Question:Question: Exercise 2.1: Consider a modified version of the Delay component, called OddDelay, that has a Boo...

Exercise 2.1: Consider a modified version of the `Delay` component, called `OddDelay`, that has a Boolean input variable *in*, a Boolean output variable *out*, and two Boolean state variables x and y. Both the state variables are initialized to 0, and the reaction description is given by:

`if` y `then` $out := x$ `else` $out := 0$;
$x := in$;
$y := \neg y$.

Describe in words the behavior of the component `OddDelay`. List a possible execution of the component if it is supplied with the sequence of inputs $0, 1, 1, 0, 1, 1$ for the first six rounds. ■

Solution:
The component behaves with respect to the boolean function yx as when y is true then whatever is the value of x will be outputted or else output is zero as its an AND operation.
And the bit 'y' keeps flipping after every bit input of x.
Given input 0,1,1,0,1,1
assuming y=0 initially we get
output 1:- 0 (y is flipped as well ,now y:-1)
output 2:-1 (y is flipped as well now y:-0)
output 3:- 0 (y is flipped as well now y:-1)
output 4:- 0 (y is flipped as well now y:-0)
output 5:- 0 (y is flipped as well now y:-1)
output 6:- 1 (y is flipped as well now y:-0)

Question:Question: Exercise 2.2: Describe the component OddDelay from Exercise 2.1 as an extended-state machine with...

Exercise 2.2: Describe the component `OddDelay` from Exercise 2.1 as an extended-state machine with two modes. The mode of the state machine should capture the value of the state variable y, while the state variable x should be updated using assignments in the mode-switches. ■

Exercise 2.1: Consider a modified version of the `Delay` component, called `OddDelay`, that has a Boolean input variable *in*, a Boolean output variable *out*, and two Boolean state variables x and y. Both the state variables are initialized to 0, and the reaction description is given by:

$$\texttt{if } y \texttt{ then } out := x \texttt{ else } out := 0;$$
$$x := in;$$
$$y := \neg y.$$

Describe in words the behavior of the component `OddDelay`. List a possible execution of the component if it is supplied with the sequence of inputs $0, 1, 1, 0, 1, 1$ for the first six rounds. ■

!!! ANSWER !!!
As per chegg policy i am answering only one part..
2.1) The component behaves with respect to the boolean function yx as when y is true then whatever is the value of x will be outputted or else output is zero as its an AND operation.
And the bit 'y' keeps flipping after every bit input of x.
Given input 0,1,1,0,1,1
assuming y=0 initially we get
output 1:- 0 (y is flipped as well ,now y:-1)
output 2:-1 (y is flipped as well now y:-0)
output 3:- 0 (y is flipped as well now y:-1)
output 4:- 0 (y is flipped as well now y:-0)
output 5:- 0 (y is flipped as well now y:-1)
output 6:- 1 (y is flipped as well now y:-0)

Hope you like it..........
PLZ UPVOTE............ THANKS!!!!!!!!!!!!!!!!!!!

Question:Question: We want to design a reactive component with three Boolean input variables x, y, and reset and a B...

ANSWER

Before explaining the answer , A brief introduction about the relative components and its used types.

Reactive Components

A reactive component interacts with other components via inputs and outputs, executing in a sequence of rounds. The component maintains an internal state. In each round, it reads its inputs, and based on its current state and the inputs, it computes to produce outputs and to update the internal state.

We use typed variables to describe components. The commonly used types are:

_nat denoting the set of natural numbers;

_ int denoting the set of integers;

_ real denoting the set of real numbers;

_ bool denoting the Boolean valued type f0; 1g

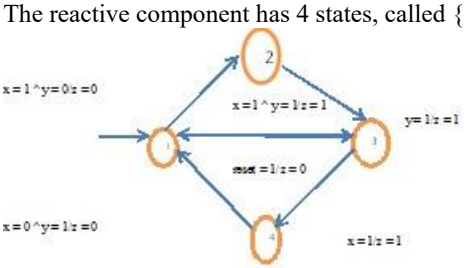
event denoting the enumerated type (\perp ,au)where T denotes that the event is present and denotes that the event is absent.

we allow the event type to be parameterized by another type, and the event can be absent, or can be present with a value of the parameter type.

