T-79.179 **Spring 2005**
Parallel and Distributed Digital Systems
Exercise 5 Solutions
21.3-3.4 2005

1. For a detailed overview of the μCRL language, see Chapter 2 of the manual. The μCRL specification for the two buffers put in a sequence looks like:

```
sort Bool
func T,F: -> Bool
map  and,or: Bool # Bool -> Bool
     not: Bool -> Bool
     eq: Bool # Bool -> Bool
     var x:Bool
rew  and(T,T)=T
     and(F,x)=F
     and(x,F)=F
     or(T,x)=T
     or(x,T)=T
     or(F,F)=F
     not(F)=T
     not(T)=F
     eq(T,T)=T
     eq(T,F)=F
     eq(F,T)=F
     eq(F,F)=T

sort D
func d1,d2: -> D
map  eq: D # D -> Bool
rew  eq(d1,d1)=T
     eq(d2,d2)=T
     eq(d1,d2)=F
     eq(d2,d1)=F
```



```
act
     r1,s2,r3,s3,c3: D

comm
     s3|r3=c3

proc
     Q1 = sum(d:D,r1(d).s3(d).Q1)
     Q2 = sum(d:D,r3(d).s2(d).Q2)

init  hide({c3},encap({s3,r3}, Q2 || Q1))
```

Suppose **buffers.mcrl** is the file name of the above declaration. Instruction
mcrl buffers.mcrl
determines whether the specification is a correct μCRL declaration.

Most tools of the μCRL tool set require specifications in a so-called Linear Process Operator (LPO) format. The specification can be linearised to LPO format, e.g., by the instruction:
mcrl -tbf -regular buffers.mcrl
This instruction produces a file **buffers.tbf** which contains an LPO. The file **buffer.tbf** can be studied, e.g., with the pretty printer by the instruction:
pp buffer.tbf .

2. From a file **buffers.tbf**, instruction
instantiator buffers.tbf
generates a file **buffers.aut** which contains the corresponding state space. This can be studied, e.g., by instruction
less buffers.aut.

3. From a file **buffers.tbf**, instruction
instantiator -deadlock buffers.tbf
generates a file **buffers.dlk** which contains the list of deadlocking states. This can be studied, e.g., by instruction
less buffers.dlk .