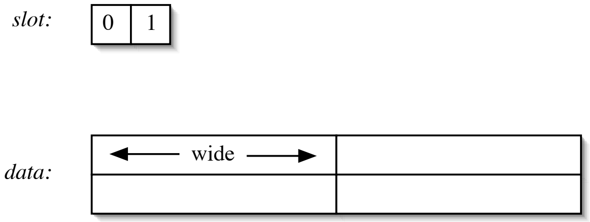
Simpson's 4-slot algorithm, proved in three slides

Richard Bornat
School of Computing, Middlesex University
(and Matthew Parkinson, ditto)

20th December 2005

Data structures: a bit array and a wide data array



Programming is really hard: only nine lines, no CAS, and you still can't understand it

Programming is really hard: only nine lines, no CAS, and you still can't understand it

```
var    reading, latest : bit
       slot : array bit of bit
       data : array bit of array bit of datatype

procedure  write    (item : datatype);
           var      pair, index : bit;
           begin
                    pair := not(reading);
                    index := not(slot[pair]);
                    data[pair, index] := item;
                    slot[pair] := index;
                    latest := pair
           end;

procedure  read :   datatype;
           var      pair, index : bit;
           begin
                    pair := latest;
                    reading := pair;
                    index := slot[pair];
                    read := data[pair, index]
           end;
```

Separation logic

Separation logic

- $E \mapsto F$ is a single-celled heap with address E and content F.

Separation logic

- $E \mapsto F$ is a single-celled heap with address E and content F.
- $E \mapsto F_0, F_1$ is a two-celled heap; $E \mapsto F_0, F_1, F_2$ is three cells; and so on for four-, five-, ... celled heaps.

Separation logic

- $E \mapsto F$ is a single-celled heap with address E and content F.
- $E \mapsto F_0, F_1$ is a two-celled heap; $E \mapsto F_0, F_1, F_2$ is three cells; and so on for four-, five-, ... celled heaps.
- E and F must be 'pure' expressions that don't mention the heap (don't use $\mapsto$).

Separation logic

- $E \mapsto F$ is a single-celled heap with address E and content F.
- $E \mapsto F_0, F_1$ is a two-celled heap; $E \mapsto F_0, F_1, F_2$ is three cells; and so on for four-, five-, ... celled heaps.
- E and F must be 'pure' expressions that don't mention the heap (don't use $\mapsto$).
- $A \star B$ is separation of heaps; $A \wedge B$, $A \vee B$, $\neg A$, $A \rightarrow B$, $\forall x \cdot P(x)$, $\exists x \cdot P(x)$ are as normal. $A \wedge B$ expresses coincidence of heaps; you don't need to know about $A \mathrel{-\!\!\star} B$.

Separation logic

- $E \mapsto F$ is a single-celled heap with address E and content F.
- $E \mapsto F_0, F_1$ is a two-celled heap; $E \mapsto F_0, F_1, F_2$ is three cells; and so on for four-, five-, ... celled heaps.
- E and F must be 'pure' expressions that don't mention the heap (don't use $\mapsto$).
- $A \star B$ is separation of heaps; $A \wedge B$, $A \vee B$, $\neg A$, $A \longrightarrow B$, $\forall x \cdot P(x)$, $\exists x \cdot P(x)$ are as normal. $A \wedge B$ expresses coincidence of heaps; you don't need to know about $A \mathrel{-\!\!\star} B$.
- $E \mapsto F_0, F_1$ is just shorthand for $E \mapsto F_0 \star E + 1 \mapsto F_1$.

A modified Hoare logic

A modified Hoare logic

- $\{Q\}\ C\ \{R\}$ is a resourced and partial correctness assertion. C will not go wrong (exceed its allocated resources) if started with resource Q, and will, if it terminates, deliver resource R.

