ANSWER SHEET
CS 522, Embedded Systems
Endsem Exam, Monsooon 2023-24
Department of Computer Science and Engineering
IIT Guwahati

Name: Chandrabhushan Reddy Roll No.: 200101027

Date: 25/11/23

Student's Signature: ____________

Invigilator's Signature: ____________

Examiner's Signature: ____________

Do not write in this box.

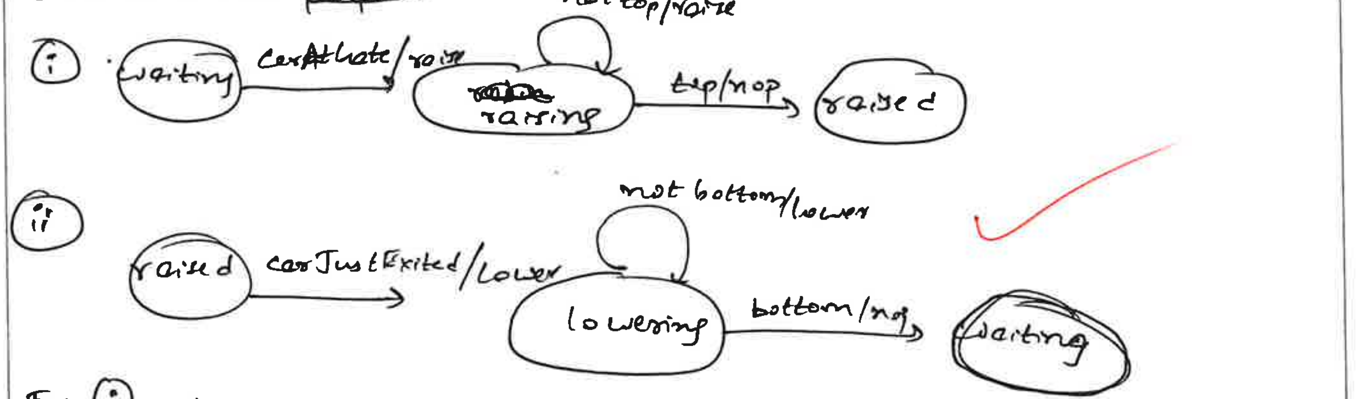
Question:	1	2	3	4	5	6	Total
Marks:	8	6	10	10	10	6	50
Score:	8	5	5	7	10	3	38

Answers

Q1(a) 4+4=8

Two desirable ~~exception~~ behaviours :-

4



In (i) when car arrives at gate, the state changes from waiting to raising and the gate keeps on raising until it reaches the top.

Similarly in (ii) when car just exits the gate, the state changes from raised to lowering and the gate keeps on lowering until it reaches the bottom.

Q1(b)

Two undesirable behaviours:-

(i) When the gate is in "raised" state and a car wants to "Enter" the parking lot then there is no corresponding transition for it. There is only transition for carJustExited there is no transition for carJustEntered. Because of this when the car enters the parking lot no transition happens and the parking gate will remain in "raised" state until another car exits.

4

(ii) When the gate is in "Lowering" state and if a car is at a gate then the car should wait until the gate has been completely lowered and then it starts raising again. The given controller will not immediately stop the lowering and start the raising when a car comes in lowering state.

Q2

Q2(a)

Let P_1, P_2 and P_3 be atomic propositions. If cruise is accelerating $P_1 \to 1$ $P_2 \to 0$ and $P_3 \to 0$. If cruise is cruising $P_1 \to 0$, $P_2 \to 1$, $P_3 \to 0$. If cruise is decelerating, $P_1 \to 0$, $P_2 \to 0$ and $P_3 \to 1$.

2

A temporal property is as follows:-

$$\square(P_1 \vee P_2 \vee P_3)$$

This property states that globally, at any point of time, cruise should be either accelerating, cruising or decelerating. It cannot be in any other state.

