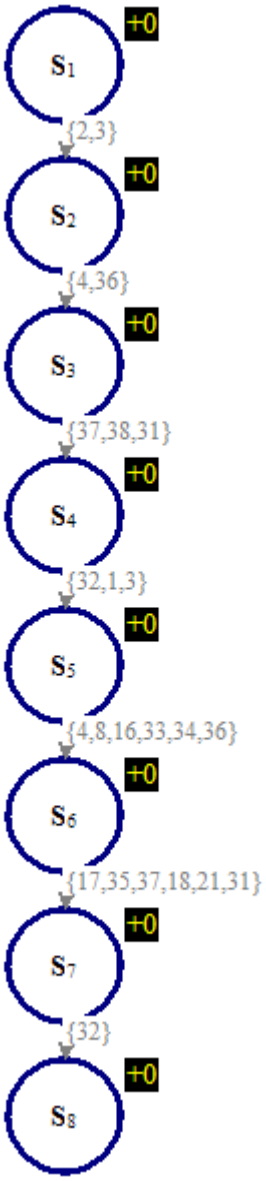

Figure 88 RG of TEST_Uint_Input_Z_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: MUINT_Inputs_Z

Model Descriptions:

Modelling of multiple UINT inputs with zero.

Model Details:

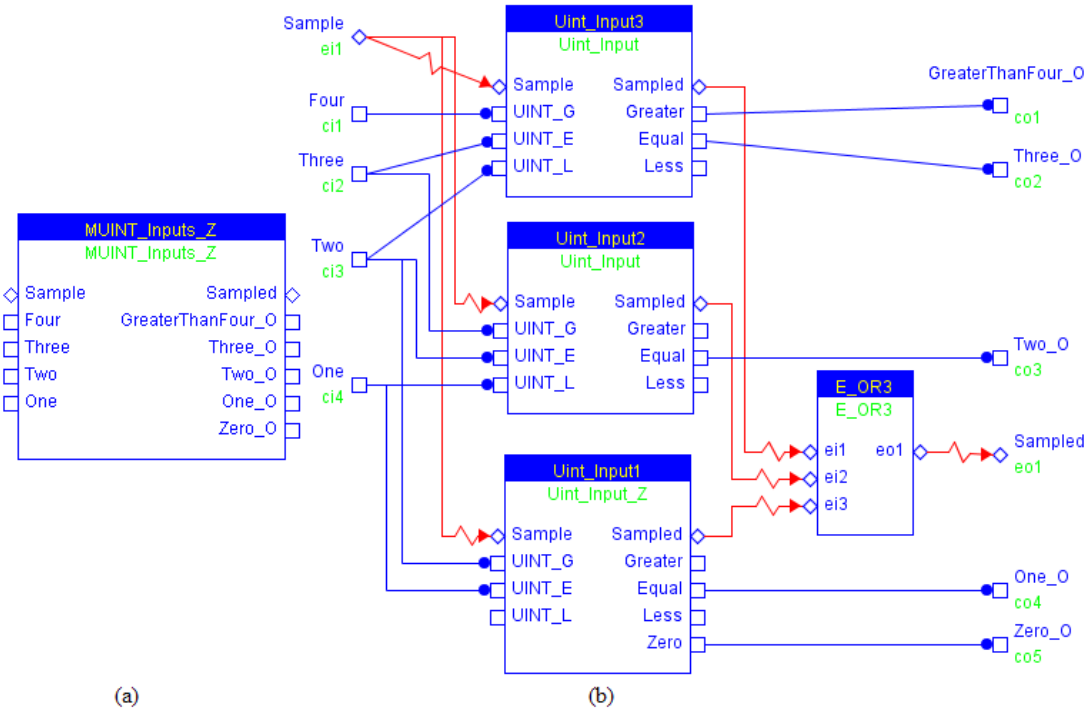

Figure 89 MUINT_Inputs_Z.xml

Simple Testing Scenario:

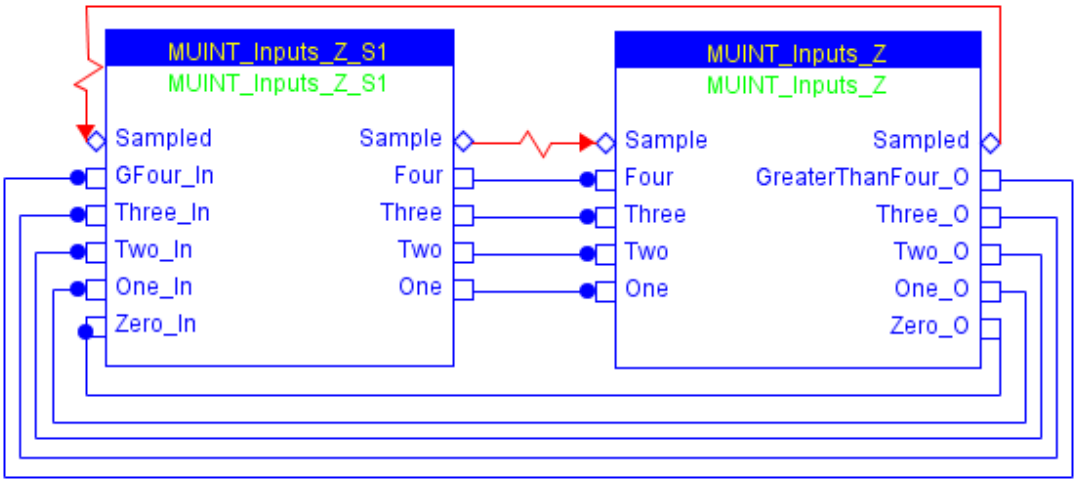

Figure 90 TEST_MUINT_Inputs_Z_S1.xml

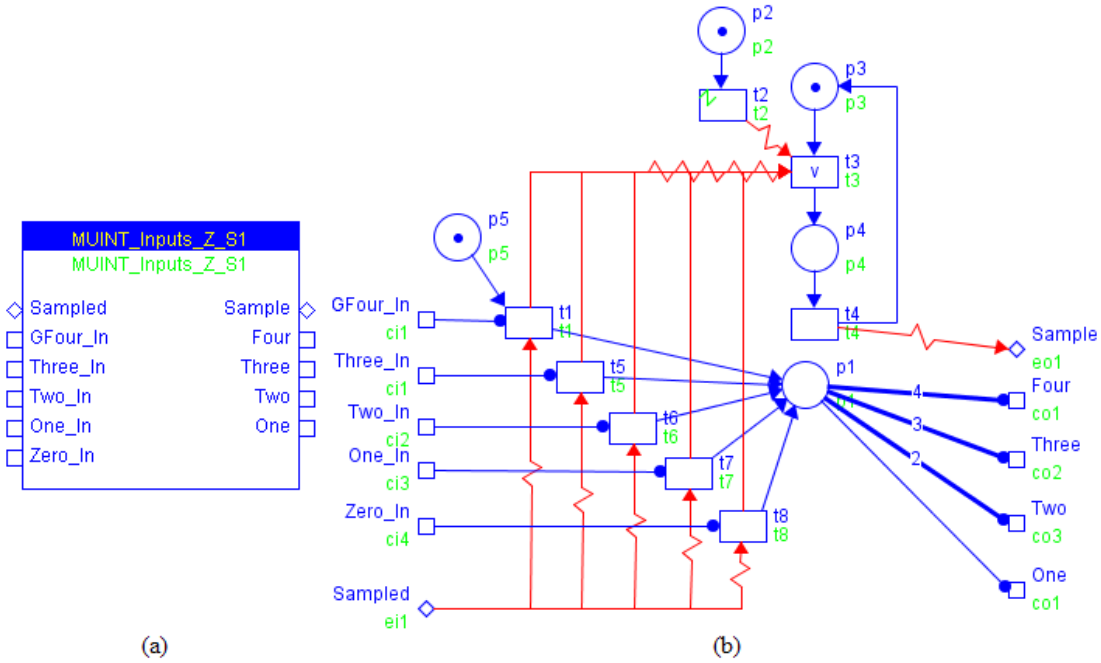

Figure 91 MUINT_Inputs_Z_S1.xml

ViVe Reachability Graph:

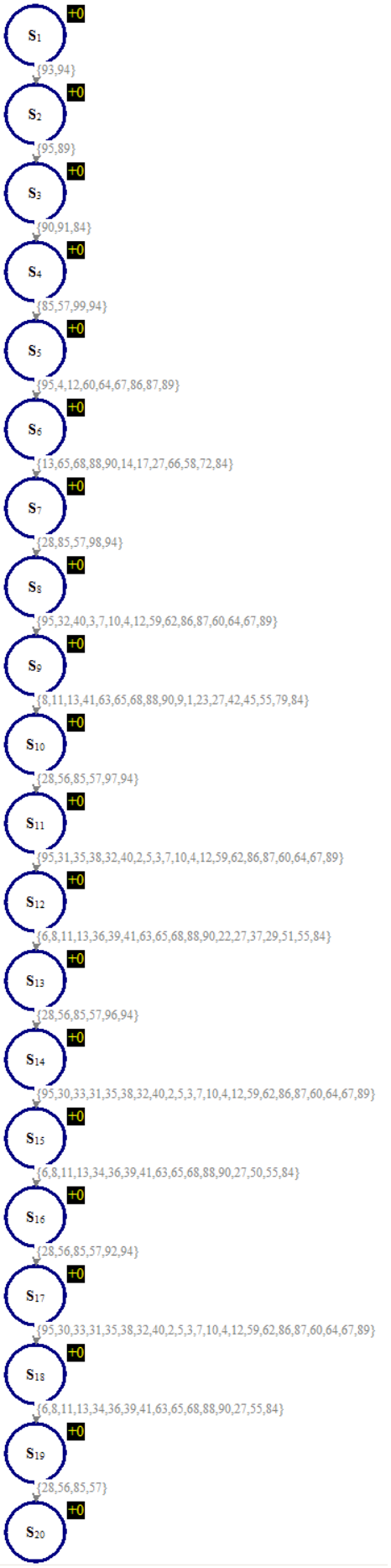

Figure 92 RG of TEST_MUINT_Inputs_Z_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: MSampled

Model Descriptions:

MSampled models rendezvous of M event input signals without the reset option. The following diagram illustrates a rendezvous of two event input signals, which can be extended to M inputs easily. The event output signal *SO* will be issued upon the occurrence of *Sampled1* and *Sampled2*.

Model Details:

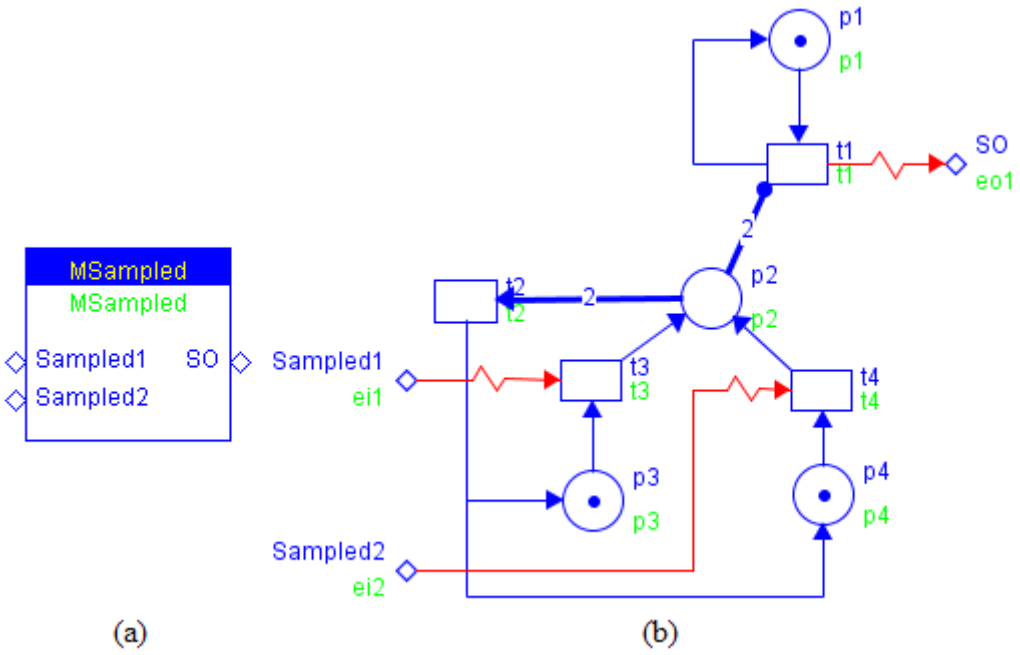

Figure 93 MSampled.xml

Simple Testing Scenario:

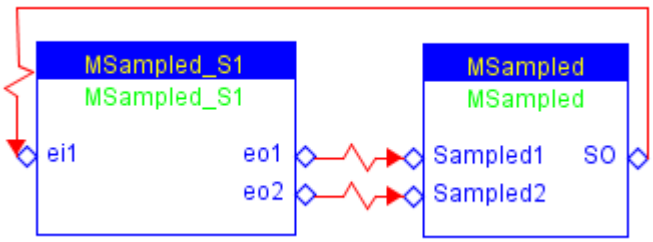

Figure 94 TEST_MSampled.xml

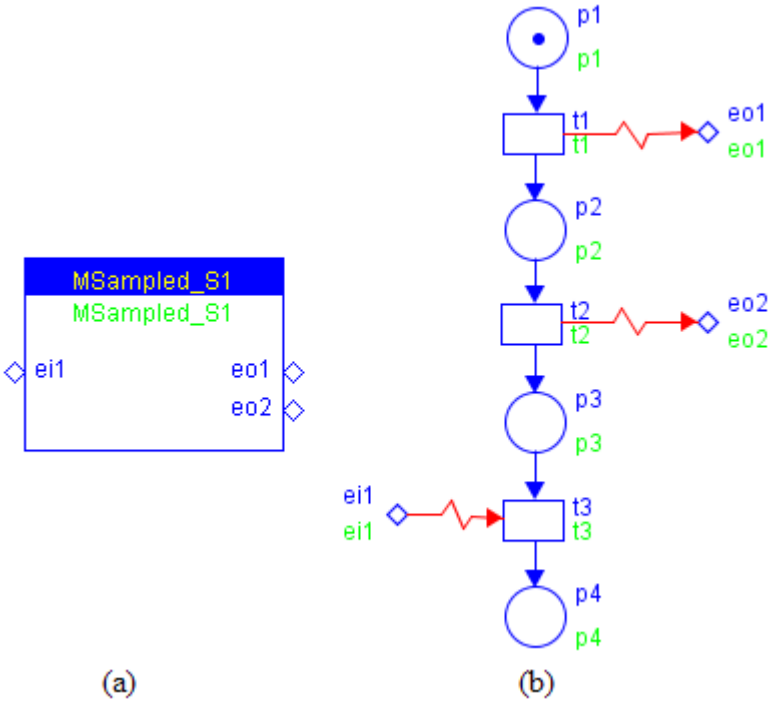

Figure 95 MSampled_S1.xml

ViVe Reachability Graph:

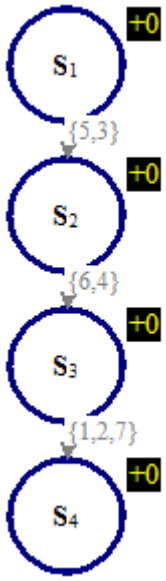

Figure 96 RG of TEST_MSampled_S1

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_F_TRIG

Model Descriptions:

E_F_TRIG models Boolean falling edge detection.

Model Details:

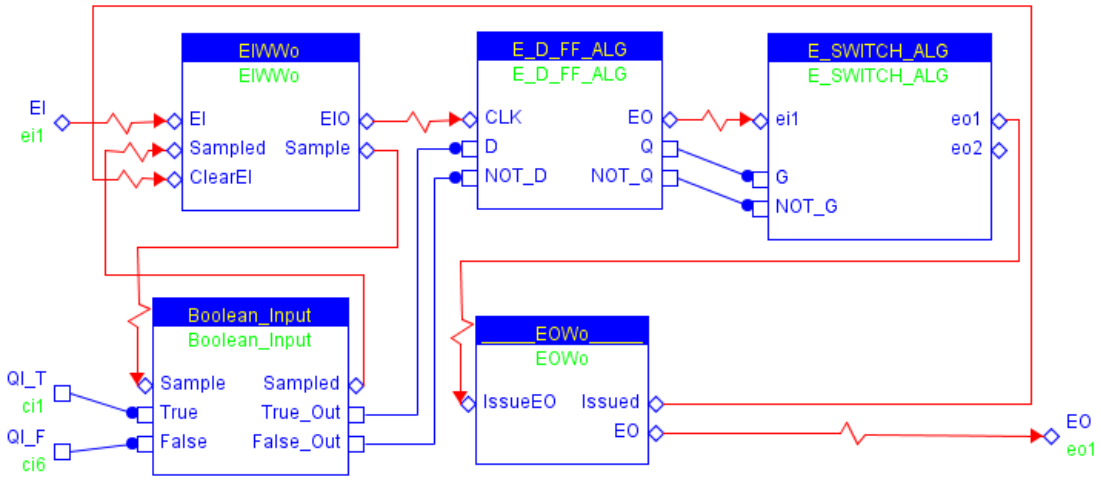

Figure 97 E_F_TRIG.xml

Simple Testing Scenario:

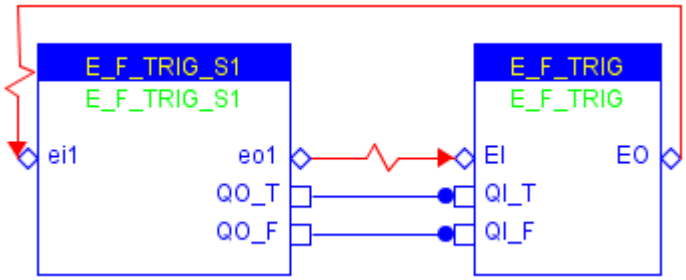

Figure 98 TEST_E_F_TRIG_S1.xml

Figure 99 E_F_TRIG_S1.xml

ViVe Reachability Graph:

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: E_R_TRIG

Model Descriptions:

E_R_TRIG models Boolean rising edge detection.

Model Details:

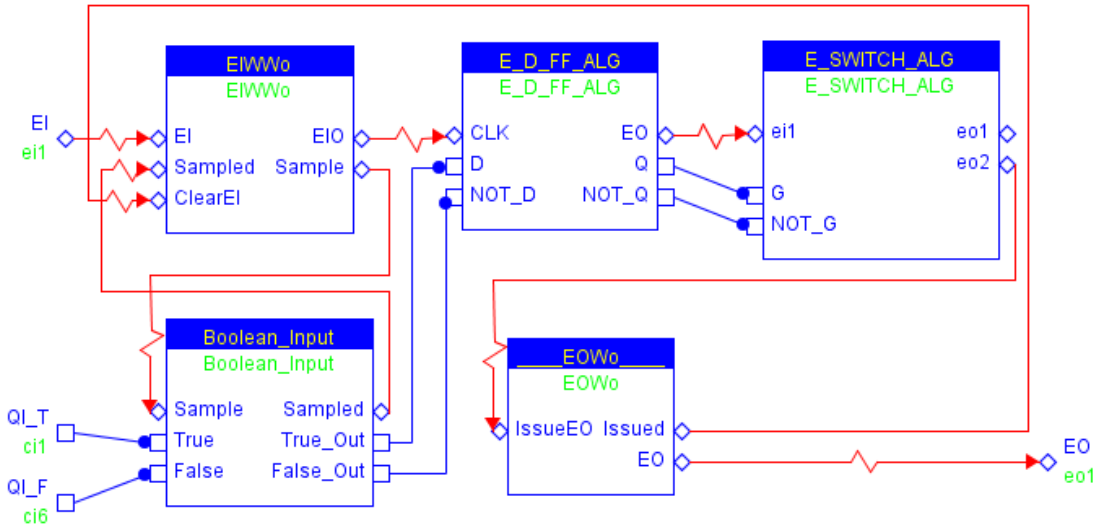

Figure 100 E_R_TRIG.xml

Simple Testing Scenario:

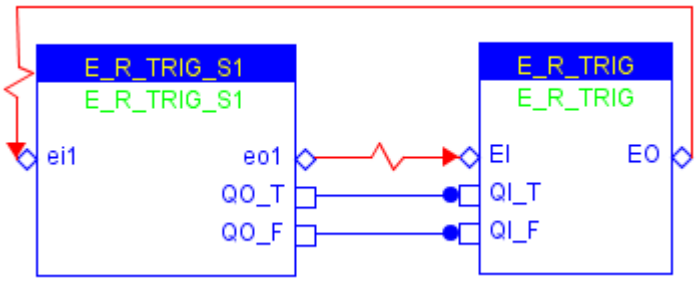

Figure 101 TEST_E_R_TRIG_S1.xml

Figure 102 E_R_TRIG _S1.xml

ViVe Reachability Graph:

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:

Model Name: <Sample>

Model Descriptions:

Model Details:

Simple Testing Scenario:

ViVe Reachability Graph:

Model Verification:

Properties to be checked:

CTL formulae:

Miscellaneous:
