A Concurrent Cyclic Buffer

John C. Reynolds
Carnegie Mellon University

March 22, 2011

(Based on class notes dated March 15, 2002)

We assume that an n-element array (where $\mathsf{n} \geq 1$) has been allocated at location l. These variables will be used throughout our program but never assigned to. We will also use the variables

f	number of free elements
b	number of busy elements
$\mathsf{i}, \mathsf{i}_f, \mathsf{i}_b$	pointers to oldest free element
$\mathsf{j}, \mathsf{j}_f, \mathsf{j}_b$	pointers to oldest busy element
x, y	values to be placed in or removed from buffers

Without mentioning it explicitly in our assertions, we assume that the variables $\mathsf{i}, \mathsf{i}_f, \mathsf{i}_b, \mathsf{j}, \mathsf{j}_f, \mathsf{j}_b$ will range over the integers between l and $\mathsf{l} + \mathsf{n} - 1$ inclusive.

We write $x =_{\mathsf{n}} y$ when x and y are equal modulo n. We write $\oplus$ for the modular addition operator such that $x \oplus y =_{\mathsf{n}} x + y$ and $\mathsf{l} \leq x \oplus y < \mathsf{l} + \mathsf{n}$.

—

We introduce $\mathbf{resource}\ \mathsf{free}(\mathsf{f}, \mathsf{i}_f, \mathsf{j}_f)$, with the invariant

$$RI_{\mathsf{free}} \equiv 0 \leq \mathsf{f} \leq \mathsf{n} \wedge \mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} \wedge \bigodot_{\mathsf{k}=0}^{\mathsf{f}-1} (\mathsf{i}_f \oplus \mathsf{k}) \mapsto -.$$

Then

$$\begin{array}{l}
\{((\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f) \ * \ RI_{\mathsf{free}}) \wedge \mathsf{f} > 0\} \\
\{((\mathsf{i} \mapsto -) \wedge \mathsf{i} = \mathsf{i}_f) \ * \\
\quad (0 \leq \mathsf{f} - 1 \leq \mathsf{n} \wedge \mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f \oplus 1 + \mathsf{f} - 1 \wedge \bigodot_{\mathsf{k}=1}^{\mathsf{f}-1} (\mathsf{i}_f \oplus \mathsf{k}) \mapsto -)\} \\
\mathsf{f} := \mathsf{f} - 1 \ ; \mathsf{i}_f := \mathsf{i}_f \oplus 1 \\
\{((\mathsf{i} \mapsto -) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f) \ * \ RI_{\mathsf{free}}\},
\end{array}$$

so that

$$\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f\} \\
\mathsf{getfree} & \equiv & \mathbf{with}\ \mathsf{free}\ \mathbf{when}\ \mathsf{f} > 0\ \mathbf{do} \\
 & & \quad (\mathsf{f} := \mathsf{f} - 1 \ ; \mathsf{i}_f := \mathsf{i}_f \oplus 1) \\
 & & \{(\mathsf{i} \mapsto -) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f\}.
\end{array}$$

—

$$RI_{\mathsf{free}} \equiv 0 \leq \mathsf{f} \leq \mathsf{n} \wedge \mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} \wedge \bigodot_{\mathsf{k}=0}^{\mathsf{f}-1}(\mathsf{i}_f \oplus \mathsf{k}) \mapsto -.$$

Also,

$$\{((\mathsf{j} \mapsto -) \wedge \mathsf{j} = \mathsf{j}_f) \ * \ RI_{\mathsf{free}}\} \tag{1}$$

$$\{(\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_f) \ *$$

$$(0 \leq \mathsf{f}+1 \leq \mathsf{n} \wedge \mathsf{j}_f \oplus 1 =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} + 1 \wedge \bigodot_{\mathsf{k}=0}^{\mathsf{f}}(\mathsf{i}_f \oplus \mathsf{k}) \mapsto -)\} \tag{2}$$

$$\mathsf{f} := \mathsf{f}+1 \ ; \mathsf{j}_f := \mathsf{j}_f \oplus 1$$

$$\{(\mathbf{emp} \wedge \mathsf{j} \oplus 1 = \mathsf{j}_f) \ * \ RI_{\mathsf{free}}\},$$

To justify $\mathsf{f}+1 \leq \mathsf{n}$ in (2), we must show that (1) implies $\mathsf{f} \neq \mathsf{n}$. Assume the contrary and $n \geq 1$. Then

$$\bigodot_{\mathsf{k}=0}^{\mathsf{f}-1}(\mathsf{i}_f \oplus \mathsf{k}) \mapsto - \quad \text{implies} \quad \bigodot_{\mathsf{k}=0}^{\mathsf{n}-1}(\mathsf{i}_f \oplus \mathsf{k}) \mapsto -$$

$$\text{implies} \quad \mathsf{i}_f \mapsto - \ * \ \mathbf{true}$$

$$\mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} \quad \text{implies} \quad \mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f + \mathsf{n}$$

$$\text{implies} \quad \mathsf{j}_f = \mathsf{i}_f$$

(since $\mathsf{l} \leq \mathsf{j}_f, \mathsf{i}_f \leq \mathsf{l} + \mathsf{n} - 1$), so that

$$((\mathsf{j} \mapsto -) \wedge \mathsf{j} = \mathsf{j}_f) \wedge \mathsf{j}_f = \mathsf{i}_f \text{ implies } \mathsf{i}_f \mapsto -$$

and thus

$$(1) \text{ implies } (\mathsf{i}_f \mapsto -) \ * \ (\mathsf{i}_f \mapsto - \ * \ \mathbf{true})$$

