CS684 Spring 2024-25

CS684: Embedded System Course

Assignment 3: Lustre/Heptagon

PROBLEM STATEMENTS

Q1.

Consider the following node display discussed in class.

```
node display(updown:bool) returns (o,q:int)
var
   last z: int = 20;
   y:int;
let
  q = z;
  switch updown
  | true  do  y = 100 -> pre(y)+1;
                      z = (last z) + 1;
                      o=y + z
  | false do  y = 10  -> pre(y)-1;
                      z = (last z) - 1;
                      o=y + z
  end
tel
```

What will be the values of output flows o and q for the input flow

- updown = 0 0 1 1 1 0 1 0 0 1 1 1

Answer format: Create q1 text file. List the output values of "o" in first line and q in second line in comma-separated format.

Q2.

Consider the following node which is a variant of the node discussed in the class. The change is that the transitions are of then type here instead of the continue type as shown in class. Hence the behaviour of the automaton differs.

Complete the table below giving the output of this node for the first 11 cycles in a table. Also give the start state (ST) and then next state (NS) for each of these cycles.

```
node myautomaton() returns (y:int; stup:bool; v:int)
var last x:int = 2;
let
   y = x;
   automaton
      state Up
            var w:int;
            do x = (last x) + 1; stup = true;
              w = 0 -> pre(w)+1; v=w;
      until  x >= 5 then Down
      state Down
         var w:int;
         do x = (last x) - 1;  stup = false;
         w = 50 -> pre(w)-2; v=w;
      until x <= 3 then Up
   end
tel
```

Flow Cycle	ST State	Y v st up	NS State

Answer format: Create q2 text file. List the values for 11 cycles in comma-separated format. Make sure to write each expression's value for 11 cycles on a separate line.

Q3.

Consider the following node. Complete the table below giving the output of this node for the first 10 cycles in the table. Also give the start state (ST), active state (AS) and then next state (NS) for each of these cycles.

```
node myautomaton(c: bool) returns (o: int; stup:bool)
```

```
let
  automaton
    state Up
      do o = 60 -> pre(o)+2; stup = true;
    unless c continue Down
    state Down
      do o = 150 -> pre(o)-2; stup = false;
    until c then Up
  end
tel
```

Fl
o
w
C
yc
le

S
T

A
S

c

o

st
up

N
S

Answer format: Create q3 text file. Without using a space, list the values for 10 cycles in comma-separated format. Make sure to write each expression's value for 10 cycles on a separate line.

Q4.

Consider the node myautomaton given below. Try to understand its structure. How many distinct modes does it have? Explain how you came up with this number.

```
node myautomaton(i1: bool; i2: bool; i3: bool; i4: bool)
 returns (o1: bool; o2: bool; o3: bool; o4: bool)
let
 automaton
  state State1
   do o1 = false; o2 = false; o3 = false; o4 = false
   until i1 then TMP12
   unless i2 then TMP11
  state TMP11
   do o1 = false; o2 = true; o3 = false; o4 = false;
   until true then State3
  state TMP12
   do o1 = true; o2 = true; o3 = false; o4 = false;
   until true then State2
```

```
state State3
  var last end1: bool = false; last end2: bool = false;
  do o1 = false; o2 = false; o3 = false; o4 = false;
     automaton
       state State1_1
         do until i3 then State2_1
       state State2_1
         do end1 = true;
     end;
     automaton
       state State1_2
         do unless i4 then State2_2
       state State2_2
         do end2 = true
     end
   until end1 & end2 then State1
   unless i1 then State4
      |   i2 then State2

state State2
   var l1: bool;
   do o1 = false; o2 = false; o3 = false; o4 = l1; l1 = true;
   until i4 then State4

state State4
   var l2: bool;
```

```
      do o1 = false; o2 = false; o3 = l2; o4 = false; l2 = true;
      until i4 then State1
    end
tel
```

Answer format: Create q4 text file. Mention number of modes in first line and explanation in second line.

Q5.

Consider the following controller for farm road crossing. Try to understand its working. Try to simulate it using Heptagon simulator.

A farm road (or side road) crosses a main road. Traffic light controller must turn on or off the lights maingreen, mainred, sidegreen, sidered. An input "carwait" is true if a car is waiting on the farm road. Input "second" is the timer input which becomes true for one clock cycle every one second. Thus the count of "second" gives how much time has elapsed.

```
node traffic(carwait,second:bool)
    returns (maingreen, mainred,sidegreen,sidered:bool)
var timegreen:int;
let
  automaton
    state Maingreen
      do  timegreen = 180 -> if (((pre(timegreen)) > 0) and second) then
                              pre(timegreen)-1  else pre(timegreen);

        maingreen = true; mainred = false;
        sidegreen = false; sidered = true;
      until ((timegreen <= 0) and carwait) then Sidegreen
    state Sidegreen
```

```
        do
            timegreen = 60 ->
                if (((pre(timegreen)) > 0) and second) then pre(timegreen)-1
                        else pre(timegreen);
            maingreen = false; mainred = true;
            sidegreen = true; sidered = false;
        until ((timegreen <=0)  and not carwait) then Maingreen
    end
tel
```

1. Modify the above controller by adding outputs “mainyellow and sideyellow. The aim is that traffic light must remain yellow for 10 seconds before turning red.
2. What are some of the requirements over the traffic node? For example, one simple requirement is that at most one of maingreen and sidegreen can be true in any clock cycle. List as many requirements (written in English) as you think are appropriate for the modified controller you have designed in part (1).

Answer format: For question 5.1, submit `q51.ept` file. And for question 5.2, create `q52` text file and mention different requirements in separate lines.

Q6.

Model a Gas Burner Controller as a Heptagon node `(Code required)`

```
node controller(flame: bool; sec: bool) returns (gas: bool; spark: bool)
```

to meet the following requirements. Preferably Use the automaton construct.

“Controller keeps gas on/off using the output gas and strikes a flame using the output spark. It can detect whether flame is on/off using input flame. Flow sec is a second beacon. It is true at each clock cycle where one second has elapsed since the previous such value. Flame will not occur unless the gas has accumulated for at least 15 seconds. Not every spark results in flame. Flame also occasionally goes off due to wind. Making a spark after Gas has leaked for more than 60 seconds causes an explosion. Hence, after a leakage longer than 60 seconds, the Gas must be turned off and allowed to dissipate for 120

seconds to reach a safe state. The controller should try to keep the flame on as much as possible."

Please explain your design. Simulate your code using the Heptagon simulator for various sample inputs and submit the output produced.

Answer format: Submit `q6.ept`. Explain your design in comments in code file.

Submission Instructions

- Create a folder named **`<RollNo>_assignment3`**.
- Copy and paste all the files which has to be submitted inside the newly created folder according to the structure as shown below:

<RollNo>_assignment3

```
|
└── q1 (text file)
|
└── q2 (text file)
|
└── q3 (text file)
|
└── q4 (text file)
|
└── q5
|   |   q51.ept
|   |   q52 (text file)
|
└── q6.ept
```

Note: Folder name and file name should be same as mentioned in the above structure.

- Compress the folder in a `.tar.gz` file and submit it on moodle.