We want to design a reactive component with three Boolean input variables x , y ,and reset and a Boolean output variable z . The desired behavior is the following. The component waits until it has encountered a round in which the input variable x is high and a round in which the input variable y is high, and as soon as both of these have been encountered, it sets the output z to high. It repeats the behavior when, in a subsequent round, the input variable reset is high. By default the output z is low. For instance, if x is high in rounds 2,3,7,12, y is high in rounds 5,6,10, and reset is high in round 9, then z should be high in rounds 5 and 12. Design a synchronous reactive component that captures this behavior. You may want to use the extended-state machine notation.

Solution:

The reactive component has 4 states, called {1, 2, 3, 4}, and its state machine is as in below figure



The component can be designed as follows:

• Input variables:

bool x, y, reset

• Output variables:

bool z

{1, 2, 3, 4} s : s = 1

(s = 4 ^ reset = 1) ! (s′ = 1 ^ z = 0)|

(((s = 1 ^ x = 1 ^ y = 1) _ (s = 2 ^ y = 1) _ (s = 3 ^ x = 1)) ! (s′ = 4 ^ z = 1)|

(((s = 1) ! ((x = 1) ! (s′ = 2)|((y = 1) ! (s′ = 3)|(s′ = 1)))|(s′ = s)) ^ (z = 0)))

Design an event-triggered combinational component ClockedMax with two input variables x and y of type nat and an input event variable clock . The output variable z of the component should be of type event (nat) such that the value of z should be the maximum of the inputs x and y during rounds in which clock is present.

Event-triggered Component

For a synchronous reactive component C = (I;O; S; Init;React), a set J\subset I of input variables is said to be a trigger if

1. all input variables in J are of event type,

2. every non-event output variable in O is latched, and

3. if i is an input with all events in J absent (that is, for all input variables

-cej, 1:0) =1) then for all states s, if s i=o t is a reaction then s = t and o(y) =\perp for all event output variables y.

A component C is said to be event-triggered if there exists a subset J\subset I of its input variables such that J is a trigger for C.

solution;

Clocked Delay is an event-triggered component: every time there is an event clock, the component saves the current input (variable x) in a state variable (s) and outputs the (old) value of s.

• Input variables:

bool x

event clock
• Output variables:
event(bool) y
bool s : s = 0
clock? ! (s′ = x ^ y = s)|(s′ = s ^ y =?)
----------I hope my answer met all your requirements.....Thank You----------

Question:Question: Question from principles of physical cyber system by rajeev alur 1) We want to design a reactive ...

Before explaining the answer , A brief introduction about the relative components and its used types.

Reactive Components

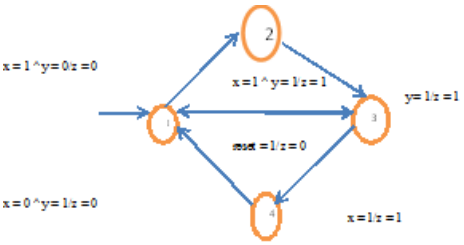
A reactive component interacts with other components via inputs and outputs, executing in a sequence of rounds. The component maintains an internal state. In each round, it reads its inputs, and based on its current state and the inputs, it computes to produce outputs and to update the internal state.

We use typed variables to describe components. The commonly used types are:

_nat denoting the set of natural numbers;
_ int denoting the set of integers;
_ real denoting the set of real numbers;
_ bool denoting the Boolean valued type f0; 1g
event denoting the enumerated type ($\perp, \tau$)where T denotes that the event is present and denotes that the event is absent.
we allow the event type to be parameterized by another type, and the event can be absent, or can be present with a value of the parameter type.
We want to design a reactive component with three Boolean input variables x , y ,and reset and a Boolean output variable z . The desired behavior is the following. The component waits until it has encountered a round in which the input variable x is high and a round in which the input variable y is high, and as soon as both of these have been encountered, it sets the output z to high. It repeats the behavior when, in a subsequent round, the input variable reset is high. By default the output z is low. For instance, if x is high in rounds 2,3,7,12, y is high in rounds 5,6,10, and reset is high in round 9, then z should be high in rounds 5 and 12. Design a synchronous reactive component that captures this behavior. You may want to use the extended-state machine notation.

