DISCRETE EVENT SYSTEMS

Control of an Ozone Generation System

By,

M.MD. SOHAIL,

Student no: 40187593.

Submitted to,

Professor. SHAHIN HASTRUDI ZAD.

Abstract:

The following project deals with designing the local controller of a set of components contributing to the functioning of an Ozone Generation system.The local controller is a component of the system which can dictate the functioning of each component.The local controller is designed using “Supervisory control theory [2] ” which deals with the subject of Discrete Event System , DES (A set of Events and States interconnected in a proper sequence) acting as a supervisor for an another Discrete Event system. The project deals with various events or a set of occurances both controllable and uncontrollable by the local controller.The Ozone Generation System consists of multiple valves,sensors and a single Power Supply Unit arranged as shown in the figure 2.Each component can be represented as its own unique automata(a type of model used to represent a DES).These automata also interact with each other which also must be catered too by the local controller and the plant model.The plant model is a type of automata which represents all the possible events of the Ozone Generation System along with its interactions.The local controller acts as a supervisor to this plant model for its operation.Thus, this project deals with stepwise construction of the plant model, the local controller(supervisor) and its working in the Ozone Generation System specified in problem.For the following project MATLAB software was used to implement the design of the automata mentioned in the following project.The toolbox named DECK Discrete Event Control Kit is used which contains a set of commands which can be used to create and perform operations with automata.

Introduction:

Discrete Event Systems models help the engineers to well plan out the events and states of a process and can help assess any losses in case of failure. Automata is a type of model used to represent Discrete Event Systems. Automata consists of states represented with a circle with a name of the state in it and events are represented by directional arrows pointing towards other states and originating from another state. Automata also consists of other components which mark the states. A singular arrow with no origin pointing a state represents initial state, the arrow protruding from a state with no destination represents marked state. The following can be seen in Figure 1 which represents working of a valve.

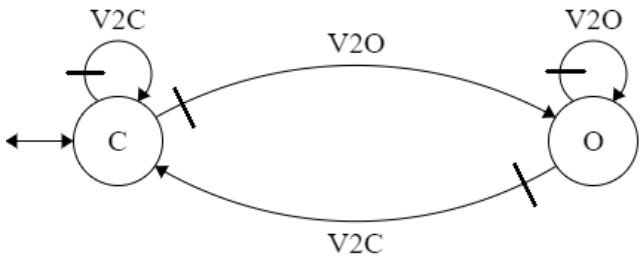


Figure 1: An automata of a valve

Here, “C”, “O” are states and V2O, V2C are events.

States:

C-Valve Closed

O-Valve Opened

Events:

V2O-Opening the valve

V2C-Closing the valve

Here, the initial and marked state is C as we assume valve to be closed initially. The small mark on the event arrows signifies whether the event is controllable or not and here, we assume the events V2O, V2C to be controllable. Once we understand what an automaton is, we begin the project using a MATLAB toolbox named DECK (Discrete Event Control Kit) [1]. This toolbox enables

users to code automatons on MATLAB and perform various operations. The following project follows these steps:

1) Modelling of each component of the Ozone Generation System (OGS) from the figure 2
2) Finding the interaction of the components of the Ozone Generation System
3) Creating a plant model which represent all possible outcomes of the OGS
4) Modelling automata which define the specifications mentioned in the problem.
5) Creating Supervisor automata which can dictate the plant model to follow the desired specifications.
6) Testing the supervisor automata created and show its working.

Problem Statement:

The problem requires the project to create a local supervisor which follows the specifications mentioned below:

1) OGS must follow a proper sequence of Start-up and Shutdown and perform all the steps required for Start-up and Shutdown. The Start-up and Shutdown begin by the commands issued by the master controller, but the commands must be ignored in between both start-up and shutdown sequence.
2) The OGS has to make sure there is adequate supply of Oxygen during a shutdown sequence until Ozone's concentration becomes low.

Plant Model:

To create the required local supervisor, we must first create the plant model, the plant model is created by the following steps:

1) Design the automata of individual components mentioned in the Figure 2
2) Finding interactions between components
3) Performing sync of all the automata designed. Sync being merging of specified automata together.

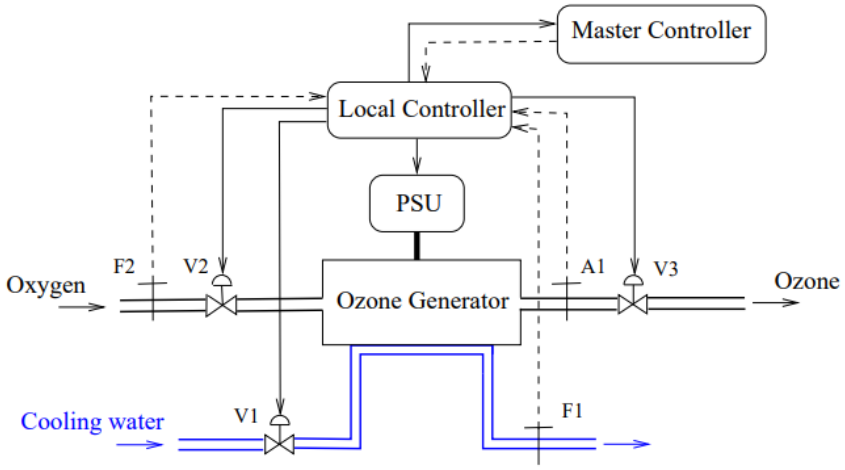


Figure 2: Ozone Generation System

Components:

Valves:

The problem specifies the valves to have 2 states Closed and Open and 2 Events, Opening of the Valve, Closing of the Valve. The figures 3,4 and 5 are the automata models of the three valves present in the OGS showed in Figure 2. As we observe from these automata, the valve is opened when the opening event happens and vice versa but, if closing is triggered for already closed valve is ignored. The following events are controllable hence, the event arrows possess the mark in their automata. The valves are assumed to be closed initially.

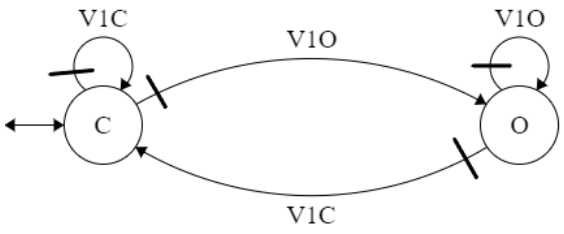


Figure 3: Valve 1

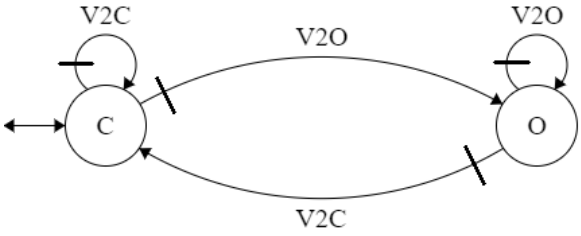


Figure 4: Valve 2

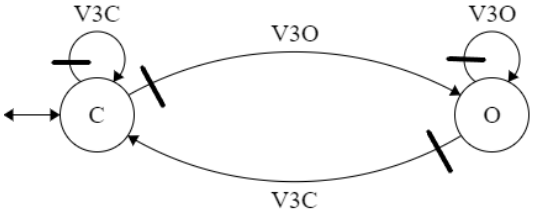


Figure 5: Valve 3

States:

C-Valve Closed

O-Valve Opened

Events:

V2O-Opening the valve 2

V2C-Closing the valve 2

V3O-Opening the valve 3

V3C-Closing the valve 3

V1O-Opening the valve 1

V1C-Closing the valve 1

Flow sensors:

There are two flow sensors present in the OGS F1, F2, these sensors detect if there is high or low passage of fluid within their respective pipes. The automata of these sensors are shown in Figures 6 and 7. The sensors have 2 states and 2 types of events mentioned below. These events cannot be controlled as they are readings.