But this is false.

$$RI_{\mathsf{free}} \equiv 0 \leq \mathsf{f} \leq \mathsf{n} \wedge \mathsf{j}_f =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} \wedge \bigodot_{\mathsf{k}=0}^{\mathsf{f}-1} (\mathsf{i}_f \oplus \mathsf{k}) \mapsto -.$$

Also,

$$\{((\mathsf{j} \mapsto -) \wedge \mathsf{j} = \mathsf{j}_f) \ * \ RI_{\mathsf{free}}\} \tag{1}$$

$$\{(\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_f) \ * \\ (0 \leq \mathsf{f}+1 \leq \mathsf{n} \wedge \mathsf{j}_f \oplus 1 =_{\mathsf{n}} \mathsf{i}_f + \mathsf{f} + 1 \wedge \\ \bigodot_{\mathsf{k}=0}^{\mathsf{f}} (\mathsf{i}_f \oplus \mathsf{k}) \mapsto -)\} \tag{2}$$

$$\mathsf{f} := \mathsf{f}+1 \ ; \mathsf{j}_f := \mathsf{j}_f \oplus 1$$

$$\{(\mathbf{emp} \wedge \mathsf{j} \oplus 1 = \mathsf{j}_f) \ * \ RI_{\mathsf{free}}\},$$

so that

$$\begin{array}{rl} & \{(\mathsf{j} \mapsto -) \wedge \mathsf{j} = \mathsf{j}_f\} \\ \mathsf{putfree} \ \equiv & \mathbf{with}\ \mathsf{free}\ \mathbf{when\ true\ do} \\ & \quad (\mathsf{f} := \mathsf{f}+1 \ ; \mathsf{j}_f := \mathsf{j}_f \oplus 1) \\ & \{\mathbf{emp} \wedge \mathsf{j} \oplus 1 = \mathsf{j}_f\}. \end{array}$$

—

Next we introduce $\mathbf{resource}\ \mathsf{busy}(\mathsf{b}, \mathsf{j}_b, \mathsf{i}_b)$, with invariant

$$RI_{\mathsf{busy}} \equiv 0 \leq \mathsf{b} \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} \wedge \bigodot_{\mathsf{k}=0}^{\mathsf{b}-1} (\mathsf{j}_b \oplus \mathsf{k}) \mapsto -.$$

Then

$$\begin{array}{l}
\{((\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b) \ * \ RI_{\mathsf{busy}}) \wedge \mathsf{b} > 0\} \\
\{((\mathsf{j} \mapsto -) \wedge \mathsf{j} = \mathsf{j}_b) \ * \\
\quad (0 \leq \mathsf{b} - 1 \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b \oplus 1 + \mathsf{b} - 1 \wedge \bigodot_{\mathsf{k}=1}^{\mathsf{b}-1} (\mathsf{j}_b \oplus \mathsf{k}) \mapsto -)\} \\
\mathsf{b} := \mathsf{b} - 1 \ ; \mathsf{j}_b := \mathsf{j}_b \oplus 1 \\
\{((\mathsf{j} \mapsto -) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b) \ * \ RI_{\mathsf{busy}}\},
\end{array}$$

so that

$$\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b\} \\
\mathsf{getbusy} & \equiv & \mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathsf{b} > 0\ \mathbf{do} \\
 & & \quad (\mathsf{b} := \mathsf{b} - 1 \ ; \mathsf{j}_b := \mathsf{j}_b \oplus 1) \\
 & & \{(\mathsf{j} \mapsto -) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b\}.
\end{array}$$

—

$$RI_{\mathsf{busy}} \equiv 0 \leq \mathsf{b} \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} \wedge \odot_{\mathsf{k}=0}^{\mathsf{b}-1}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto -.$$