A modified Hoare logic

- $\{Q\}\, C\, \{R\}$ is a resourced and partial correctness assertion. C will not go wrong (exceed its allocated resources) if started with resource Q, and will, if it terminates, deliver resource R.
- The 'small axioms' of assignment are

$$\{\mathbf{emp}\}\, x := \mathrm{new}()\, \{x \mapsto \_\}$$
$$\{E \mapsto \_\}\ \mathrm{dispose}\, E\, \{\mathbf{emp}\}$$
$$\{R[E/x]\}\, x := E\, \{R\} \quad (\text{the Hoare axiom})$$
$$\{E \mapsto F\}\, x := [E]\, \{x = F \wedge E \mapsto F\} \quad (x \text{ not free in } E, F)$$
$$\{E \mapsto \_\}\, [E] := F\, \{E \mapsto F\}$$

Three inference rules

Three inference rules

- The frame rule: $\dfrac{\{Q\}\; C\; \{R\}}{\{P \star Q\}\; C\; \{P \star R\}}$ (modifies $C \bigcap$ free $P = \{\}$)

Three inference rules

- The frame rule: $\dfrac{\{Q\}\, C\, \{R\}}{\{P \star Q\}\, C\, \{P \star R\}}$ (modifies $C \bigcap$ free $P = \{\}$)

- The concurrency rule (has horrid side-condition):

$$\frac{\{Q_1\}\, C_1\, \{R_1\} \quad \{Q_2\}\, C_2\, \{R_2\} \quad \ldots \quad \{Q_n\}\, C_n\, \{R_n\}}{\{Q_1 \star Q_2 \star \cdots \star Q_n\}\, C_1 \mid\mid C_2 \mid\mid \cdots \mid\mid C_n\, \{R_1 \star R_2 \star \cdots \star R_n\}}$$

Three inference rules

- The frame rule: $\dfrac{\{Q\}\ C\ \{R\}}{\{P \star Q\}\ C\ \{P \star R\}}$ (modifies $C \bigcap$ free $P = \{\}$)

- The concurrency rule (has horrid side-condition):

$$\frac{\{Q_1\}\ C_1\ \{R_1\} \quad \{Q_2\}\ C_2\ \{R_2\} \quad \ldots \quad \{Q_n\}\ C_n\ \{R_n\}}{\{Q_1 \star Q_2 \star \cdots \star Q_n\}\ C_1 \mid\mid C_2 \mid\mid \cdots \mid\mid C_n\ \{R_1 \star R_2 \star \cdots \star R_n\}}$$

- The CCR rule (has *atrocious* side condition):

$$\frac{\{(Q \star I_b) \wedge G\}\ C\ \{R \star I_b\}}{\{Q\}\ \text{with}\ b\ \text{when}\ G\ \text{do}\ C\ \text{od}\ \{R\}}$$

Recent simplifications (not explained here)

Recent simplifications (not explained here)

- Permissions (fractions of $\longmapsto$, counts of $\rightarrowtail$) to allow sharing of heap;

Recent simplifications (not explained here)

- Permissions (fractions of $\mapsto$, counts of $\rightarrowtail$) to allow sharing of heap;
- Variable permissions, to allow variables to be resource;

Recent simplifications (not explained here)

- Permissions (fractions of $\mapsto$, counts of $\rightarrowtail$) to allow sharing of heap;
- Variable permissions, to allow variables to be resource;
- Trivial side conditions;

Recent simplifications (not explained here)

- Permissions (fractions of $\mapsto$, counts of $\rightarrowtail$) to allow sharing of heap;
- Variable permissions, to allow variables to be resource;
- Trivial side conditions;
- No side conditions at all (very new, this!).

Nine lines are now ten, *with added auxiliary proof-variables*

write: with $bundle$ when true do $pair :=$ not$(reading)$; $wuse := pair$ od;
$index :=$ not$(slot[pair])$;
$data[pair, index] := item$;
with $bundle$ when true do $slot[pair] := index$; $wuse := -1$ od;
with $bundle$ when true do $latest := pair$ od

read: with $bundle$ when true do $pair := latest$ od;
with $bundle$ when true do $reading := pair$ od;
with $bundle$ when true do $index := slot[pair]$; $ruse := index$ od;
$read := data[pair, index]$;
with $bundle$ when true do $ruse := -1$ od

What the writer owns

(A point of notation: I've used a special form of $\mapsto$ to describe a heap, just to make the slides easy to read.