States:

L-Low reading

H-High reading

Events:

F1H-High flow in pipe of F1

F1L-Low flow in pipe if F1

F2H-High flow in pipe of F2

F2L-Low flow in pipe if F2

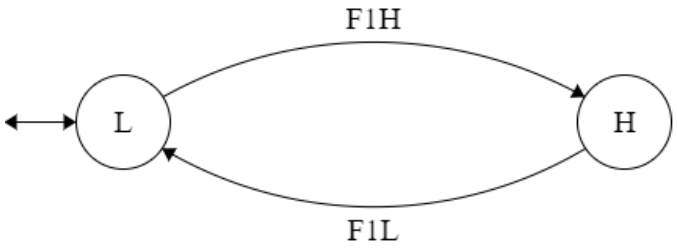


Figure 6: F1 sensor

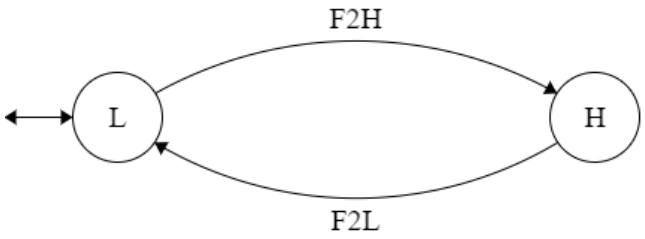


Figure 7: F2 sensor

Power Supply Unit:

There is a Power Supply Unit (PSU) in OGS. The following component provides power to the OGS and contains 2 states, On state and Off state. The events of PSU are PSUON turning PSU on and PSUOFF turning PSU off. The automata of PSU are mentioned in figure 8. The following events are controllable hence are marked in the automata mentioned in the figure 8.

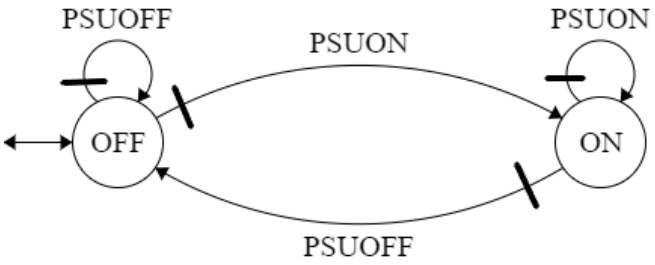


Figure 8: Power Supply Unit

States:

OFF-The PSU is OFF

ON-The PSU is ON

Events:

PSUON-The PSU is being turned on

PSUOFF-The PSU is being turned off

Analyser:

There is an Analyser in OGS which checks for high or low concentration of Ozone and changes the reading. The Analyser consists of 2 states, one for low concentration and one for high. The automaton of Analyser is shown in figure 9. These events cannot be controlled as they are readings.

States:

L-Low reading

H-High reading

Events:

AH-Ozone concentration is high

AL-Ozone concentration is Low

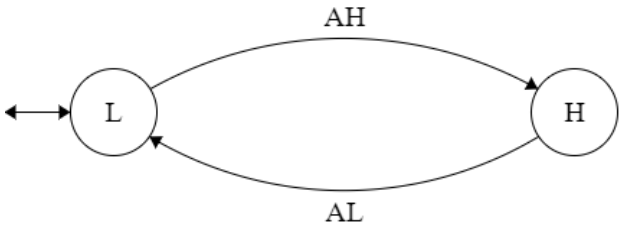


Figure 9: Analyser

Local Automaton:

A part of local controller must be capable of sending messages of Start up finish and Shutdown finish to the master controller of the OGS.As the order of messages sent doesn't matter, we consider a singular state. The events here represent messages sent to all the components when each sequence is done. The message events are controllable by local controller. The automaton of this requirement is named local automata and is shown in figure 10.

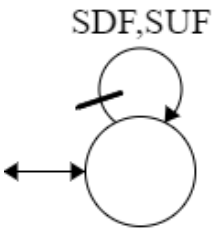


Figure 10: Local automaton

Events:

SUF-Start-Up Finish (End)

SDF-Shutdown Finish (End)

Master Controller:

The master controller is capable of issuing commands to the local controller and gives commands of Start-up and Shutdown. The automaton of Master controller is shown in figure 11. It only has 1 state as master controller doesn't depend on anything and can randomly issue Start-up and Shutdown commands to the OGS.

Events:

SUS-Start-Up Start (Begin)

SDS-Shutdown Start (Begin)

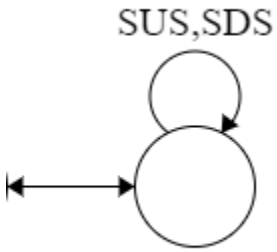


Figure 11: Master controller

Interactions:

In a system all the components are not independent to each other and sometimes interact with each other. These interactions have to be taken into account while creating plant model. From the given problem we find 3 interactions between the components namely F1-V1, F2-V2-V3, A-F1-PSU.First we deal with F1-V1

F1-V1:

The sensor F1 depends on the state of the Valve 1 directly. If we observe figure 2, we find that F1 is placed at same pipe as V1 hence for flow to exist in the pipe V1 must be open and for no flow to exist the V1 must be closed. Hence, we need to take the following interaction into account. The automaton of the following interaction is shown in Figure 12. The automaton is designed by taking V1 automaton and self-looping F1L and F1H events to C and O states respectively.

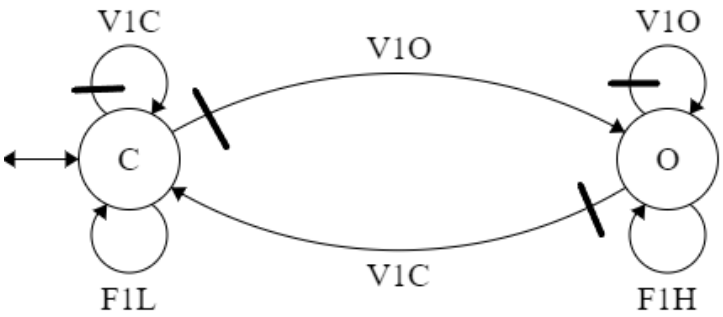


Figure 12: Interaction 1 F1V1

F2-V2-V3:

From the figure 2, we observe that Valve 2 and Valve 3 are connected to the same pipe. This pipe also contains a flow meter F2. Hence, for the flow to exist in this pipe, both valves must be open then only the F2 reading can go high and also vice versa for low. Hence, to find the interaction of F2-V2-V3, we must find the sync of both V2-V3 and self-loop F2L for all the states except when both the valves are open and self-loop F2H when both the valves are open. The automaton of the following interaction is shown in Figure 13.

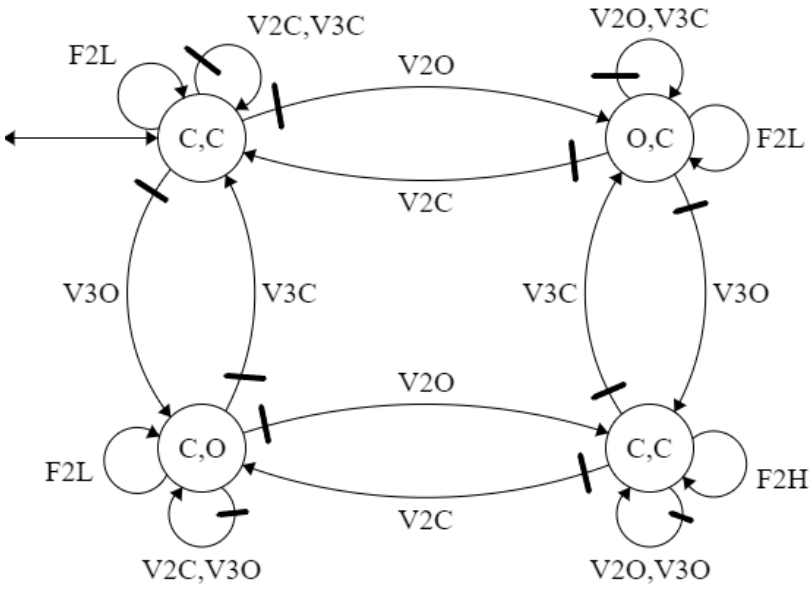


Figure 13: Interaction 2 F2V2V3

A-F1-PSU:

From both the figure and problem we know that the Ozone concentration is dependant on both F1 and the PSU. From the figure we observe F1 being connected to same pipe as Ozone Generator and PSU and for Ozone concentration to increase there must be flow of oxygen in the pipe. Hence, the reading of the analyser is dependant on F1 and PSU in such a way that AH only occurs when F1 is H and PSU is ON. To draw the interaction, we perform sync of PSU and F1 and self-loop AL on all states except when F1 is High and PSU is on and self-loop AH on the state when F1 is High and PSU is On. The automaton for the following interaction is shown in figure 14.