Also,

$$\begin{array}{l}
\{((\mathsf{i} \mapsto -) \wedge \mathsf{i} = \mathsf{i}_b) \ * \ RI_{\mathsf{busy}}\} \\
\{(\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_b) \ * \\
\quad (0 \leq \mathsf{b}+1 \leq \mathsf{n} \wedge \mathsf{i}_b \oplus 1 =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} + 1 \wedge \odot_{\mathsf{k}=0}^{\mathsf{b}}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto -)\} \\
\mathsf{b} := \mathsf{b}+1 \ ; \ \mathsf{i}_b := \mathsf{i}_b \oplus 1 \\
\{(\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b) \ * \ RI_{\mathsf{busy}}\},
\end{array}$$

so that

$$\begin{array}{rcl}
 & & \{(\mathsf{i} \mapsto -) \wedge \mathsf{i} = \mathsf{i}_b\} \\
\mathsf{putbusy} & \equiv & \mathbf{with}\ \mathsf{busy}\ \mathbf{when\ true\ do} \\
 & & \quad (\mathsf{b} := \mathsf{b}+1 \ ; \ \mathsf{i}_b := \mathsf{i}_b \oplus 1) \\
 & & \{\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b\}.
\end{array}$$

Note the symmetry: The second resource, its invariant, and its critical regions can be obtained from the first (or vice-versa) by interchanging free and busy (both as region names and as subscripts), interchanging f and b (both as variables and as subscripts), and interchanging i and j (as possibly subscripted variables).

—

Next, our first process is

$$
\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b\} \\
\mathsf{p}_1 & \equiv & \mathbf{while}\ \mathsf{test}_1\ \mathbf{do} \\
 & & \quad \big(\{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b\} \\
 & & \quad \mathsf{Produce\ x}\ ; \\
 & & \quad \mathsf{getfree}\ ; \\
 & & \quad \{(\mathsf{i} \mapsto -) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b\} \\
 & & \quad [\mathsf{i}] := \mathsf{x}\ ; \\
 & & \quad \{(\mathsf{i} \mapsto -) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b\} \\
 & & \quad \mathsf{putbusy}\ ; \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b\} \\
 & & \quad \mathsf{i} := \mathsf{i} \oplus 1 \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b\}\big) \\
 & & \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \neg\, \mathsf{test}_1\}.
\end{array}
$$

Here we can use the resource variables i_f and i_b in assertions, since they will not be set by the second process p_2, which will only execute the critical regions getbusy and putfree.

—

Similarly, in the second process, we will be able to use the resource variables j_f and j_b, since they are not set by the first process, which only executes the critical regions getfree and putbusy:

$$
\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f\} \\
\mathsf{p}_2 & \equiv & \mathbf{while}\ \mathsf{test}_2\ \mathbf{do} \\
 & & \quad \big(\{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f\} \\
 & & \quad \mathsf{getbusy}\ ; \\
 & & \quad \{(\mathsf{j} \mapsto -) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f\} \\
 & & \quad \mathsf{y} := [\mathsf{j}]\ ; \\
 & & \quad \{(\mathsf{j} \mapsto -) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f\} \\
 & & \quad \mathsf{putfree}\ ; \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} \oplus 1 = \mathsf{j}_f\} \\
 & & \quad \mathsf{j} := \mathsf{j} \oplus 1\ ; \\
 & & \quad \mathsf{Consume\ y} \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f\}\big) \\
 & & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \neg\, \mathsf{test}_2\}.
\end{array}
$$

Except for the occurrence of “Produce” in one and “Consume” in the other, the two process are symmetric in the same sense as the two resources.

—

Finally, we have the main program

$$
\begin{array}{l}
\{\odot_{\mathsf{k}=0}^{\mathsf{n}-1}(\mathsf{l}+\mathsf{k}) \mapsto -\} \\
\mathsf{f} := \mathsf{n}\ ;\ \mathsf{b} := 0\ ; \\
\mathsf{i} := \mathsf{l}\ ;\ \mathsf{i}_f := \mathsf{l}\ ;\ \mathsf{i}_b := \mathsf{l}\ ;\ \mathsf{j} := \mathsf{l}\ ;\ \mathsf{j}_f := \mathsf{l}\ ;\ \mathsf{j}_b := \mathsf{l}\ ; \\
\{RI_{\mathsf{free}} \ * \ RI_{\mathsf{busy}} \ * \ (\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b) \ * \\
\qquad (\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f)\} \\
\mathbf{resource}\ \mathsf{free}(\mathsf{f}, \mathsf{i}_f, \mathsf{j}_f), \mathsf{busy}(\mathsf{b}, \mathsf{j}_b, \mathsf{i}_b)\ \mathbf{in}\ (\mathsf{p}_1 \parallel \mathsf{p}_2) \\
\{RI_{\mathsf{free}} \ * \ RI_{\mathsf{busy}} \ * \ (\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \neg\, \mathsf{test}_1) \ * \\
\qquad (\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \neg\, \mathsf{test}_1)\}.
\end{array}
$$