For example, $data[pair, index] \mapsto \_$ replaces $data + 2 \star pair + index \mapsto \_$.

There is no change in meaning.)

What the writer owns

(A point of notation: I've used a special form of $\mapsto$ to describe a heap, just to make the slides easy to read.

For example, $data[pair, index] \mapsto \_$ replaces $data + 2 \star pair + index \mapsto \_$.

There is no change in meaning.)

$$latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index$$

$$\models \left(\begin{array}{l} slot[0] \xmapsto[0.5]{} \_ \star slot[1] \xmapsto[0.5]{} \_ \star \\ \text{if } wuse \geq 0 \text{ then } data[pair, index] \mapsto \_ \text{ else } \mathbf{emp} \text{ fi} \end{array} \right)$$

What the reader owns

$$
\begin{aligned}
&reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \\
&\quad \models \text{if } ruse \geq 0 \text{ then } data[pair, index] \mapsto \_ \text{ else } \mathbf{emp} \text{ fi}
\end{aligned}
$$

The bundle owns the rest

$$latest_{0.5},\ reading_{0.5},\ slot_{0.5},\ data_{0.33},\ wuse_{0.5},\ ruse_{0.5}$$

$$\models \exists s \cdot \left(\begin{array}{l}
slot[0] \xmapsto[0.5]{} s(0) \star slot[1] \xmapsto[0.5]{} s(1) \star \\
\text{if } wuse \geq 0 \wedge ruse \geq 0 \text{ then} \\
\quad data[reading, \text{not}(ruse)] \mapsto _ \star data[wuse, s(wuse)] \mapsto _ \\
\text{elsf } wuse \geq 0 \text{ then} \\
\quad data[wuse, s(wuse)] \mapsto _ \star \\
\quad data[\text{not}(wuse), s(\text{not}(wuse))] \mapsto _ \star data[\text{not}(wuse), \text{not}(s(\text{not}(wuse)))] \mapsto _ \\
\text{elsf } ruse \geq 0 \text{ then} \\
\quad data[reading, \text{not}(ruse)] \mapsto _ \star \\
\quad data[\text{not}(reading), s(\text{not}(reading))] \mapsto _ \star data[\text{not}(reading), \text{not}(s(\text{not}(reading)))] \mapsto _) \\
\text{else} \\
\quad data \mapsto _, _, _, _ \\
\text{fi}
\end{array} \right)$$

The writer

$$\left\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \xmapsto[0.5]{} \_ \star slot[1] \xmapsto[0.5]{} \_\ \right\}$$

with $bundle$ when true do $pair :=$ not$(reading)$; $wuse := pair$ od;

$index :=$ not$(slot[pair])$;

$data[pair, index] := item$;

with $bundle$ when true do $slot[pair] := index$; $wuse := -1$ od;

with $bundle$ when true do $latest := pair$ od

$$\left\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \xmapsto[0.5]{} \_ \star slot[1] \xmapsto[0.5]{} \_\ \right\}$$

The writer

$$\left\{ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \right\}$$

with $bundle$ when true do $pair := \mathrm{not}(reading)$; $wuse := pair$ od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \exists i \cdot \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} i \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, \mathrm{not}(i)] \mapsto \_ \end{array} \right) \end{array} \right\}$$

$index := \mathrm{not}(slot[pair])$;

$data[pair, index] := item$;

with $bundle$ when true do $slot[pair] := index$; $wuse := -1$ od;

with $bundle$ when true do $latest := pair$ od

$$\left\{ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \right\}$$

The writer

$$\left\{ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \xmapsto[0.5]{} \_ \star slot[1] \xmapsto[0.5]{} \_ \right\}$$

with *bundle* when true do *pair* := not(*reading*); *wuse* := *pair* od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \exists i \cdot \left(\begin{array}{l} slot[pair] \xmapsto[0.5]{} i \star slot[\mathrm{not}(pair)] \xmapsto[0.5]{} \_ \star \\ data[pair, \mathrm{not}(i)] \mapsto \_ \end{array} \right) \end{array} \right\}$$