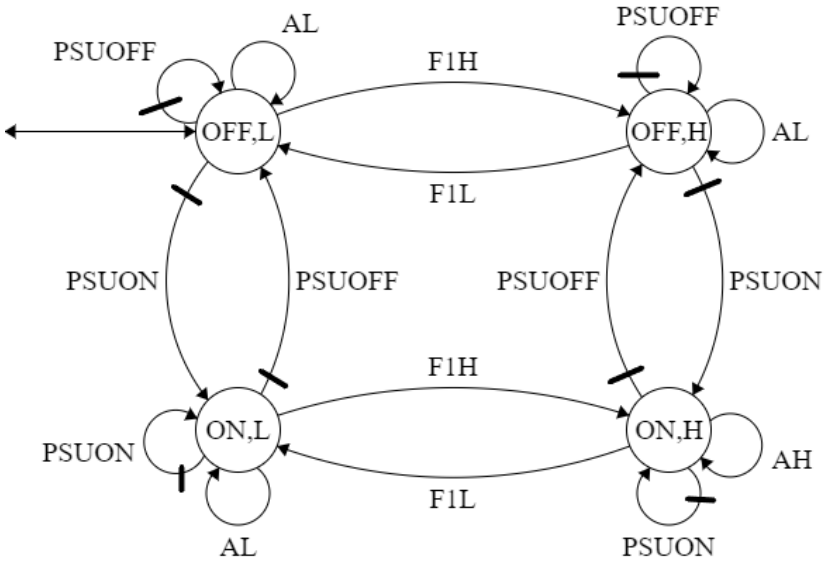


Figure 14: Interaction3 AF1PSU

Hence, to create the plant model we sync all the components and interactions to create an automaton which contains all the possible states and events of the whole OGS combined. As the following calculation is too complex to do by hand in this project, we use DECK [1] toolbox to code and design the plant model.

The list of the models on which sync is performed to obtain plant model is:

i) Valve 1
ii) Valve 2
iii) Valve 3
iv) Sensor F1
v) Sensor F2
vi) Power Supply Unit
vii) Analyser
viii) Local automaton
ix) Master controller
x) Interaction F1-V1
xi) Interaction F2-V2-V3
xii) Interaction A-F1-PSU

From the DECK [1] (MATLAB) results we find the plant model consists of 128 states and 1728 transitions in total with only first state being marked.

The plant automaton is shown in figure 15 but is drawn partially because of space and complexity constraints but is fully designed and implemented in MATLAB using DECK toolbox.

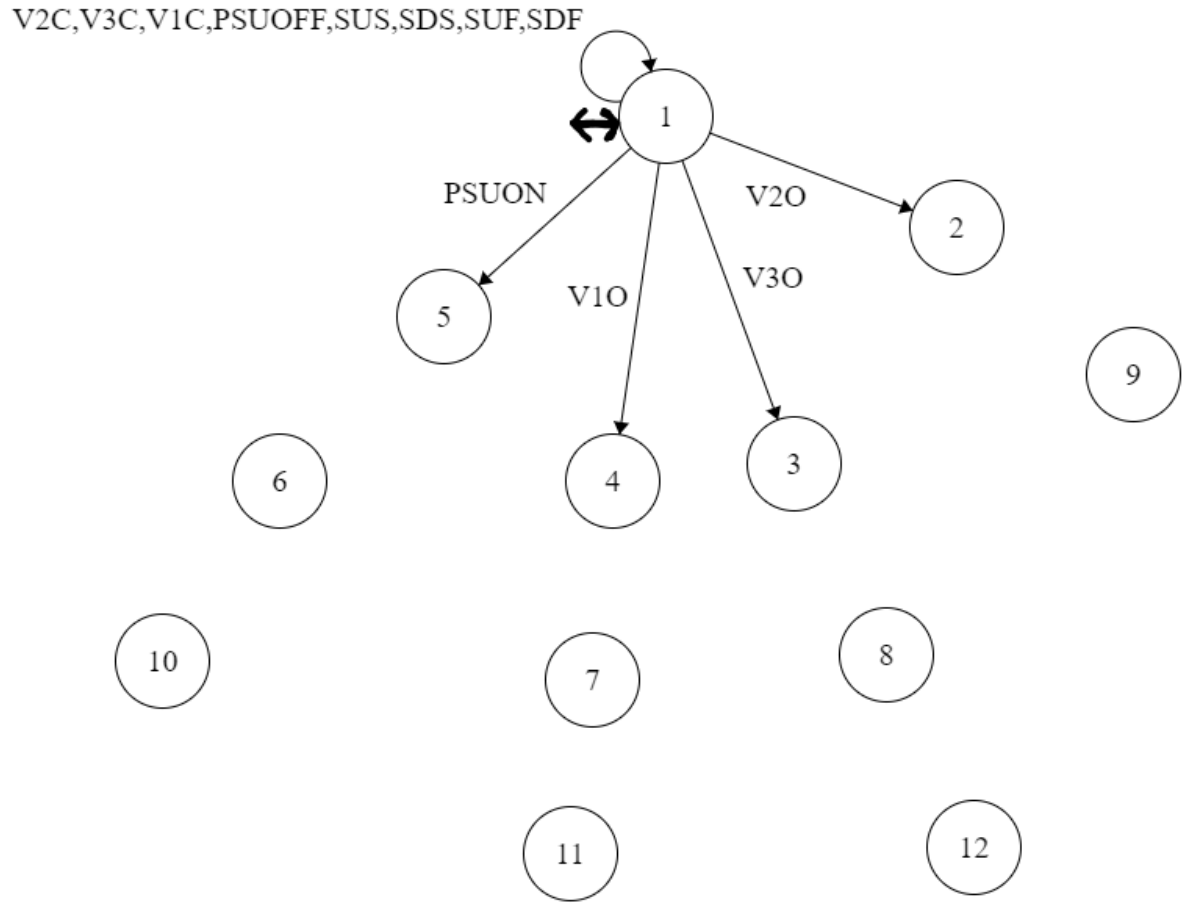


Figure 15: Partial plant automaton

Specifications:

SPEC1:

From the Problem definition the sequence of Start-up and Shutdown is already specified. Thus, for the automaton for SPEC1 we use step by step sequence of events and design an automaton. The sequence specified is:

1) Arrival of SUS (Start-up Command)
2) Start the flow of cooling water (F1H)
3) Start the flow of oxygen (F2H)
4) Start the power supply unit (PSUON)
5) Send Start-up-Finish message to Master-controller (SUF)
6) Arrival of SDS (Shutdown Command)
7) stop the power supply unit (PSUOFF)
8) stop the flow of oxygen (F2L)
9) stop the flow of cooling water (F1L)
10) Send Shutdown-complete message to Master-Controller (SDF).