—

Deleting assertions and putting everything together, we get

$$
\begin{array}{l}
\mathsf{f} := \mathsf{n} \;;\; \mathsf{b} := 0 \;; \\
\mathsf{i} := \mathsf{I} \;;\; \mathsf{i}_f := \mathsf{I} \;;\; \mathsf{i}_b := \mathsf{I} \;;\; \mathsf{j} := \mathsf{I} \;;\; \mathsf{j}_f := \mathsf{I} \;;\; \mathsf{j}_b := \mathsf{I} \;; \\
\textbf{resource}\ \mathsf{free}(\mathsf{f}, \mathsf{i}_f, \mathsf{j}_f), \mathsf{busy}(\mathsf{b}, \mathsf{j}_b, \mathsf{i}_b)\ \textbf{in} \\
\quad \big(\textbf{while}\ \mathsf{test}_1\ \textbf{do} \\
\qquad \big(\mathsf{Produce}\ \mathsf{x} \;; \\
\qquad \textbf{with}\ \mathsf{free}\ \textbf{when}\ \mathsf{f} > 0\ \textbf{do} \\
\qquad\quad (\mathsf{f} := \mathsf{f} - 1 \;;\; \mathsf{i}_f := \mathsf{i}_f \oplus 1) \;; \\
\qquad [\mathsf{i}] := \mathsf{x} \;; \\
\qquad \textbf{with}\ \mathsf{busy}\ \textbf{when true do} \\
\qquad\quad (\mathsf{b} := \mathsf{b} + 1 \;;\; \mathsf{i}_b := \mathsf{i}_b \oplus 1) \;; \\
\qquad \mathsf{i} := \mathsf{i} \oplus 1\big) \\
\quad \| \ \textbf{while}\ \mathsf{test}_2\ \textbf{do} \\
\qquad \big(\textbf{with}\ \mathsf{busy}\ \textbf{when}\ \mathsf{b} > 0\ \textbf{do} \\
\qquad\quad (\mathsf{b} := \mathsf{b} - 1 \;;\; \mathsf{j}_b := \mathsf{j}_b \oplus 1) \;; \\
\qquad \mathsf{y} := [\mathsf{j}] \;; \\
\qquad \textbf{with}\ \mathsf{free}\ \textbf{when true do} \\
\qquad\quad (\mathsf{f} := \mathsf{f} + 1 \;;\; \mathsf{j}_f := \mathsf{j}_f \oplus 1) \;; \\
\qquad \mathsf{j} := \mathsf{j} \oplus 1 \;; \\
\qquad \mathsf{Consume}\ \mathsf{y}\big)\big)
\end{array}
$$

—

It is easy to see that $\mathsf{i}_f, \mathsf{i}_b, \mathsf{j}_f, \mathsf{j}_b$ are all auxilliary. Removing the assignments to these variables gives:

$$
\begin{array}{l}
\mathsf{f} := \mathsf{n}\ ;\ \mathsf{b} := 0\ ;\ \mathsf{i} := \mathsf{l}\ ;\ \mathsf{j} := \mathsf{l}\ ; \\
\mathbf{resource}\ \mathsf{free}(\mathsf{f}), \mathsf{busy}(\mathsf{b})\ \mathbf{in} \\
\quad (\mathbf{while}\ \mathsf{test}_1\ \mathbf{do} \\
\qquad (\mathsf{Produce\ x}\ ; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when}\ \mathsf{f} > 0\ \mathbf{do}\ \mathsf{f} := \mathsf{f} - 1\ ; \\
\qquad [\mathsf{i}] := \mathsf{x}\ ; \\
\qquad \mathbf{with}\ \mathsf{busy}\ \mathbf{when\ true\ do}\ \mathsf{b} := \mathsf{b} + 1\ ; \\
\qquad \mathsf{i} := \mathsf{i} \oplus 1) \\
\quad \| \ \mathbf{while}\ \mathsf{test}_2\ \mathbf{do} \\
\qquad (\mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathsf{b} > 0\ \mathbf{do}\ \mathsf{b} := \mathsf{b} - 1\ ; \\
\qquad \mathsf{y} := [\mathsf{j}]\ ; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when\ true\ do}\ \mathsf{f} := \mathsf{f} + 1\ ; \\
\qquad \mathsf{j} := \mathsf{j} \oplus 1\ ; \\
\qquad \mathsf{Consume\ y}))
\end{array}
$$

—

Keeping Track of the Buffer Contents

We introduce the following sequence variables:

- α the sequence of values that have been produced
- β the sequence of values that have been consumed
- γ the sequence of values currently stored in the buffer

along with predicates "produced" and consumed" satisfying

$$\begin{array}{lll} \{\text{produced } \alpha\} & & \{\text{consumed } \beta\} \\ \text{Produce x} & \text{and} & \text{Consume y} \\ \{\text{produced } \alpha{\cdot}\text{x}\} & & \{\text{consumed } \beta{\cdot}\text{y}\} \end{array}$$

Then we add these variables to $\mathbf{resource}$ $\text{busy}(\text{b}, \text{j}_b, \text{i}_b, \alpha, \beta, \gamma)$, and modify its invariant:

$$RI_{\text{busy}} \equiv 0 \leq \text{b} \leq \text{n} \wedge \text{i}_b =_{\text{n}} \text{j}_b + \text{b} \wedge \underline{\alpha = \beta{\cdot}\gamma \wedge \#\gamma = \text{b}} \wedge \bigodot_{\text{k}=0}^{\text{b}-1}(\text{j}_b \oplus \text{k}) \mapsto \underline{\gamma_{\text{k}}}.$$