$index := \mathrm{not}(slot[pair]);$

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \left(\begin{array}{l} slot[pair] \xmapsto[0.5]{} \mathrm{not}(index) \star slot[\mathrm{not}(pair)] \xmapsto[0.5]{} \_ \star \\ data[pair, index] \mapsto \_ \end{array} \right) \end{array} \right\}$$

$data[pair, index] := item;$

with *bundle* when true do *slot*[*pair*] := *index*; *wuse* := −1 od;

with *bundle* when true do *latest* := *pair* od

$$\left\{ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \xmapsto[0.5]{} \_ \star slot[1] \xmapsto[0.5]{} \_ \right\}$$

The writer

$$\left\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \ \right\}$$

with $bundle$ when true do $pair := \mathrm{not}(reading)$; $wuse := pair$ od;

$$\left\{\begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \exists i \cdot \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} i \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, \mathrm{not}(i)] \mapsto \_ \end{array}\right) \end{array}\right\}$$

$index := \mathrm{not}(slot[pair])$;

$$\left\{\begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} \mathrm{not}(index) \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, index] \mapsto \_ \end{array}\right) \end{array}\right\}$$

$data[pair, index] := item$;

$$\left\{\begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} \mathrm{not}(index) \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, index] \mapsto item \end{array}\right) \end{array}\right\}$$

with $bundle$ when true do $slot[pair] := index$; $wuse := -1$ od;

with $bundle$ when true do $latest := pair$ od

$$\left\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \ \right\}$$

The writer

$$\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_\ \}$$

with $bundle$ when true do $pair := \mathrm{not}(reading)$; $wuse := pair$ od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \exists i \cdot \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} i \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, \mathrm{not}(i)] \mapsto \_ \end{array} \right) \end{array} \right\}$$

$index := \mathrm{not}(slot[pair])$;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} \mathrm{not}(index) \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, index] \mapsto \_ \end{array} \right) \end{array} \right\}$$

$data[pair, index] := item$;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \left(\begin{array}{l} slot[pair] \underset{0.5}{\longmapsto} \mathrm{not}(index) \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star \\ data[pair, index] \mapsto item \end{array} \right) \end{array} \right\}$$

with $bundle$ when true do $slot[pair] := index$; $wuse := -1$ od;

$$\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_\ \}$$

with $bundle$ when true do $latest := pair$ od

$$\{\ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_\ \}$$

Details of the first writer step

$\left\{ \; latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \vDash wuse = -1 \wedge slot[0] \xrightarrow[0.5]{} \_ \star slot[1] \xrightarrow[0.5]{} \_ \; \right\}$

with *bundle* when true do

$pair := \mathrm{not}(reading);$

$wuse := pair$

od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \vDash wuse = pair \wedge \exists i \cdot \left(slot[pair] \xrightarrow[0.5]{} i \star slot[\mathrm{not}(pair)] \xrightarrow[0.5]{} \_ \star data[pair, \mathrm{not}(i)] \mapsto \_ \right) \end{array} \right\}$$

Details of the first writer step

$$\left\{\ \mathit{latest}_{0.5}, \mathit{slot}_{0.5}, \mathit{data}_{0.33}, \mathit{wuse}_{0.5}, \mathit{pair}, \mathit{index} \vDash \mathit{wuse} = -1 \wedge \mathit{slot}[0] \xrightarrow[0.5]{} \_ \star \mathit{slot}[1] \xrightarrow[0.5]{} \_\ \right\}$$