During all the states, except 1 and 6 Start-up and Shutdown commands must be ignored. Hence, we put self-loop of SUS and SDS to 2,3,4,5,7,8,9,10. But also for state 1 SDS (Shutdown command) must be ignored as it already OFF and for 6 SUS (Start-up command) must be ignored as it is already ON. For specifications all the states should be marked. Hence by following the above sequence we get the following automaton shown in figure 15:

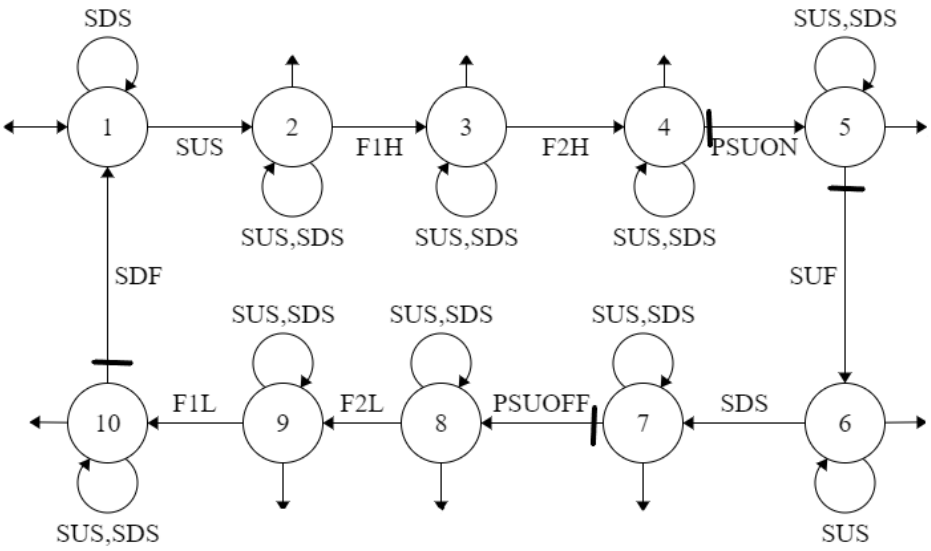


Figure 15: SPEC1

SPEC2:

From the problem statement, we find that during shutdown process the OGS must supply oxygen to the system until the Ozone's concentration becomes less, for this we must consider a unique automaton which consists of 4 states,

ON: OGS is ON

OFF: OGS is OFF

Isu: Intermediate state start-up (implies during start-up)

Isd: Intermediate state shutdown (implies during shutdown).

After interconnecting the sequences, the designed automaton is shown in the figure 16.

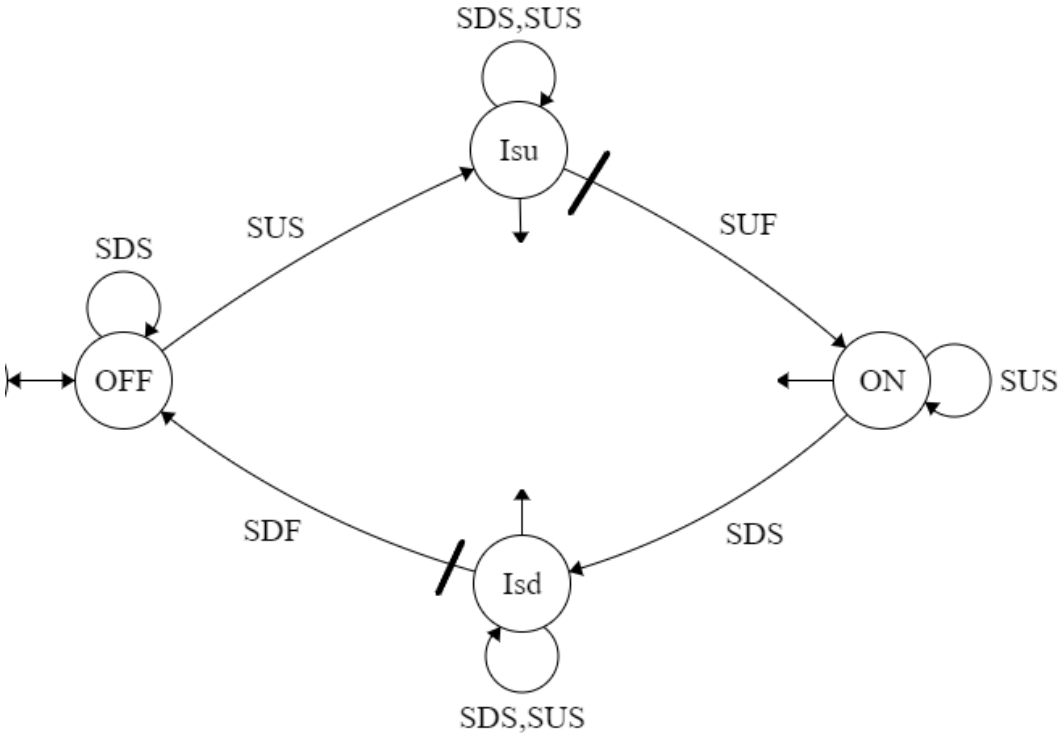


Figure 16: SPEC2 part1

As it is specified by the problem that during Shutdown and when Ozone concentration is high there must be supply of oxygen. Hence to cater for the interaction between the automaton from the figure 16 and Analyser, we find the sync of the both automata. After we find the sync, we do self-loop F2L and F2H to all the states of the automaton except the state when the automaton is during shutdown and Ozone concentration is high. During the state when the automaton is during shutdown and Ozone concentration is high, we only self-loop F2H to that particular state. As the following automaton is specification all the states must be marked. The final automaton is shown in figure 17, for simplicity events of sync of SPEC2 part 1 and Analyser are not drawn here

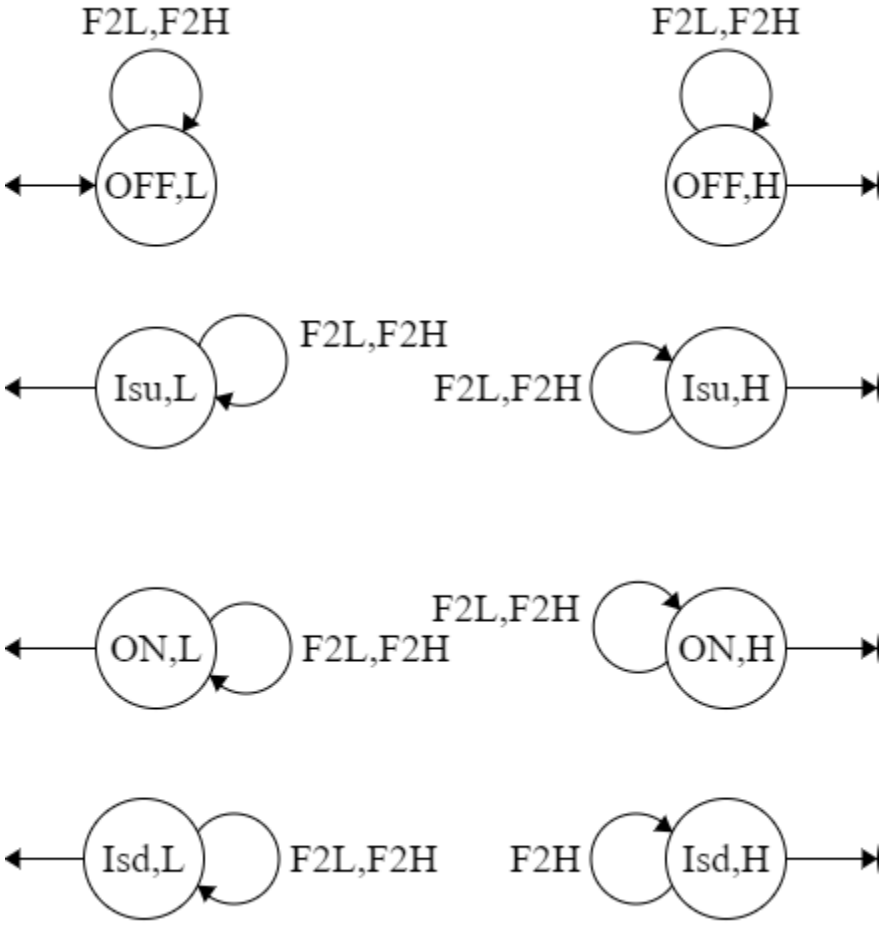


Figure 17: SPEC2

Supervisor Design:

As per Supervisory Control Theory, for a Specification automaton and a plant automaton there exists an automaton which can supervisor the plant automata to get the specified sequences or requirements present in the Specification Automaton here, for ease of name the Specification automata will be named as Spec.The working of supervisor is shown in figure 18 where the set of both the Plant model and Local Supervisor is called Model under supervision.