We also introduce

$$\begin{array}{rclrcl} \text{first x}{\cdot}\alpha & = & \text{x} & \text{last } \beta{\cdot}\text{x} & = & \text{x} \\ \text{rest x}{\cdot}\alpha & = & \alpha & \text{prev } \beta{\cdot}\text{x} & = & \beta \end{array}$$

—

$$RI_{\mathsf{busy}} \equiv 0 \leq \mathsf{b} \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} \wedge \underline{\alpha = \beta{\cdot}\gamma \wedge \#\gamma = \mathsf{b}} \wedge \odot_{\mathsf{k}=0}^{\mathsf{b}-1}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto \underline{\gamma_{\mathsf{k}}}.$$

Next, we modify the critical regions for busy. We have

$$\begin{array}{l}
\{((\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \underline{\mathsf{consumed}\ \beta}) \ * \ RI_{\mathsf{busy}}) \wedge \mathsf{b} > 0\} \\
\{((\mathsf{j} \mapsto \underline{\mathsf{last}(\beta{\cdot}\mathsf{first}\ \gamma)}) \wedge \mathsf{j} = \mathsf{j}_b \wedge \underline{\mathsf{consumed}\ \beta}) \ * \\
\quad (0 \leq \mathsf{b} - 1 \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b \oplus 1 + \mathsf{b} - 1 \wedge \\
\qquad \underline{\alpha = \beta{\cdot}(\mathsf{first}\ \gamma){\cdot}(\mathsf{rest}\ \gamma) \wedge \#\mathsf{rest}\ \gamma = \mathsf{b} - 1} \wedge \\
\qquad \odot_{\mathsf{k}=1}^{\mathsf{b}-1}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto \underline{(\mathsf{rest}\ \gamma)_{\mathsf{k}-1}}\} \\
\mathsf{b} := \mathsf{b} - 1 \ ; \mathsf{j}_b := \mathsf{j}_b \oplus 1 \ ; \underline{\beta := \beta{\cdot}\mathsf{first}\ \gamma \ ; \gamma := \mathsf{rest}\ \gamma} \\
\{(\underline{\#\beta > 0} \wedge (\mathsf{j} \mapsto \underline{\mathsf{last}\ \beta}) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \underline{\mathsf{consumed}(\mathsf{prev}\ \beta)}) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad * \ RI_{\mathsf{busy}}\},
\end{array}$$

so that

$$\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \underline{\mathsf{consumed}\ \beta}\} \\
\mathsf{getbusy} & \equiv & \mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathsf{b} > 0\ \mathbf{do} \\
 & & \quad (\mathsf{b} := \mathsf{b} - 1 \ ; \mathsf{j}_b := \mathsf{j}_b \oplus 1 \ ; \\
 & & \quad \underline{\beta := \beta{\cdot}\mathsf{first}\ \gamma \ ; \gamma := \mathsf{rest}\ \gamma}) \\
 & & \{\underline{\#\beta > 0} \wedge (\mathsf{j} \mapsto \underline{\mathsf{last}\ \beta}) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \\
 & & \qquad\qquad \wedge \underline{\mathsf{consumed}(\mathsf{prev}\ \beta)}\}.
\end{array}$$

—

$$RI_{\mathsf{busy}} \equiv 0 \leq \mathsf{b} \leq \mathsf{n} \wedge \mathsf{i}_b =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} \wedge \underline{\alpha = \beta{\cdot}\gamma \wedge \#\gamma = \mathsf{b}} \wedge \odot_{\mathsf{k}=0}^{\mathsf{b}-1}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto \underline{\gamma_{\mathsf{k}}}.$$

Also,

$$\begin{array}{l}
\{((\mathsf{i} \mapsto \underline{\mathsf{x}}) \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}\ \alpha{\cdot}\mathsf{x}}) \ * \ RI_{\mathsf{busy}}\} \\
\{(\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}\ \alpha{\cdot}\mathsf{x}}) \ * \\
\quad (0 \leq \mathsf{b}+1 \leq \mathsf{n} \wedge \mathsf{i}_b \oplus 1 =_{\mathsf{n}} \mathsf{j}_b + \mathsf{b} + 1 \wedge \\
\qquad \underline{\alpha{\cdot}\mathsf{x} = \beta{\cdot}\gamma{\cdot}\mathsf{x} \wedge \#\gamma{\cdot}\mathsf{x} = \mathsf{b}+1} \wedge \odot_{\mathsf{k}=0}^{\mathsf{b}}(\mathsf{j}_b \oplus \mathsf{k}) \mapsto \underline{(\gamma{\cdot}\mathsf{x})_{\mathsf{k}}})\} \\
\mathsf{b} := \mathsf{b}+1 \ ; \ \mathsf{i}_b := \mathsf{i}_b \oplus 1 \ ; \ \underline{\alpha := \alpha{\cdot}\mathsf{x} \ ; \ \gamma := \gamma{\cdot}\mathsf{x}} \\
\{(\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b \wedge \underline{\mathsf{produced}\ \alpha}) \ * \ RI_{\mathsf{busy}}\},
\end{array}$$