Q2(b)

Let Q be propositional variables. Let "Q" represents transition from accelerating to decelerating. Let "V" represent the current speed and "V_{thr}" represent the maximum upper bound of velocity that the cruise can have. Then,

2

$$\square\big((P_1 \wedge (V > V_{thr})) \longrightarrow Q\big)$$

This says that whenever $V > V_{thr}$ and P_1 is true i.e. the mode of cruise is accelerating then the transition from accelerating state to decelerating state should happen (i.e. $Q \to$ true)

Note:- P_1 represents that the state of cruise is accelerating

My answer 2a

Q2(c)

Let P_1, P_2 and P_3 be the propositional variables as declared in A2a then,

$$\square\left[(P_1 \wedge \sim P_2 \wedge \sim P_3) \vee (\sim P_1 \wedge P_2 \wedge \sim P_3) \vee (\sim P_1 \wedge \sim P_2 \wedge P_3)\right]$$

The above assertion states that globally, at any state, only one of P_1, P_2, P_3 can be true i.e., At any point of time the cruise can only be in exactly one of accelerating, cruising and decelerating states. It cannot simultaneously be in one or more of the above mentioned states.

Q3

Q3(a)

Given implementation does not check and handle integer overflow error.

1

Q3(b)

~~After taking user input the following code would be added.~~

```
~~# Variables
desiredTemperature_check : long integer
# User Input
desiredTemperature = getUserInput()

#Integer Overflow Checking and Handling
if(desiredTemperature > INT_MAX):
    displayErrorMessage("Integer overflow~~
```

Q3(c)

The dead logic present in the code is at the ~~Temperature control logic~~ condition of the while loop. Instead of while(true) ~~it~~ should have been while (desiredTemperature != currentTemperature). Because of this dead logic, when the desiredTemperature becomes equal to the currentTemperature ~~[illegible]~~ the process gets stuck inside the while loop forever. The process will not terminate.

0

Q4

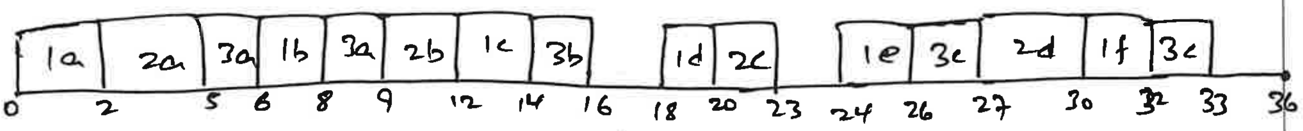
Q4(a)

If RMS is used, T_1 would have highest priority and T_3 would have lowest priority. This is because T_1 has the ~~[illegible]~~ least value of period and T_3 has the highest value of period.

1

Q4(b)

The RMS schedule will look as follows :-



The above schedule repeats for every 36 time units.

Response times for tasks 1a, 1b, 1c, 1d, 1e and 1f are 2,2,2,2,2 and 2 respectively

Response times for tasks 2a, 2b, 2c and 2d are 3,3,3, and 3 respectively

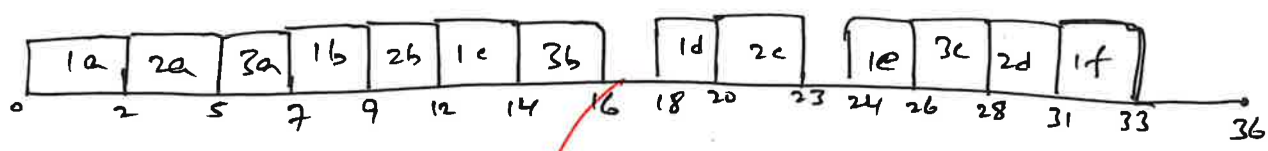
Response times for tasks 3a, 3b and 3c are 4, 2 and 7 respectively.

∴ Worst case response time for tasks of type T_1, type T_2 and type T_3 are 2, 3 and 7 respectively.