Local Supervisor for SPEC1:

From the specifications mentioned in the problem and designed SPEC1 as seen in Figure 15. We use the command mentioned in DECK [1] called which can create supervisor automaton when plant automaton, Specification automaton and set of Uncontrollable events are provided. The command used is called “supcon”

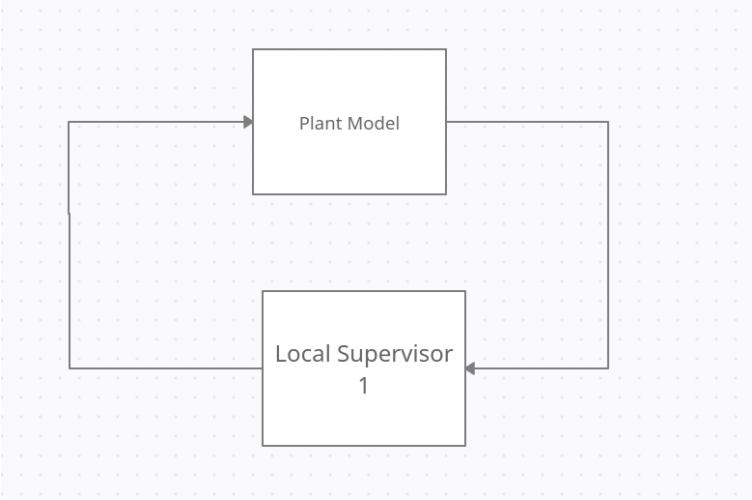


Figure 18

Hence, for the Design of Local Supervisor 1 the steps are:

1) Self-loop all the possible events of plant automata SPEC1 which are already not present in SPEC1 so that we can ignore such events. For this “selfloop” command is used.
2) Make a set of Uncontrollable events to be given as input to “supcon”.
3) Give the new SPEC1, the plant model and set of Uncontrollable events to supcon.
4) Record the findings

Result of Local Supervisor 1:

After Following the steps mentioned above, we obtain an automaton with 60 states, 466 transitions and 3 marked states. The marked states are [1, 4, 43]. The partial automaton of Local supervisor 1 is shown in figure 19. Due to space and complexity constraints the following automaton is partially drawn and its working is shown.

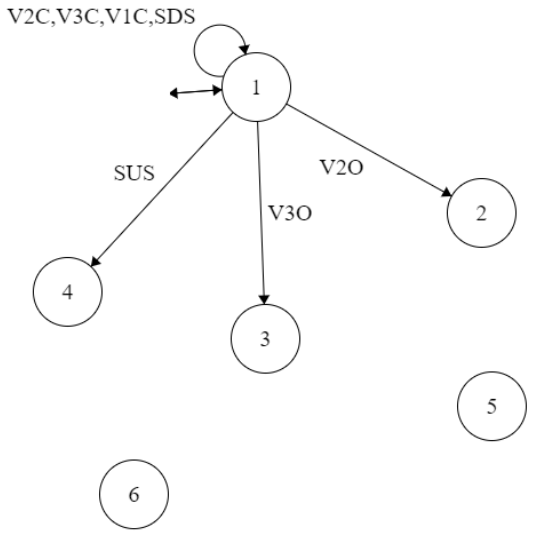


Figure 19

If we observe the automaton and compare it to plant automaton, there are few disablements already from state 1 to make sure the Supervisor follows the required Spec. This set of disablements can help the supervisor to produce the results required. To understand more clearly, we are going to use project and disable few extra events in plant under supervision to get an automaton which properly explains the working. After using the command “project” and disabling all events except Analyser Highs and Lows, F2 Highs and Lows and SUS and SDS commands, we obtain an automaton which explains possible events as specified by SPEC1. The automaton was found to contain 14 states and 44 transitions with 1,2,7 and 10 as marked states. As the order of magnitude of such automaton is high the automaton is drawn partially and shown in figure 20.

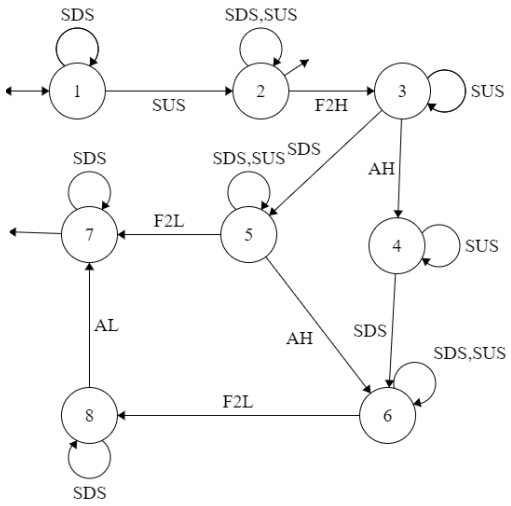


Figure 20: Projection of local Supervisor 1

The following automaton shows the sequence of Events as specified in SPEC1 hence always whenever SUS is triggered, F2H or AH or both are proceeded by it and if SDS is triggered, F2L or AL or both are triggered. This explains us the working of Automaton under Supervision 1.

Local Supervisor for both SPEC1 and SPEC2:

As provided by problem, the project also requires the Supervisor to consider both SPEC1 and SPEC2 as well, hence the steps to obtaining local supervisor with both SPEC1 and SPEC2 are:

1) Self-loop SPEC1 with the events not already present in SPEC1
2) Repeat the same for SPEC2
3) Find the product of SPEC1 and SPEC2
4) Use the product, the plant model and the list of Uncontrollable events to create local supervisor 2 using the command supcon on DECK [1]. The obtained automaton will be named Local supervisor 2.
5) Assess the findings.

Results:

Hence after following the steps mentioned above, we obtain an automaton with number of states 34 and 244 transitions and three marked states. The marked states being 1,4,34. Due to space and complexity constraints a partial automaton of Local supervisor 2 is shown in figure 21. From the figure 21 we observe the Supervisor 1 is same as Supervisor 2 for the state 1 but there has

been some reduction in states and events as another condition has been placed on the supervisor to cater for.

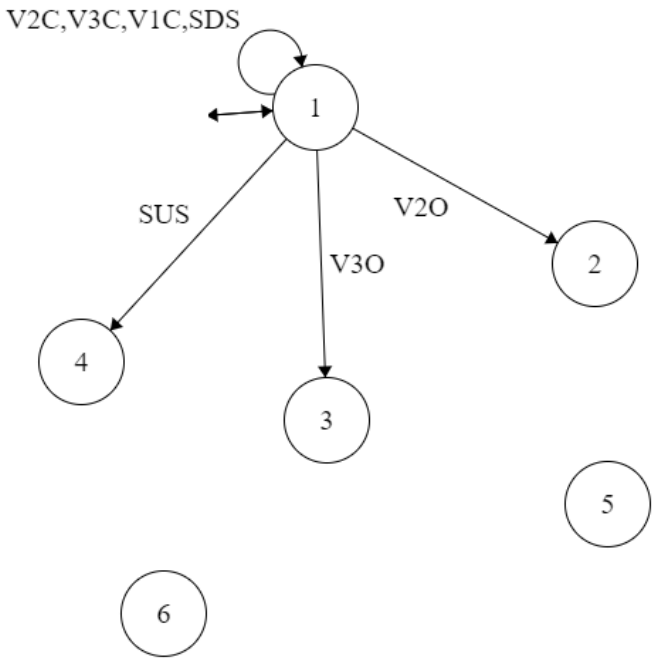


Figure 21: Local Supervisor 2

To assess our findings, we again perform projection on the plant under supervision automaton. This gives us a better view of working of the Local Supervisor 2. After project command we obtain an automaton with 9 states ,25 transitions and 4 states. The following automaton is shown in the figure 22.