so that

$$\begin{array}{lll}
 & & \{(\mathsf{i} \mapsto \underline{\mathsf{x}}) \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}\ \alpha{\cdot}\mathsf{x}}\} \\
\mathsf{putbusy} & \equiv & \mathbf{with}\ \mathsf{busy}\ \mathbf{when\ true\ do} \\
 & & \quad (\mathsf{b} := \mathsf{b}+1 \ ; \ \mathsf{i}_b := \mathsf{i}_b \oplus 1 \ ; \ \underline{\alpha := \alpha{\cdot}\mathsf{x} \ ; \ \gamma := \gamma{\cdot}\mathsf{x}}) \\
 & & \{\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b \wedge \underline{\mathsf{produced}\ \alpha}\}.
\end{array}$$

—

The first process becomes

$$
\begin{array}{rcl}
 & & \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha}\} \\
\mathsf{p}_1 & \equiv & \mathbf{while}\ \underline{\mathsf{k}_1 < \mathsf{N}}\ \mathbf{do} \\
 & & \big(\{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha \wedge \mathsf{k}_1 < \mathsf{N}}\} \\
 & & \quad \mathsf{Produce\ x}\ ;\ \underline{\mathsf{k}_1 := \mathsf{k}_1 + 1}\ ; \\
 & & \quad \mathsf{getfree}\ ; \\
 & & \quad \{(\mathsf{i} \mapsto -) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha \cdot \mathsf{x}}\} \\
 & & \quad [\mathsf{i}] := \mathsf{x}\ ; \\
 & & \quad \{(\mathsf{i} \mapsto \underline{\mathsf{x}}) \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha \cdot \mathsf{x}}\} \\
 & & \quad \mathsf{putbusy}\ ; \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{i} \oplus 1 = \mathsf{i}_f \wedge \mathsf{i} \oplus 1 = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha}\} \\
 & & \quad \mathsf{i} := \mathsf{i} \oplus 1 \\
 & & \quad \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha}\}\big) \\
 & & \{\mathbf{emp} \wedge \mathsf{i} = \mathsf{i}_f \wedge \mathsf{i} = \mathsf{i}_b \wedge \underline{\mathsf{produced}'\ \alpha \wedge \mathsf{k}_1 = \mathsf{N}}\},
\end{array}
$$

Here the variable k_1 keeps track of the length of α, and

$$\mathsf{produced}'\ \alpha = \mathsf{produced}\ \alpha \wedge \#\alpha = \mathsf{k}_1 \wedge \mathsf{k}_1 \leq \mathsf{N}.$$

The $\mathbf{while}$ command terminates when k_1 reaches N.

—

The second process becomes

$$
\begin{array}{lll}
 & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \underline{\mathsf{consumed}'\ \beta}\} & \\
\mathsf{p}_2 & \equiv \ \mathbf{while}\ \underline{\mathsf{k}_2 < \mathsf{N}}\ \mathbf{do} & \\
 & \quad \big(\{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \underline{\mathsf{consumed}'\ \beta \wedge \mathsf{k}_2 < \mathsf{N}}\} & \\
 & \quad \mathsf{getbusy}\ ; & \\
 & \quad \{\underline{\#\beta > 0} \wedge (\mathsf{j} \mapsto \underline{\mathsf{last}\ \beta}) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f & \\
 & \qquad\qquad \wedge\ \underline{\mathsf{consumed}'(\mathsf{prev}\ \beta) \wedge \mathsf{k}_2 < \mathsf{N}}\} & \\
 & \quad \mathsf{y} := [\mathsf{j}]\ ; & \\
 & \quad \{(\mathsf{j} \mapsto -) \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f & \\
 & \quad \wedge\ \underline{\#\beta > 0 \wedge \mathsf{y} = \mathsf{last}\ \beta \wedge \mathsf{consumed}'(\mathsf{prev}\ \beta) \wedge \mathsf{k}_2 < \mathsf{N}}\} & \\
 & \quad \mathsf{putfree}\ ; & \\
 & \quad \{\mathbf{emp} \wedge \mathsf{j} \oplus 1 = \mathsf{j}_b \wedge \mathsf{j} \oplus 1 = \mathsf{j}_f & \\
 & \quad \wedge\ \underline{\#\beta > 0 \wedge \mathsf{y} = \mathsf{last}\ \beta \wedge \mathsf{consumed}'(\mathsf{prev}\ \beta) \wedge \mathsf{k}_2 < \mathsf{N}}\} & \\
 & \quad \mathsf{j} := \mathsf{j} \oplus 1\ ; & \\
 & \quad \mathsf{Consume\ y}\ ;\ \underline{\mathsf{k}_2 := \mathsf{k}_2 + 1} & \\
 & \quad \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f & \\
 & \qquad\qquad \wedge\ \underline{\#\beta > 0 \wedge \mathsf{y} = \mathsf{last}\ \beta \wedge \mathsf{consumed}'((\mathsf{prev}\ \beta)\cdot\mathsf{y})}\} & \\
 & \quad \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \underline{\mathsf{consumed}'\ \beta}\}\big) & \\
 & \{\mathbf{emp} \wedge \mathsf{j} = \mathsf{j}_b \wedge \mathsf{j} = \mathsf{j}_f \wedge \underline{\mathsf{consumed}'\ \beta \wedge \mathsf{k}_2 = \mathsf{N}}\}. &
\end{array}
$$