with *bundle* when true do

$$\left\{\begin{array}{l} \mathit{latest}, \mathit{reading}_{0.5}, \mathit{slot}, \mathit{data}_{0.66}, \mathit{wuse}, \mathit{pair}, \mathit{index} \\ \vDash \exists s \cdot \left(\begin{array}{l} \mathit{wuse} = -1 \wedge \mathit{slot} \mapsto s(0), s(1) \star \\ \mathit{data}[\mathrm{not}(\mathit{reading}), s(\mathrm{not}(\mathit{reading}))] \mapsto \_ \star \mathit{data}[\mathrm{not}(\mathit{reading}), \mathrm{not}(s(\mathrm{not}(\mathit{reading})))] \mapsto \_ \star \\ \ \mathrm{if}\ \mathit{ruse} \geq 0 \quad \mathrm{then}\ \mathit{data}[\mathit{reading}, \mathrm{not}(\mathit{ruse})] \mapsto \_ \\ \qquad\qquad\qquad \mathrm{else}\ \mathit{data}[\mathit{reading}, s(\mathit{reading})] \mapsto \_ \star \mathit{data}[\mathit{reading}, \mathrm{not}(s(\mathit{reading}))] \mapsto \_ \\ \ \mathrm{fi} \end{array}\right) \end{array}\right\}$$

$\mathit{pair} := \mathrm{not}(\mathit{reading});$

$\mathit{wuse} := \mathit{pair}$

od;

$$\left\{\begin{array}{l} \mathit{latest}_{0.5}, \mathit{slot}_{0.5}, \mathit{data}_{0.33}, \mathit{wuse}_{0.5}, \mathit{pair}, \mathit{index} \\ \quad \vDash \mathit{wuse} = \mathit{pair} \wedge \exists i \cdot \left(\ \mathit{slot}[\mathit{pair}] \xrightarrow[0.5]{} i \star \mathit{slot}[\mathrm{not}(\mathit{pair})] \xrightarrow[0.5]{} \_ \star \mathit{data}[\mathit{pair}, \mathrm{not}(i)] \mapsto \_\ \right) \end{array}\right\}$$

Details of the first writer step

$$\left\{ latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \land slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \right\}$$

with *bundle* when true do

$$\left\{ \begin{array}{l} latest, reading_{0.5}, slot, data_{0.66}, wuse, pair, index \\ \models \exists s \cdot \left(\begin{array}{l} wuse = -1 \land slot \mapsto s(0), s(1) \star \\ data[\mathsf{not}(reading), s(\mathsf{not}(reading))] \mapsto \_ \star data[\mathsf{not}(reading), \mathsf{not}(s(\mathsf{not}(reading)))] \mapsto \_ \star \\ \text{ if } ruse \geq 0 \text{ then } data[reading, \mathsf{not}(ruse)] \mapsto \_ \\ \qquad \text{else } data[reading, s(reading)] \mapsto \_ \star data[reading, \mathsf{not}(s(reading))] \mapsto \_ \\ \text{ fi} \end{array} \right) \end{array} \right\}$$

$pair := \mathsf{not}(reading);$

$$\left\{ \begin{array}{l} latest, reading_{0.5}, slot, data_{0.66}, wuse, pair, index \\ \models \exists s \cdot \left(\begin{array}{l} wuse = -1 \land pair = \mathsf{not}(reading) \land slot \mapsto s(0), s(1) \star \\ data[\mathsf{not}(reading), s(\mathsf{not}(reading))] \mapsto \_ \star data[\mathsf{not}(reading), \mathsf{not}(s(\mathsf{not}(reading)))] \mapsto \_ \star \\ \text{ if } ruse \geq 0 \text{ then } data[reading, \mathsf{not}(ruse)] \mapsto \_ \\ \qquad \text{else } data[reading, s(reading)] \mapsto \_ \star data[reading, \mathsf{not}(s(reading))] \mapsto \_ \\ \text{ fi} \end{array} \right) \end{array} \right\}$$

$wuse := pair$

od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \models wuse = pair \land \exists i \cdot \left(slot[pair] \underset{0.5}{\longmapsto} i \star slot[\mathsf{not}(pair)] \underset{0.5}{\longmapsto} \_ \star data[pair, \mathsf{not}(i)] \mapsto \_ \right) \end{array} \right\}$$