Solution:

The reactive component has 4 states, called {1, 2, 3, 4}, and its state machine is as in below figure



The component can be designed as follows:

• Input variables:

bool x, y, reset

• Output variables:

bool z

{1, 2, 3, 4} s : s = 1
(s = 4 ^ reset = 1) ! (s′ = 1 ^ z = 0)| (((s = 1 ^ x = 1 ^ y = 1) _ (s = 2 ^ y = 1) _ (s = 3 ^ x = 1)) ! (s′ = 4 ^ z = 1)| (((s = 1) ! ((x = 1) ! (s′ = 2)|((y = 1) ! (s′ = 3)|(s′ = 1)))|(s′ = s)) ^ (z = 0)))

Design an event-triggered combinational component ClockedMax with two input variables x and y of type nat and an input event variable clock . The output variable z of the component should be of type event (nat) such that the value of z should be the maximum of the inputs x and y during rounds in which clock is present.

Event-triggered Component

For a synchronous reactive component C = (I;O; S; Init;React), a set $J \subset I$ of input variables is said to be a trigger if

1. all input variables in *J* are of event type,
2. every non-event output variable in *O* is latched, and
3. if i is an input with all events in J absent (that is, for all input variables $x \epsilon j, I(x) = \perp$) then for all states s, if s i=o t is a reaction then s = t and o(y) =$\perp$ for all event output variables y.

A component C is said to be event-triggered if there exists a subset $J \subset I$ of its input variables such that J is a trigger for C.

solution;

Clocked Delay is an event-triggered component: every time there is an event clock, the component saves the current input (variable x) in a state variable (s) and outputs the (old) value of s.

• Input variables:

bool x

event clock

• Output variables:

event(bool) y

bool s : s = 0
clock? ! (s′ = x ^ y = s)|(s′ = s ^ y =?)

Question:Question: Exercise 2.4: Consider the component OddDelay from exercise 2.1. Is the component finite-state? D...

Exercise 2.4: Consider the component `OddDelay` from exercise 2.1. Is the component finite-state? Draw the corresponding Mealy machine. ■

Exercise 2.1: Consider a modified version of the `Delay` component, called `OddDelay`, that has a Boolean input variable *in*, a Boolean output variable *out*, and two Boolean state variables x and y. Both the state variables are initialized to 0, and the reaction description is given by:

```
if y then out := x else out := 0;
x := in;
y := ¬y.
```

Describe in words the behavior of the component `OddDelay`. List a possible execution of the component if it is supplied with the sequence of inputs $0, 1, 1, 0, 1, 1$ for the first six rounds. ■

Answer :

exercise - 2.1)

The component behaves with respect to the boolean function yx as when y is true then whatever is the value of x will be outputted or else output is zero as its an AND operation.
And the bit 'y' keeps flipping after every bit input of x.
Given input 0,1,1,0,1,1
assuming y=0 initially we get
output 1:- 0 (y is flipped as well ,now y:-1)
output 2:-1 (y is flipped as well now y:-0)
output 3:- 0 (y is flipped as well now y:-1)
output 4:- 0 (y is flipped as well now y:-0)
output 5:- 0 (y is flipped as well now y:-1)
output 6:- 1 (y is flipped as well now y:-0)
.....
I provided you complete and correct answers for your question.So, please encourage me by giving an Upvote.
If u have any doubt or if u want anything else, please drop a comment here.
Thank you:)

Question:Question: Design an event-triggered combinational component ClockedMax with two input variables x and y of ...

Event-triggered Component

For a synchronous reactive component C = (I;O; S; Init;React), a set $J \subset I$ of input variables is said to be a trigger if
1. all input variables in *J* are of event type,
2. every non-event output variable in *O* is latched, and
3. if i is input with all events in J absent (that is, for all input variables $x \epsilon j, I(x) = \perp$) then for all states s, if s i=o t is a reaction then s = t and o(y) =$\perp$ for all event output variables y.

Component C is said to be event-triggered if there exists a subset $J \subset I$ of its input variables such that J is a trigger for C.

solution

Clocked Delay is an event-triggered component: every time there is an event clock, the component saves the current input (variable x) in a state variable (s) and outputs the (old) value of s.