Here the variable k_2 keeps track of the length of β, and

$$\mathsf{consumed}'\ \beta = \mathsf{consumed}\ \beta \wedge \#\beta = \mathsf{k}_2 \wedge \mathsf{k}_2 \leq \mathsf{N}.$$

The $\mathbf{while}$ command terminates when k_2 reaches N.

—

Finally, we have

$\{\bigodot_{k=0}^{n-1}(l + k) \mapsto - \wedge \underline{\mathsf{produced}\ \epsilon \wedge \mathsf{consumed}\ \epsilon}\}$

$f := n\ ;\ b := 0\ ;\ \underline{\alpha := \epsilon\ ;\ \beta := \epsilon\ ;\ \gamma := \epsilon\ ;\ k_1 := 0\ ;\ k_2 := 0}\ ;$

$i := l\ ;\ i_f := l\ ;\ i_b := l\ ;\ j := l\ ;\ j_f := l\ ;\ j_b := l\ ;$

$\{RI_{\mathsf{free}} * RI_{\mathsf{busy}} * (\mathbf{emp} \wedge i = i_f \wedge i = i_b \wedge \underline{\mathsf{produced}'\ \alpha}) *$

$\quad (\mathbf{emp} \wedge j = j_b \wedge j = j_f \wedge \underline{\mathsf{consumed}'\ \beta})\}$

$\mathbf{resource}\ \mathsf{free}(f, i_f, j_f), \mathsf{busy}(b, j_b, i_b, \alpha, \beta, \gamma)\ \mathbf{in}\ (p_1 \parallel p_2)$

$\{RI_{\mathsf{free}} * RI_{\mathsf{busy}} *$

$\quad (\mathbf{emp} \wedge i = i_f \wedge i = i_b \wedge \underline{\mathsf{produced}'\ \alpha \wedge k_1 = N}) *$

$\quad (\mathbf{emp} \wedge j = j_b \wedge j = j_f \wedge \underline{\mathsf{consumed}'\ \beta \wedge k_2 = N})\}$

$\{\underline{\alpha = \beta{\cdot}\gamma \wedge \#\alpha = \#\beta = N \wedge \mathsf{produced}\ \alpha \wedge \mathsf{consumed}\ \beta}\}$

$\{\underline{\alpha = \beta \wedge \gamma = \epsilon \wedge \mathsf{produced}\ \alpha \wedge \mathsf{consumed}\ \beta}\}$