Details of the first writer step

$$\left\{ \; latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \models wuse = -1 \wedge slot[0] \underset{0.5}{\longmapsto} \_ \star slot[1] \underset{0.5}{\longmapsto} \_ \; \right\}$$

with *bundle* when true do

$$\left\{ \begin{array}{l} latest, reading_{0.5}, slot, data_{0.66}, wuse, pair, index \\ \models \exists s \cdot \left(\begin{array}{l} wuse = -1 \wedge slot \mapsto s(0), s(1) \star \\ data[\mathrm{not}(reading), s(\mathrm{not}(reading))] \mapsto \_ \star data[\mathrm{not}(reading), \mathrm{not}(s(\mathrm{not}(reading)))] \mapsto \_ \star \\ \;\; \mathrm{if}\ ruse \geq 0 \;\; \mathrm{then}\ data[reading, \mathrm{not}(ruse)] \mapsto \_ \\ \qquad\qquad\qquad \mathrm{else}\ data[reading, s(reading)] \mapsto \_ \star data[reading, \mathrm{not}(s(reading))] \mapsto \_ \\ \;\; \mathrm{fi} \end{array} \right) \end{array} \right\}$$

$pair := \mathrm{not}(reading);$

$$\left\{ \begin{array}{l} latest, reading_{0.5}, slot, data_{0.66}, wuse, pair, index \\ \models \exists s \cdot \left(\begin{array}{l} wuse = -1 \wedge pair = \mathrm{not}(reading) \wedge slot \mapsto s(0), s(1) \star \\ data[\mathrm{not}(reading), s(\mathrm{not}(reading))] \mapsto \_ \star data[\mathrm{not}(reading), \mathrm{not}(s(\mathrm{not}(reading)))] \mapsto \_ \star \\ \;\; \mathrm{if}\ ruse \geq 0 \;\; \mathrm{then}\ data[reading, \mathrm{not}(ruse)] \mapsto \_ \\ \qquad\qquad\qquad \mathrm{else}\ data[reading, s(reading)] \mapsto \_ \star data[reading, \mathrm{not}(s(reading))] \mapsto \_ \\ \;\; \mathrm{fi} \end{array} \right) \end{array} \right\}$$

$wuse := pair$

$$\left\{ \begin{array}{l} latest, reading_{0.5}, slot, data_{0.66}, wuse, pair, index \\ \models \exists s \cdot \left(\begin{array}{l} wuse = pair \wedge pair = \mathrm{not}(reading) \wedge slot \mapsto s(0), s(1) \star \\ data[\mathrm{not}(reading), s(\mathrm{not}(reading))] \mapsto \_ \star data[\mathrm{not}(reading), \mathrm{not}(s(\mathrm{not}(reading)))] \mapsto \_ \star \\ \;\; \mathrm{if}\ ruse \geq 0 \;\; \mathrm{then}\ data[reading, \mathrm{not}(ruse)] \mapsto \_ \\ \qquad\qquad\qquad \mathrm{else}\ data[reading, s(reading)] \mapsto \_ \star data[reading, \mathrm{not}(s(reading))] \mapsto \_ \\ \;\; \mathrm{fi} \end{array} \right) \end{array} \right\}$$

od;

$$\left\{ \begin{array}{l} latest_{0.5}, slot_{0.5}, data_{0.33}, wuse_{0.5}, pair, index \\ \quad \models wuse = pair \wedge \exists i \cdot \left(slot[pair] \underset{0.5}{\longmapsto} i \star slot[\mathrm{not}(pair)] \underset{0.5}{\longmapsto} \_ \star data[pair, \mathrm{not}(i)] \mapsto \_ \right) \end{array} \right\}$$

The reader is even easier than the writer!

$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$

with *bundle* when true do *pair* := *latest* od;

with *bundle* when true do *reading* := *pair* od;

with *bundle* when true do *index* := *slot*[*pair*]; *ruse* := *index* od;

read := *data*[*pair*, *index*];

with *bundle* when true do *ruse* := −1 od

$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$

The reader is even easier than the writer!