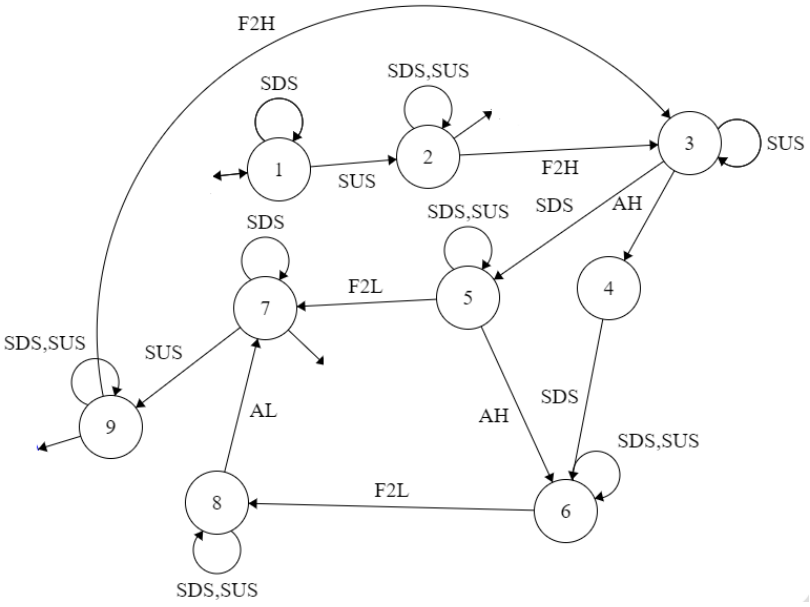


Figure 22: Projection of Local supervisor 2

Hence, from the projection of local supervisor we can see that the sequences are not changed but there have been some disablements to cater for Specification 2. Thus, we can verify that the local Supervisor is the automaton required for the local controller to execute the specifications of OGS.

Conclusion:

Hence, we conclude that the local supervisor 2 is the supervisory automaton required to supervise the Ozone Generation System according to the specifications provided. The following fact can be validated by checking the projection of plant model under supervision of Local supervisor 2, as the sequences were following the specifications required without error. To summarise, the project, we created a supervisory automaton which can supervise (according to specifications) an Ozone Generation system shown in the figure 1 by using a MATLAB toolbox named DECK. During the project we encountered many problems pertaining to specification 2 which was a bit complex to understand and implement in DECK. The flexibility and the robustness of DECK Kit made the project easier and faster. The project can improve its accuracy by catering for failure in it, as during the project we neglected failure and obtained the result but in real life scenario failure are bound to happen within components and with DES models we should be able to cater to them as well. Hence, we conclude our project.

Appendix:

EVENT TABLE:

The following table 1 shows the event name and the code given to the said event in DECK [1].

EVENT NAME	CODE allotted in DECK
Valve 2 Open (V2O)	10
Valve 2 Close (V2C)	11
Valve 3 Open (V3O)	12
Valve 3 Close (V3C)	13
Valve 1 Open (V1O)	14
Valve 1 Close (V1C)	15
Analyser Reads High (AH)	20
Analyser Reads Low (AL)	21
F1 Reads High (F1H)	18
F1 Reads Low (F1L)	19
F2 Reads High (F2H)	16
F2 Reads Low (F2L)	17
Turn PSU On (PSUON)	22
Turn PSU Off (PSUOFF)	23
Shut down Finish (SDF)	31
Start-up Finish (SUF)	33
Shut down Start (SDS)	30
Start-up Start (SUS)	32

Table 1

DECK MATLAB CODE:

STEP 1:

```
clc
clear all

%Step 1

% V2 Valve
NV2=2;
V2O=10;% V2 Valve Open
V2C=11;% V2 Valve Close
TLV2=[1 V2C 1; 1 V2O 2; 2 V2O 2; 2 V2C 1];
XmV2=[1];
V2= automaton(NV2,TLV2,XmV2);

% V3 Valve
NV3=2;
V3O=12;% V3 Valve Open
V3C=13;% V3 Valve Close
TLV3=[1 V3C 1; 1 V3O 2; 2 V3O 2; 2 V3C 1];
XmV3=[1];
V3= automaton(NV3,TLV3,XmV3);

% V1 Valve
NV1=2;
V1O=14;% V1 Valve Open
V1C=15;% V1 Valve Close
TLV1=[1 V1C 1; 1 V1O 2; 2 V1O 2; 2 V1C 1];
XmV1=[1];
V1= automaton(NV1,TLV1,XmV1);


%A1- Analyser:

NA1=2;
AH=20; %A1 Reads High
AL=21; %A1 Reads Low
TLA1=[1 AH 2;2 AL 1];
XmA1=1;
A=automaton(NA1,TLA1,XmA1);

% F2 Flow Sensor:

NF2=2;
F2H=16; %F2 Reads High
F2L=17; %F2 Reads Low
TLF2=[1 F2H 2;2 F2L 1];
XmF2=1;
F2=automaton(NF2,TLF2,XmF2);

% F1 Flow Sensor:

NF1=2;
```

```
F1H=18;%F1 Reads High
F1L=19;%F1 Reads Low
TLF1=[1 F1H 2;2 F1L 1];
XmF1=1;
F1=automaton(NF1,TLF1,XmF1);

% PSU

NPSU=2;
PSUON=22; % Turn PSU ON
PSUOFF=23; % Turn PSU OFF
TLPSU=[1 PSUOFF 1; 1 PSUON 2; 2 PSUON 2;2 PSUOFF 1];
XmPSU=1;
PSU=automaton(NPSU,TLPSU,XmPSU);

%Local automaton


SDF=31; % Shut down Finish
SUF=33; % Start up Finish

Nlocal=1;
localtl=[1 SUF 1;1 SDF 1];
Xmlocal=1;

local=automaton(Nlocal,localtl,Xmlocal);


%Master Controller
SDS=30; % Shut down Start
SUS=32; % Start up Start

NMaster=1;
mastertl=[1 SUS 1;1 SDS 1];
Xmmaster=1;

master=automaton(NMaster,mastertl,Xmmaster);

%-------------------------------------------------------------------------
---
%interactions

NF1V1=2;
TLF1V1=[ TLV1 ; 1 F1L 1; 2 F1H 2];
XmF1V1=[1];
F1V1=automaton(NF1V1,TLF1V1,XmF1V1);

%--------------------------------

[F2V2V3,states]=sync(V2,V3);
for i=1:size(states,1)
   if (states(i,1)==2  && states(i,2)==2)
     F2V2V3.TL=[F2V2V3.TL;i F2H  i];
   else
     F2V2V3.TL=[F2V2V3.TL; i F2L i];
   end
end
```

```
%---------------------------------

[A1PSUF1,states]=sync(PSU,F1);
for i=1:size(states,1)
   if (states(i,1)==2  && states(i,2)==2)
     A1PSUF1.TL=[A1PSUF1.TL;i AH  i];
   else
     A1PSUF1.TL=[A1PSUF1.TL; i AL i];
   end
end

%plantmodel

plantautomaton=sync(V2,V1,V3,A,F2,F1,local,master,PSU,F1V1,F2V2V3,A1PSUF1);

%SPEC1:


SPEC1N=10;
SPEC1TL=[1 SDS 1;1 SUS 2; 2 F1H 3;3 F2H 4;4 PSUON 5;5 SUF 6;6 SUS 6;6 SDS
7;7 PSUOFF 8; 8 F2L 9; 9 F1L 10; 10 SDF 1];
for i=[2,3,4,5,7,8,9,10]
    SPEC1TL=[SPEC1TL; i SUS i; i SDS i ];
end

XmSPEC1=[1,2,3,4,5,6,7,8,9,10];

SPEC1=automaton(SPEC1N,SPEC1TL,XmSPEC1);
SPEC1=selfloop(SPEC1,[AH,AL,V1O,V1C,V2O,V2C,V3O,V3C]);

UC=[AH,AL,F2H,F2L,F1L,F1H,SUS,SDS];
Supervisor1=supcon(SPEC1,plantautomaton,UC)

EUC=[V1C,V1O,V2C,V2O,V3O,V3C,F1H,F1L,SUF,SDF,PSUON,PSUOFF];

project1=project(product(Supervisor1,plantautomaton),EUC)
```