—

Deleting assertions and putting everything together, we get

$$
\begin{array}{l}
\mathsf{f} := \mathsf{n}\ ;\ \mathsf{b} := 0\ ;\ \underline{\alpha := \epsilon\ ;\ \beta := \epsilon\ ;\ \gamma := \epsilon\ ;\ \mathsf{k}_1 := 0\ ;\ \mathsf{k}_2 := 0}\ ; \\
\mathsf{i} := \mathsf{I}\ ;\ \mathsf{i}_f := \mathsf{I}\ ;\ \mathsf{i}_b := \mathsf{I}\ ;\ \mathsf{j} := \mathsf{I}\ ;\ \mathsf{j}_f := \mathsf{I}\ ;\ \mathsf{j}_b := \mathsf{I}\ ; \\
\mathbf{resource}\ \mathsf{free}(\mathsf{f}, \mathsf{i}_f, \mathsf{j}_f), \mathsf{busy}(\mathsf{b}, \mathsf{j}_b, \mathsf{i}_b, \underline{\alpha, \beta, \gamma})\ \mathbf{in} \\
\quad \big(\mathbf{while}\ \underline{\mathsf{k}_1 < \mathsf{N}}\ \mathbf{do} \\
\qquad \big(\mathsf{Produce}\ \mathsf{x}\ ;\ \underline{\mathsf{k}_1 := \mathsf{k}_1 + 1}\ ; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when}\ \mathsf{f} > 0\ \mathbf{do} \\
\qquad\quad (\mathsf{f} := \mathsf{f} - 1\ ;\ \mathsf{i}_f := \mathsf{i}_f \oplus 1)\ ; \\
\qquad [\mathsf{i}] := \mathsf{x}\ ; \\
\qquad \mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathbf{true}\ \mathbf{do} \\
\qquad\quad (\mathsf{b} := \mathsf{b} + 1\ ;\ \mathsf{i}_b := \mathsf{i}_b \oplus 1\ ;\ \underline{\alpha := \alpha \cdot \mathsf{x}\ ;\ \gamma := \gamma \cdot \mathsf{x}})\ ; \\
\qquad \mathsf{i} := \mathsf{i} \oplus 1\big) \\
\quad \| \ \mathbf{while}\ \underline{\mathsf{k}_2 < \mathsf{N}}\ \mathbf{do} \\
\qquad \big(\mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathsf{b} > 0\ \mathbf{do} \\
\qquad\quad (\mathsf{b} := \mathsf{b} - 1\ ;\ \mathsf{j}_b := \mathsf{j}_b \oplus 1\ ;\ \underline{\beta := \beta \cdot \mathsf{first}(\gamma)\ ;\ \gamma := \mathsf{rest}(\gamma)})\ ; \\
\qquad \mathsf{y} := [\mathsf{j}]\ ; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when}\ \mathbf{true}\ \mathbf{do} \\
\qquad\quad (\mathsf{f} := \mathsf{f} + 1\ ;\ \mathsf{j}_f := \mathsf{j}_f \oplus 1)\ ; \\
\qquad \mathsf{j} := \mathsf{j} \oplus 1\ ; \\
\qquad \mathsf{Consume}\ \mathsf{y}\ ;\ \underline{\mathsf{k}_2 := \mathsf{k}_2 + 1}\big)\big)
\end{array}
$$

—

In addition to $\mathsf{i}_f, \mathsf{i}_b, \mathsf{j}_f, \mathsf{j}_b$, the variables α, β, and γ are auxiliary. Once these auxiliary variables are removed, we have the program

$$
\begin{array}{l}
\mathsf{f} := \mathsf{n} \,;\, \mathsf{b} := 0 \,;\, \mathsf{i} := \mathsf{l} \,;\, \mathsf{j} := \mathsf{l} \,;\, \mathsf{k}_1 := 0 \,;\, \mathsf{k}_2 := 0 \,; \\
\mathbf{resource}\ \mathsf{free}(\mathsf{f}), \mathsf{busy}(\mathsf{b})\ \mathbf{in} \\
\quad (\mathbf{while}\ \mathsf{k}_1 < \mathsf{N}\ \mathbf{do} \\
\qquad (\mathsf{Produce}\ \mathsf{x} \,;\, \mathsf{k}_1 := \mathsf{k} + 1 \,; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when}\ \mathsf{f} > 0\ \mathbf{do}\ \mathsf{f} := \mathsf{f} - 1 \,; \\
\qquad [\mathsf{i}] := \mathsf{x} \,; \\
\qquad \mathbf{with}\ \mathsf{busy}\ \mathbf{when\ true\ do}\ \mathsf{b} := \mathsf{b} + 1 \,; \\
\qquad \mathsf{i} := \mathsf{i} \oplus 1) \\
\quad \| \ \mathbf{while}\ \mathsf{k}_2 < \mathsf{N}\ \mathbf{do} \\
\qquad (\mathbf{with}\ \mathsf{busy}\ \mathbf{when}\ \mathsf{b} > 0\ \mathbf{do}\ \mathsf{b} := \mathsf{b} - 1 \,; \\
\qquad \mathsf{y} := [\mathsf{j}] \,; \\
\qquad \mathbf{with}\ \mathsf{free}\ \mathbf{when\ true\ do}\ \mathsf{f} := \mathsf{f} + 1 \,; \\
\qquad \mathsf{j} := \mathsf{j} \oplus 1 \,; \\
\qquad \mathsf{Consume}\ \mathsf{y} \,;\, \mathsf{k}_2 := \mathsf{k}_2 + 1))
\end{array}
$$

—

Further Comments

- free and busy are super counting semaphores. In the case where $n = 1$ our buffer program reduces to the O'Hearn's split binary semaphore example (taking I to be 10).
- Variable Permissions

	f	b	i	j	i_f	j_f	i_b	j_b	γ	α	β	k_1	k_2
free	1				$\frac{1}{2}$	$\frac{1}{2}$							
busy		1					$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$		
p_1			1		$\frac{1}{2}$		$\frac{1}{2}$			$\frac{1}{2}$		1	
p_2				1		$\frac{1}{2}$		$\frac{1}{2}$			$\frac{1}{2}$		1

- Heap Permissions (All 1)

tree	i_f to $j_f - 1$	
busy	j_b to $i_b - 1$	
p_1	i_b to $i_f - 1$	(0 or 1 elements)
p_2	j_f to $j_b - 1$	(0 or 1 elements)

—