$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with *bundle* when true do $pair := latest$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with *bundle* when true do $reading := pair$ od;

with *bundle* when true do $index := slot[pair]; ruse := index$ od;

$read := data[pair, index];$

with *bundle* when true do $ruse := -1$ od
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$

The reader is even easier than the writer!

$\{\ \mathit{reading}_{0.5}, \mathit{ruse}_{0.5}, \mathit{data}_{0.33}, \mathit{pair}, \mathit{index} \vDash \mathit{ruse} = -1\ \}$

$\quad$ with bundle when true do $\mathit{pair} := \mathit{latest}$ od;

$\{\ \mathit{reading}_{0.5}, \mathit{ruse}_{0.5}, \mathit{data}_{0.33}, \mathit{pair}, \mathit{index} \vDash \mathit{ruse} = -1\ \}$

$\quad$ with bundle when true do $\mathit{reading} := \mathit{pair}$ od;

$\{\ \mathit{reading}_{0.5}, \mathit{ruse}_{0.5}, \mathit{data}_{0.33}, \mathit{pair}, \mathit{index} \vDash \mathit{ruse} = -1 \wedge \mathit{reading} = \mathit{pair}\ \}$

$\quad$ with bundle when true do $\mathit{index} := \mathit{slot}[\mathit{pair}]; \mathit{ruse} := \mathit{index}$ od;

$\quad \mathit{read} := \mathit{data}[\mathit{pair}, \mathit{index}];$

$\quad$ with bundle when true do $\mathit{ruse} := -1$ od

$\{\ \mathit{reading}_{0.5}, \mathit{ruse}_{0.5}, \mathit{data}_{0.33}, \mathit{pair}, \mathit{index} \vDash \mathit{ruse} = -1\ \}$

The reader is even easier than the writer!

$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with *bundle* when true do $pair := latest$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with *bundle* when true do $reading := pair$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1 \wedge reading = pair\ \}$
with *bundle* when true do $index := slot[pair]$; $ruse := index$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse \geq 0 \wedge reading = pair \wedge data[pair, index] \mapsto \_\ \}$
$read := data[pair, index]$;

with *bundle* when true do $ruse := -1$ od
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$

The reader is even easier than the writer!

$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with $bundle$ when true do $pair := latest$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$
with $bundle$ when true do $reading := pair$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1 \wedge reading = pair\ \}$
with $bundle$ when true do $index := slot[pair]$; $ruse := index$ od;
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse \geq 0 \wedge reading = pair \wedge data[pair, index] \mapsto \_\ \}$
$read := data[pair, index]$;

$$\left\{ \begin{array}{l} reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \\ \quad \vDash ruse \geq 0 \wedge reading = pair \wedge \exists i \cdot data[pair, index] \mapsto i \wedge read = i \end{array} \right\}$$

with $bundle$ when true do $ruse := -1$ od
$\{\ reading_{0.5}, ruse_{0.5}, data_{0.33}, pair, index \vDash ruse = -1\ \}$

The rest of the reader is too easy to bother with

$$\text{with } \mathit{bundle} \text{ when true do } \mathit{index} := \mathit{slot}[\mathit{pair}]; \mathit{ruse} := \mathit{index}$$

(in the reader) is very very *very* similar to

$$\text{with } \mathit{bundle} \text{ when true do } \mathit{pair} := \text{not}(\mathit{reading}); \mathit{wuse} := \mathit{pair} \text{ od}$$

(which I just showed you in detail from the writer),

so you don’t need to see it.

The rest of the reader is too easy to bother with

with $bundle$ when true do $index := slot[pair]$; $ruse := index$

(in the reader) is very very *very* similar to

with $bundle$ when true do $pair := \text{not}(reading)$; $wuse := pair$ od

(which I just showed you in detail from the writer),

so you don’t need to see it.

And the rest of the reader is trivial.