CS628 Spring 2001 End-term Examination

Time: 3 hrs | Total marks: 100

- ***The exam is open book, notes, papers and other reading material brought into exam hall.***
- ***Results/proofs covered in class/problem sessions/assignments may simply be cited, unless specifically asked for.***
- ***Unnecessarily lengthy solutions will be penalized.***
- ***If you need to make any assumptions, state them clearly.***

1. *[10 + 10 marks]* A *k-Sequencer*, $\mathbf{S_k}$, is a process with $k+1$ input lines: $\mathbf{a_1}, \ldots, \mathbf{a_k}$ and $\mathbf{b}$, and k output lines: $\mathbf{c_1}, \ldots, \mathbf{c_k}$. Its behavior is defined as follows:

$$
\begin{aligned}
\mathbf{S_k} \;=\; \mathbf{b}? \rightarrow\; & ((\mathbf{a_1}? \rightarrow \mathbf{c_1}! \rightarrow \mathbf{S_k}) \; [] \\
& (\mathbf{a_2}? \rightarrow \mathbf{c_2}! \rightarrow \mathbf{S_k}) \; [] \\
& \vdots \; [] \\
& (\mathbf{a_k}? \rightarrow \mathbf{c_k}! \rightarrow \mathbf{S_k})).
\end{aligned}
$$

An *n-Mixer*, $\mathbf{M_n}$, is a process with $n+1$ input lines: $\mathbf{a_1}, \ldots \mathbf{a_n}$ and m_{in}, and $n+1$ output lines: $\mathbf{c_1}, \ldots \mathbf{c_n}$ and m_{out}. Its behavior is defined as follows:

$$
\begin{aligned}
\mathbf{M_n} \;=\; & (\mathbf{a_1}? \rightarrow \mathbf{m_{out}}! \rightarrow \mathbf{m_{in}}? \rightarrow \mathbf{c_1}! \rightarrow \mathbf{M_n}) \; [] \\
& (\mathbf{a_2}? \rightarrow \mathbf{m_{out}}! \rightarrow \mathbf{m_{in}}? \rightarrow \mathbf{c_2}! \rightarrow \mathbf{M_n}) \; [] \\
& \vdots \; [] \\
& (\mathbf{a_n}? \rightarrow \mathbf{m_{out}}! \rightarrow \mathbf{m_{in}}? \rightarrow \mathbf{c_n}! \rightarrow \mathbf{M_n})
\end{aligned}
$$

   (a) Implement a k-Sequencer using one or more k-Mixer(s) and as few and as simple additional elements/processes as possible.

   (b) Implement a k-Sequencer using only m-Mixers and n-Sequencers, where $m < k$ and $1 < n < k$.

2. *[10 + 15 + 10 marks]* Consider the extended burst-mode machine in Fig. 1. The machine has one

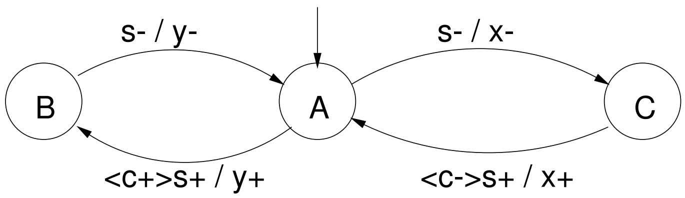

Figure 1:

conditional input, $\mathbf{c}$, that is "sampled" by $\mathbf{s}$. Suppose we know further that $\mathbf{c}$ does not change value between $\mathbf{s}+$ and $\mathbf{s}-$. However, it may change arbitrarily many times between $\mathbf{s}-$ and $\mathbf{s}+$, eventually stabilizing to 1 or 0 before the next $\mathbf{s}+$ arrives.

(a) Give a K-map design of the above machine taking into account the additional knowledge about the behavior of **c**. Your solution should be a set of K-maps, that show for each valid combination of inputs, outputs and state variables, the next values of the outputs and state variables. You must fill in only those squares in the K-maps that are needed for the XBM transitions. All squares that represent invalid combinations of inputs, outputs and state variables during the course of the XBM transitions must be left blank.

(b) Assuming that all signals are **0** initially, represent the interaction of the machine with its environment using a Petri Net.

(c) Give a trace-structure describing the above interaction. You must label state transition arcs by names of signals on which events occur, and not by names of the events themselves. Thus, your labels should be **c**, **x**, ... instead of **c**−, **x**+, ... You may omit transitions to the *Failure* state.

3. *[15 marks]* Consider the same extended burst-mode machine as in Fig. 1, but do not assume the additional information about the behavior of **c** used in the previous problem. Instead, suppose we know the following timing constraints about the machine and its environment.

   - Setup and hold constraints: Input **c** must remain stable for at least 2 time units before and until at least 1 time unit after being sampled by **s**.
   - Fundamental-mode constraint: An input burst must start at least 2 time units after the end of the previous output burst.
   - Feed-forward delay of machine: Once an input burst has arrived, the machine takes between 3 and 5 time units to start the corresponding output burst.

   Give a timed automaton representing the interaction of the machine with its environment.

4. *[15 + 15 marks]* Consider the control network of an $n-$ stage micropipeline. If k requests are sent from the sender's end, eventually k requests (and no more) arrive at the receiver's end. Similarly, if k acknowledgments are sent from the receiver's end, eventually k (and no more) arrive at the sender's end.

   Suppose we wish to bypass this restriction: We wish to have a micropipeline in which for every request sent from the sender's end, 4 requests are generated at the receiver's end. Similarly, for every 4 acknowledgments generated from the receiver's end, one acknowledgment is generated at the sender's end.

   (a) Suggest a scheme for achieving this while preserving the simple, elegant structure of micropipelines.

   (b) Give an estimate (with justification) of the number of states in the trace-structure of such a micropipeline. Clearly state the design assumptions and parameters that you are using to derive the estimate.