

CMPE 117 - Embedded Software

Homework 5

Spring Quarter 2003

Due Tuesday May 20

Question 1 [20 points]

Write an Esterel module `waitAB` with two inputs, `A` and `B`, and four outputs, `O1`, `O2`, `U1`, `U2`, that behaves as follows:

- When the module receives `A`, it should emit `O1` at the next tick, and `O2` after one more tick.
- When the module receives `B`, it should emit `U1` at the next tick, and `U2` after one more tick.
- If `A` or `B` are received while the output sequences are still being generated, the module can behave arbitrarily.

An example of behavior is as follows:

input A:		A							A			...
input B:						B						...
outputs O1,O2:			O1	O2						O1	O2	...
outputs U1,U2:							U1	U2				...

Question 2, Part 1 [5 pts] Write a module that behaves as specified above.

Question 2, Part 2 [15 pts] Modify the previous module `waitAB`, so that if `A` or `B` are received simultaneously, or while the sequences `O1`, `O2` and `U1`, `U2` are still being produced, an error signal `E` is emitted. Once `E` is emitted, it does not matter how the module behaves for the next two ticks: in particular, you are free to choose whether the output sequences `O1`, `O2` or `U1`, `U2` should be completed, or aborted. However, the module should return to normal functioning within 3 ticks from the last received input. Below is an example of behavior, where “?” denotes “don’t care” values:

input A:		A					A										...	
input B:			B				B		B				B	B			...	
outputs O1,O2:			O1	?				?	?								...	
outputs U1,U2:				?	?			?	?		U1	U2			?	?	?	...
output E:				E				E							E			...

Question 2 [20 points]

Write a module `controlM` that has inputs `R` (reset) and `B` (busy). The module must run a process `M`, which can be easily done with the instruction `run M`, and:

- when neither of `B` or `R` is present, `M` should be left to run;
- when `B` alone is present, the execution of `M` should be suspended;
- when `R` alone is present, the execution of `M` should be aborted, and restarted;
- when both `B` and `R` are present, the execution of `M` should be aborted, and `M` should be restarted as soon as both `R` `B` are absent.

Question 3 [20 points]

Write a module with inputs `A`, `B`, `R` and outputs `AF`, `BF`, `AB`. The input `R` is a reset signal. When `R` is present, the module emits:

- `AF`, if `A` was received first (before `B`) since the last reset;
- `BF`, if `B` was received first (before `A`) since the last reset;
- `AB`, if `A` and `B` were first received simultaneously since the last reset;
- no signal, if neither `A` nor `B` have occurred since the last reset.

When `R` occurs simultaneously with `A` or `B`, you are free to consider the `A` or `B` as belonging to the cycle concluded by the present occurrence of `R`, or to the next reset cycle. The following is an example of behavior, where the `A` and `B` that occur simultaneously with `R` are counted as belonging to the cycle concluded by that instance of `R`:

input A:			A				A				A	A	...	
input B:				B			B	B		B			...	
input R:		R				R			R		R	R	R	...
output:						AF			AB		BF	AF	...	

Question 4 [20 points]

Consider the following module:

```

module ABerr :
  input A, B;
  output E, O;

  loop
    trap T in
      await A;
      present B then
        exit T
      end;
      await B;
      present A then
        exit T
      end;
      emit O;
    handle T do
      emit E;
    end trap
  end loop
end module

```

Question 4, Part 1 [5 pts] Describe informally the behavior of the module.

Question 4, Part 2 [15 pts] Complete the following table indicating the sequence of outputs emitted:

input A:		A				A			A	A	A		...
input B:			B		B		B			B		B	...
output O:													...
output E:													...

Question 5 [20 points]

Consider the following portions of code. Do they satisfy constructive casualty? Explain.

Question 5, Part 1 [5 pts]

```
module M:
output A, B;
  present A then emit B end
||
  present B else emit A end
end module
```

Question 5, Part 2 [5 pts]

```
module M:
output A, B;
  present A then emit A else emit A end
||
  present A then emit B end
||
  emit B
end module
```

Question 5, Part 3 [5 pts]

```
module M:
output A, B, C;
  present A then emit B else emit C end
||
  present B then emit A end
||
  emit A
end module
```

Question 5, Part 4 [5 pts]

```
module M:
output A, B, C;
  present A then emit B else emit B end
||
  present B else emit C end
||
  present C then emit A
||
  emit C
end module
```