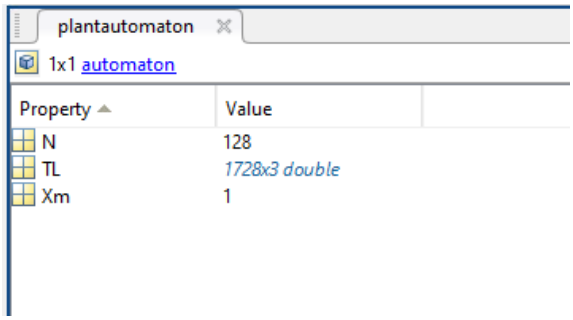
STEP 2:

```
%Step 2

%SPEC2:

NSTSD=4;
STSDTL=[1 SDS 1;1 SUS 2;2 SUS 2;2 SDS 2;2 SUF 3;3 SUS 3;3 SDS 4;4 SDS 4;4
SUS 4;4 SDF 1];
XmSTSD=[1,2,3,4];
STSD=automaton(NSTSD,STSDTL,XmSTSD);

[SPEC2,states]=sync(STSD,A);

for i=1:size(states,1)
    if (states(i,1)==4 && states(i,2) == 2)
        SPEC2.TL=[SPEC2.TL ; i F2H i];
    else
        SPEC2.TL=[SPEC2.TL; i F2L i; i F2H i];
    end
end

SPEC2=selfloop(SPEC2,[PSUON,PSUOFF,V1O,V1C,V2O,V2C,V3O,V3C,F1H,F1L]);
SPEC2.Xm=[1,2,3,4,5,6,7,8];

Supervisor2=supcon(product(SPEC1,SPEC2),plantautomaton,UC)
project2=project(product(Supervisor2,plantautomaton),EUC)
```

Results from DECK:

Plant automaton:

Automaton details:



Partial Automaton transitions for verification:

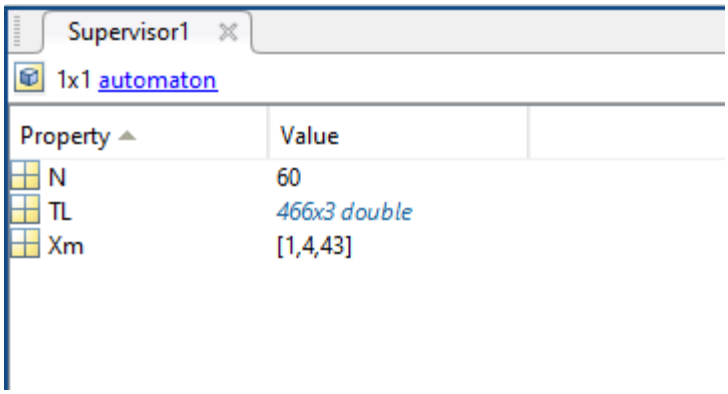
plantautomaton | plantautomaton.TL

plantautomaton.TL

	1	2	3	4	5
1	1	10	2		
2	1	11	1		
3	1	12	3		
4	1	13	1		
5	1	14	4		
6	1	15	1		
7	1	22	5		
8	1	23	1		
9	1	30	1		
10	1	31	1		
11	1	32	1		
12	1	33	1		
13	2	10	2		
14	2	11	1		
15	2	12	6		
16	2	13	2		
17	2	14	7		
18	2	15	2		
19	2	22	8		
20	2	23	2		
21	2	30	2		
22	2	31	2		
23	2	32	2		
24	2	33	2		
25	3	10	6		
26	3	11	3		
27	3	12	3		
28	3	13	1		
29	3	14	9		
30	3	15	3		
31	3	22	10		
32	3	23	3		
33	3	30	3		
34	3	31	3		
35	3	32	3		

Local Supervisor 1:

Automaton details:



Partial Transition details

Editor - Step1.m

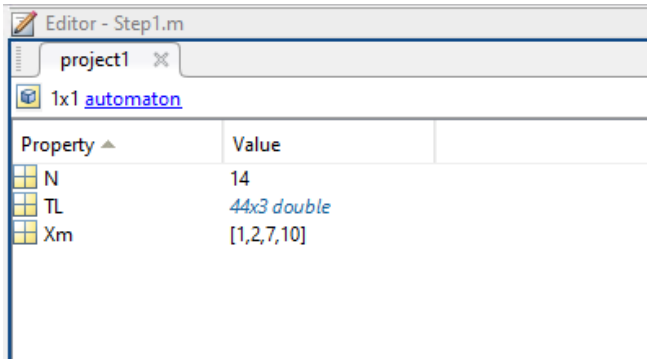
Supervisor1 | Supervisor1.TL

Supervisor1.TL

	1	2	3	4	5	6
1	1	10	2			
2	1	11	1			
3	1	12	3			
4	1	13	1			
5	1	15	1			
6	1	30	1			
7	1	32	4			
8	2	10	2			
9	2	11	1			
10	2	13	2			
11	2	15	2			
12	2	30	2			
13	2	32	5			
14	3	11	3			
15	3	12	3			
16	3	13	1			
17	3	15	3			
18	3	30	3			
19	3	32	6			
20	4	10	5			
21	4	11	4			
22	4	12	6			
23	4	13	4			
24	4	14	7			
25	4	15	4			
26	4	30	4			
27	4	32	4			
28	5	10	5			
29	5	11	4			
30	5	13	5			
31	5	14	8			
32	5	15	5			
33	5	30	5			
34	5	32	5			
35	6	11	6			

Project 1:

Automaton details:



Partial transition details:

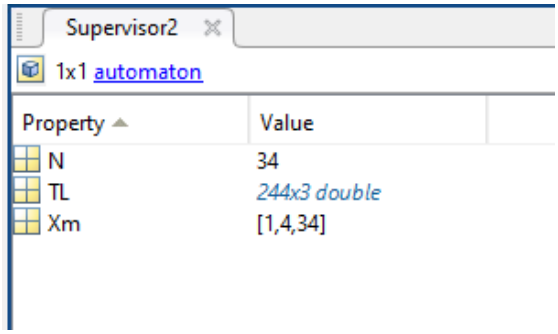
project1 | project1.TL

project1.TL

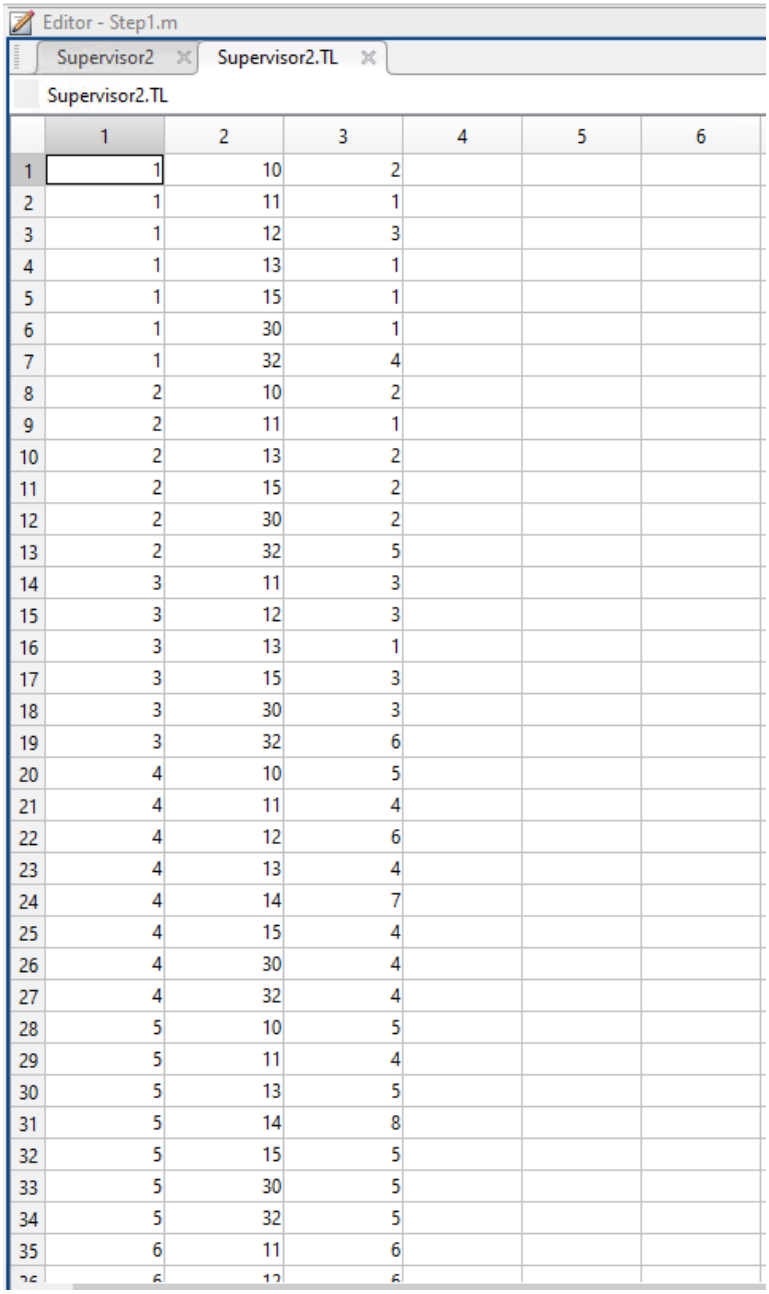
	1	2	3	4	5
1	1	30	1		
2	1	32	2		
3	2	16	3		
4	2	30	2		
5	2	32	2		
6	3	20	4		
7	3	30	5		
8	3	32	3		
9	4	30	6		
10	4	32	4		
11	5	17	7		
12	5	20	6		
13	5	30	5		
14	5	32	5		
15	6	17	8		
16	6	21	9		
17	6	30	6		
18	6	32	6		
19	7	30	7		
20	7	32	10		
21	8	21	7		
22	8	30	8		
23	8	32	11		
24	9	17	7		
25	9	30	9		
26	9	32	9		
27	10	16	3		
28	10	30	10		
29	10	32	10		
30	11	16	12		
31	11	21	10		
32	11	30	11		
33	11	32	11		
34	12	21	3		
35	12	30	13		

Local Supervisor 2:

Automaton details:



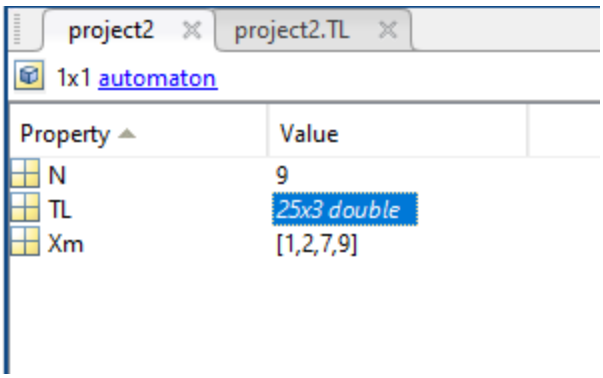
Partial transition details:



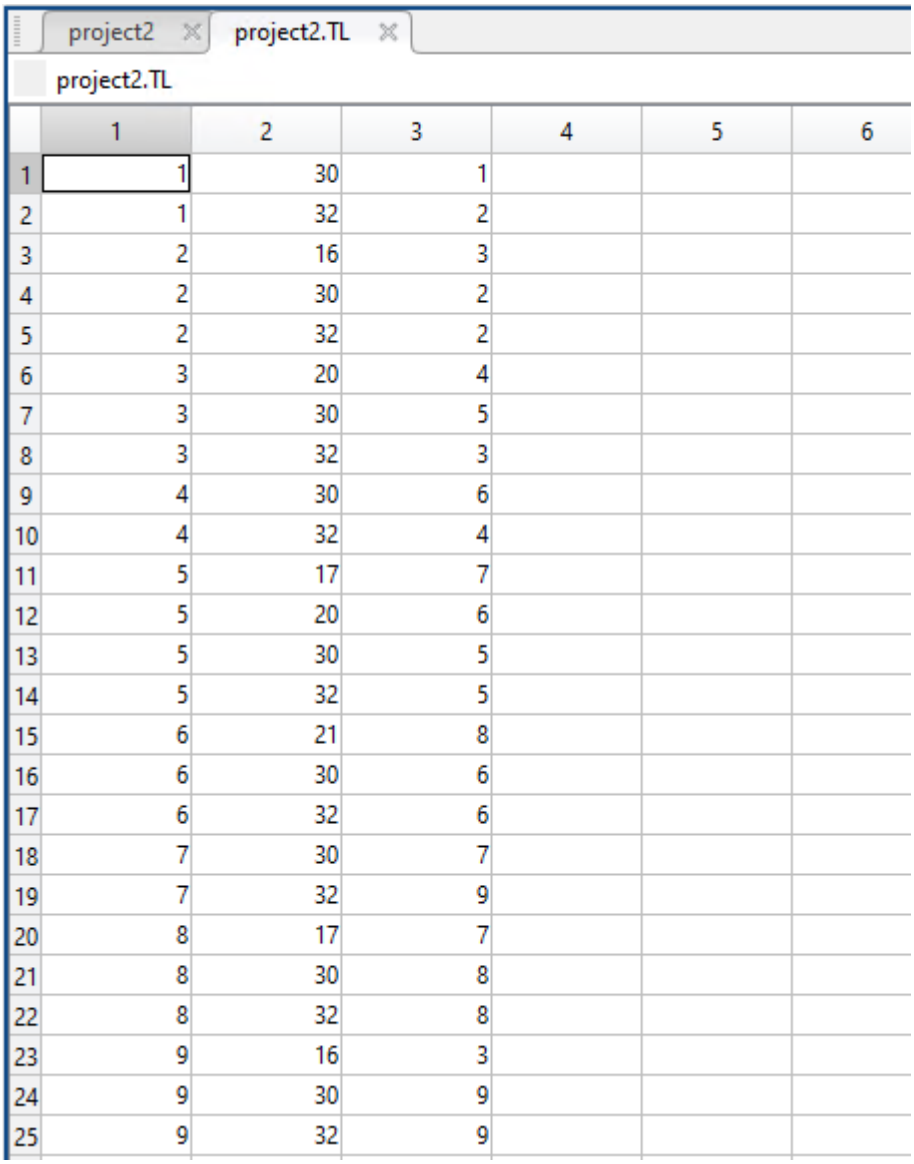
	1	2	3	4	5	6
1	1	10	2			
2	1	11	1			
3	1	12	3			
4	1	13	1			
5	1	15	1			
6	1	30	1			
7	1	32	4			
8	2	10	2			
9	2	11	1			
10	2	13	2			
11	2	15	2			
12	2	30	2			
13	2	32	5			
14	3	11	3			
15	3	12	3			
16	3	13	1			
17	3	15	3			
18	3	30	3			
19	3	32	6			
20	4	10	5			
21	4	11	4			
22	4	12	6			
23	4	13	4			
24	4	14	7			
25	4	15	4			
26	4	30	4			
27	4	32	4			
28	5	10	5			
29	5	11	4			
30	5	13	5			
31	5	14	8			
32	5	15	5			
33	5	30	5			
34	5	32	5			
35	6	11	6			

Project 2:

Automaton details:



Transition details:



	1	2	3	4	5	6
1	1	30	1			
2	1	32	2			
3	2	16	3			
4	2	30	2			
5	2	32	2			
6	3	20	4			
7	3	30	5			
8	3	32	3			
9	4	30	6			
10	4	32	4			
11	5	17	7			
12	5	20	6			
13	5	30	5			
14	5	32	5			
15	6	21	8			
16	6	30	6			
17	6	32	6			
18	7	30	7			
19	7	32	9			
20	8	17	7			
21	8	30	8			
22	8	32	8			
23	9	16	3			
24	9	30	9			
25	9	32	9			

References

[1] Discrete Event Control Kit (DECK). Department of Electrical and Computer Engineering, Concordia University, http://www.ece.concordia.ca/~shz/deck.

[2] A. Mohammadi-Idghamishi and S. Hashtrudi-Zad, “Hierarchical fault diagnosis: Application to an ozone plant,” IEEE Transactions on Systems, Man, and Cybernetics: Part C, vol. 37, no. 5, pp. 1040-1047, Sep. 2007.