**SYSC 4101A**

**Lab 7**

**Nathan MacDiarmid**

**101098993**

**Exercise 1**

**Question 1**

= []

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | a | b |
| **Output** | 0 | 0 | 1 |
| **Destination state** | X | Y | Z |

**Question 2**

= []

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Steps in test path** | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Source state** | Ø | X | Y | X | Y | Y | Z | Z | X | Z |
| **Input** | a | b | a | a | c | b | c | b | c | a |
| **Output** | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| **Destination state** | X | Y | X | Y | Y | Z | Y | X | Z | Z |

**Question 3**

:

= []

= []

= []

= []

= []

= []

= []

= []

= []

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | b | c |
| **Output** | 0 | 1 | 1 |
| **Destination state** | X | Y | Y |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | b | a |
| **Output** | 0 | 1 | 1 |
| **Destination state** | X | Y | X |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | b | b |
| **Output** | 0 | 1 | 1 |
| **Destination state** | X | Y | Z |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | a | b |
| **Output** | 0 | 0 | 1 |
| **Destination state** | X | Y | Z |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | a | a |
| **Output** | 0 | 0 | 1 |
| **Destination state** | X | Y | X |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Y |
| **Input** | a | a | c |
| **Output** | 0 | 0 | 1 |
| **Destination state** | X | Y | Y |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Z |
| **Input** | a | c | c |
| **Output** | 0 | 0 | 0 |
| **Destination state** | X | Z | Y |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Z |
| **Input** | a | c | b |
| **Output** | 0 | 0 | 0 |
| **Destination state** | X | Z | X |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Steps in test path** | | |
|  |  |  |  |
| **Source state** | Ø | X | Z |
| **Input** | a | c | a |
| **Output** | 0 | 0 | 1 |
| **Destination state** | X | Z | Z |

**Exercise 2**

**Question 1**

My = [] does not reveal this fault as mine transitions from state Y to state Z and the fault is transitioning from state Z to state Y.

**Question 2**

My = [] does reveal this fault as and would be doing the same transition. My test suite goes directly from state Z to state Z using and outputting 1, then goes from state Z to state Y using and outputting 0. Because the code implements and as the same transition from state Z to state Y, my test path will expect output 1 when running and then be able to run or , but it cannot as and both transition to the state Y result in the next to give a fault.

**Question 3**

My will not reveal the fault as the test paths are too short. While running tests, you are judging the correctness of the system based on the outputs. Since and give the correct outputs but end up in the wrong states and have no transitions afterward, there is no way to tell if the system is in the correct state based on how short the paths are.

**Exercise 3**

**Question 1**

My = [] would not detect the fault as it would not be in state Z and receive an input of c as it ends at state Z coming from Y.

**Question 2**

My = [] will not detect the fault because when it tries to run , although it will now transition to state T, it will still run transition with input a and output 1, but it will remain in state T. Since the test path ends there, it will not detect that it is in the wrong state as the input values resulted in the correct output values.

**Question 3**

My will not detect the fault as all the paths will result in the same output given the same input.

**Exercise 4**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Steps in test path** | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Source state** | Ø | X | Y | X | Y | Y | Z | Z | X | Z |
| **Input** | a | b | a | a | c | b | c | b | c | a |
| **Output** | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| **Destination state** | X | Y | X | Y | Y | Z | Y | X | Z | Z |

Based on the above table, {a} is not a characterization sequence as the output while in state Y and Z when given a as input will both result in 1. {b} is not a characterization sequence either as the output while in state X and Y with input b will also both give 1. {b, a} is not a characterization sequence either as the output while in state X or Z with input b then input a will result in 1 then 1 respectively. However, {a, b} is a characterization sequence as no states that are given input a then input b result in the same input.

**Exercise 5**

**Question 1**

= [,, ]

My test suite will detect the fault as is now the same as , so instead of transitioning back to Z, it transitions to Y. The old output would be 1 then 0, but now the output will be 1 then 1, revealing the fault.

**Question 2**

= [, ]

My test suite will still detect the fault as it detects it when transitioning from . It will be the exact same as Exercise 2 as the program will break and detect the fault before it gets to the appended {a, b} input transitions.

**Question 3**

:

= []

= []

= []

= []

= []

= []

= []

= []

= []

My test suite will detect the fault as for example will receive the correct output 1 with input a on transition . However, when given input b, it will expect 0 as the output as it expects to be coming from state Z, but it will return 1 as it is coming from state Y instead.

**Exercise 6**

For all parts in Exercise 6, now goes from state Z to state T.

**Question 1**

= [,, ]

My test suite will not detect the fault as the outputs would be the same as the original. If we added inputs {a, b, c} in addition to the added {a, b}, it would detect the fault.

**Question 2**

= [, ]

My suite will detect the fault on the transition from to as at this point it will be in state T. When a is passed, it should be in state Y and expect 1 as the output, but since it is in state T, it outputs 0, revealing the fault.

**Question 3**

:

= []

= []

= []

= []

= []

= []

= []

= []

= []

My test suite will not detect the fault as we need to run a test path that runs through transition as at that point it will be in state T. Since none of them have any input past , the program finishes at state T.