1

Q4(c)

EDF scheduling is possible. The schedule is shown below :-



The above schedule repeats for every 36 time units.

5

Q5

Q5(a) $2+4+4=\underline{10}$

2

One infinite sequence accepted by this automaton is 1111111-----
i.e. an infinite sequence of 1's. $\Rightarrow$ For this sequence the ~~[illegible]~~ accepting state is visited infinitely often.

One infinite sequence <u>not</u> accepted by this automaton is 00000---
i.e. an infinite sequence of 0's $\Rightarrow$ For this sequence the accepting state is not even visited once.

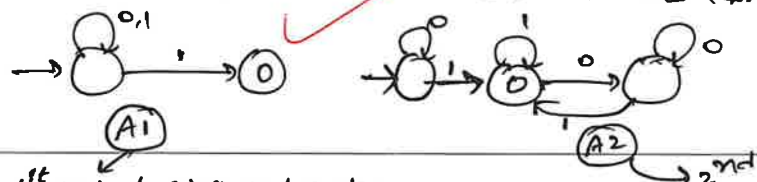
Q5(b)

4

The set accepted by this automaton <u>does not</u> change even if we remove the self-loop of "1" in the initial state. This is because

(i) If the string does not contain even one "1" then essentially both the automaton are trivially same.

(ii) If the string contains atleast one "1" then we can make the transition from S_0 to S_1 and then ~~[illegible]~~ essentially everything else can be implemented by the 2nd part of the automaton i.e. the language of automaton (A1) is same as language of automaton (A2)



$\boxed{L(A_1) \equiv L(A_2)}$

1st part of given automaton

2nd part of given automaton

Q5(c)

4

From the given Buchi automaton we can interpret that it will accept any infinite sequence of 0's and 1's ~~[illegible]~~ provided that in that string, whenever a "0" occurs, "1" should eventually occur ~~after~~ sometime after the "0".

So an example LTL formula which describes all the set of sequences accepted by this automaton is $\Box(\sim p \rightarrow \Diamond p)$ where "p" is an atomic proposition

i.e. Globally, whenever $\sim p$ happens (i.e. "0" occurs) then eventually "p" should happen (i.e. "1" should occur)

Q6

Q6(a)

Infinite schedule of these actors is possible.

From edge $A \to B$ we get $A = 2B$

From edge $B \to C$ we get $3B = 2C$

From edge $A \to C$ we get $3A = 4C$

3

So the simplest value satisfying above 3 equations are $A=4$, $B=2$ and $C=3$.

So the infinite schedule is $\underbrace{A, A, A, A, B, B, C, C, C}$, - - - Repeat infinitely often

Q6(b)

The infinite schedule is A,A,A,A,B,B,C,C,C, - - - -

After 4 A's The A→B channel would have 4 tokens
The A→C channel would have 12 tokens
The B→C channel would have 0 tokens

After 2 B's :- The A→B channel would have 0 tokens
The A→C channel would have 12 tokens
The B→C channel would have 6 tokens

After 3 C's :- The A→B channel would have 0 tokens
The A→C channel would have 0 tokens
The B→C channel would have 0 tokens

So the minimum size of buffer required is max(4+12, 12+6, 0) = 18

∴ $\boxed{\text{Minimum buffer size} = 18}$

Note:- Intermediate steps have not been mentioned because they are anyway less than the above mentioned steps.

0

Important: Use the following boxes only if you cancel one of your earlier answers. Mention the question number against the box you are using.

A3b

```
#variables
desiredTemperature_check : long integer
    | (Rest of variables code)

# User Input
 desiredTemperature_check = getUserInput()
 if (desiredTemperature_check > INT_MAX):
        displayErrorMessage(" Integer Overflow Error")
        break
 else :
     desiredTemperature = desired Temperature_check

# Temperature Control Logic
        | (Rest of the code)
```

4