• Input variables:
bool x
event clock
• Output variables:
event(bool) y
bool s : s = 0
clock? ! (s′ = x ^ y = s)|(s′ = s ^ y =?)

Question:Question: Design a nondeterministic component CounterEnv that supplies inputs to the counter of figure 2.9....

In deterministic algorithm, for a given particular input, the computer will always produce the same output going through the same states but in case of non-deterministic algorithm, for the same input, the compiler may produce different output in different runs. In fact non-deterministic algorithms can’t solve the problem in polynomial time and can’t determine what is the next step. The non-deterministic algorithms can show different behaviors for the same input on different execution and there is a degree of randomness to it.
Please provide image of 2.9 fig
Thank you!

Question:Question: Exercise 2.12 : Consider a synchronous reactive component C with an input variable x and output v...

No precedence constraint between A1 and A2:
Because, output y awaits input x, y is an output of task A1, output z doesn't await input x, z is an output of task A2.
So there is no need of task A1 needs to wait for the task A2 to be finished or task A2 needs to wait for the task A1 to be finished.
So, neither A1 < A2 nor A2 < A1 is true.
So both the tasks are independent of each other. There is no precedence constraint between the tasks A1 and A2.
P.S: If you are having any doubt, please comment here, I will surely reply you.Please ask questions seperately from next time. We will be provided very limited time for solving your query. Please understand and post the questions. Please provide your valuable feedback by a thumbsup. Your thumbsup is a result of our efforts. Your LIKES are very important for me so please LIKE…. Thanks in advance!
Note: I've answered according to chegg's policy

Question:Question: Exercise 2.13 : Consider the synchronous reactive component shown in figure 2.14. List all the po...

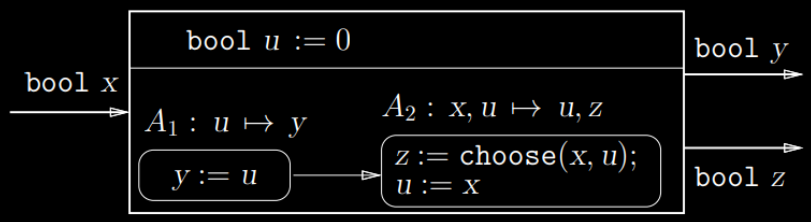


Figure 2.14: Component Specified Using Task Graph

Sol:- The possible reactions of the component are

(i) The given component is more deterministic.

(ii) Output Y is not dependent

(iii) Output Z is dependent on X.

(iv) It is input enabled, that is at least one transition has to be enabled for input.

(v) In some states there are 2 transitions,

∵ Z := choose (x, u).

From the diagram given, the expression for y

y := u.

∴ y is not dependent on x.

From the diagram given the expression for Z is

Z := choose (x, u)

∴ the Z depends on input x.

i.e. output Z awaits for x

Question:Question: Q1. a) Design a synchronous reactive component (SRC) ComputeAverage with an integer input variabl...

Q1. a) Design a synchronous reactive component (SRC) *ComputeAverage* with an integer input variable *x*, and input event variable *clock*, and a real-valued output variable *y* with the following behavior:

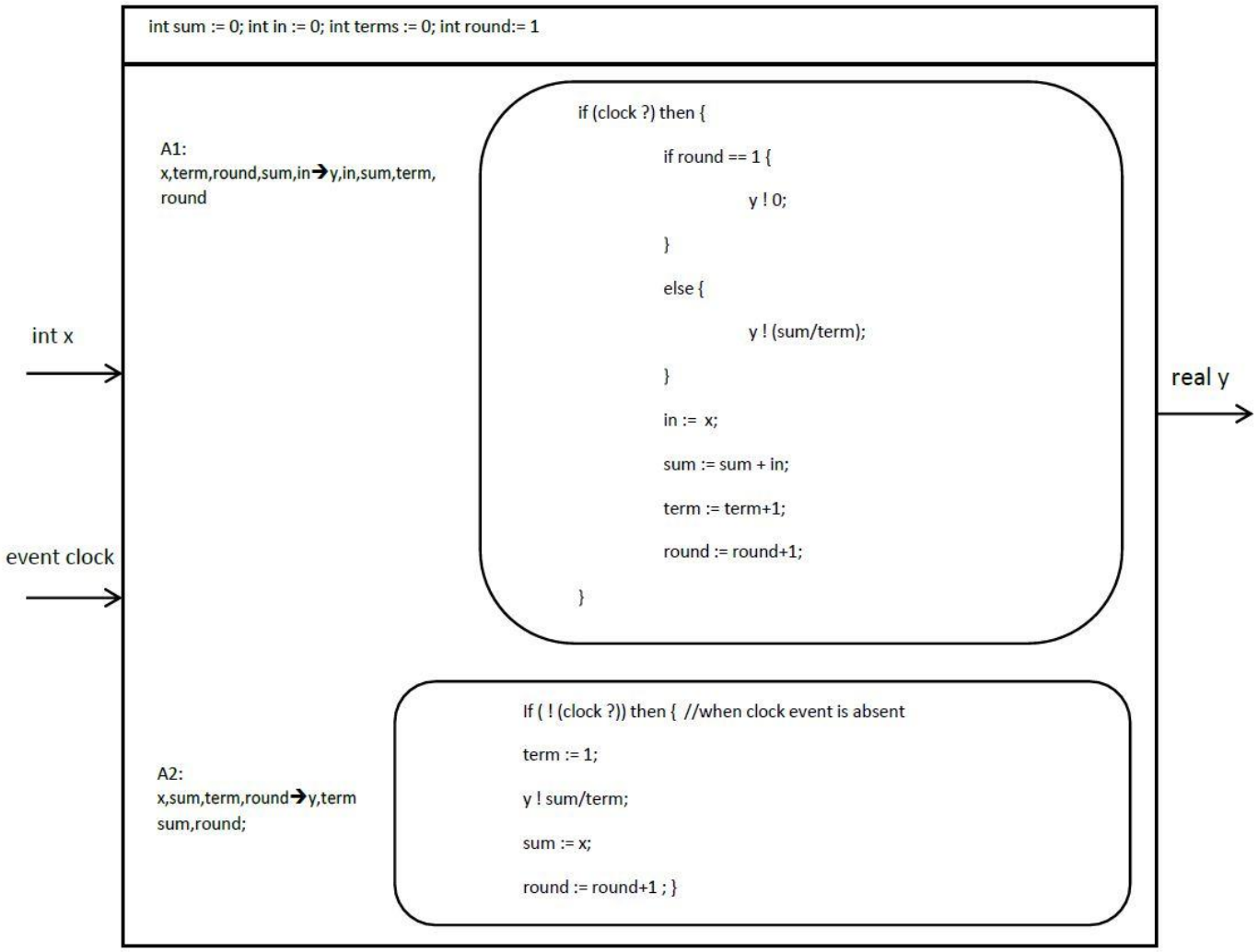
- **In the first round, the output *y* is 0.**
- **In a subsequent round *i*, let *j* < *i* be the most recent round before round *i* in which the input event *clock* is present (if *clock* is absent in all rounds before *i*, then let *j* = 0), the output should be the average of input values for *x* in round *j*, *j* + *1*, up to *i* – *1*.**
- **The component should be designed such that the output *y* does not await any input variables.**

b) The following is a sample behavior of the desired component:

Clock	⊥	⊥	T	⊥	⊥	⊥	T	⊥
x	*5*	*2*	*-3*	*1*	*6*	*5*	*-2*	*11*
y	*0*	*5*	*3.5*	*-3*	*-1*	*1.33*	*2.25*	*-2*

Show that your design of SRC produces matched sample behavior execution for each step using an execution diagram.

Here is an answer to your above question :
Answer:



The component has an integer input variable x and an event clock. It has state variables namely int sum, int in, int terms, and int round which keeps track of a number of rounds. The component is represented as a combination of two task graphs that are independent of one another. Task A1 is executed when the clock event is present and Task A2 is executed when the clock event is absent. Initially, the output y is zero whether the input event is present or not. If the clock event is present, then we assign y the average value till the previous round, and

post this we update our state variables. When the clock event is absent, we reset the number of terms traversed to 1, reset the sum to the input variable x, and update the round respectively.
In our SRC we can observe that the output y is not waiting for the input variable x and event clock directly.