Super-Theories

Technical Report

Pedro Ribeiro
Department of Computer Science
University of York, UK

March 2016

Contents

Chapter 1

Super-Theories

1.1 Revised Results with CI and TR

Lemma L.1.1.1 ***Isabelle proof available: Section 9.5***

$$\mathbf{CITR} \circ \mathbf{R1}(P) = \mathbf{CITR}(P)$$

Proof.

$\mathbf{CITR} \circ \mathbf{R1}(P)$ {Lemma L.3.6.2}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions **CIB** and **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R1} \circ \mathbf{CI4_m}(P)$
{tr_C and tr'_C are not free in **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R1} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{R1_C}$ and **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions **TR3** and **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{tr_T and tr'_T are not free in **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{R1_T}$ and **R1**}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$

{Conjunctive healthiness conditions $\mathbf{R0_T}$ and $\mathbf{R1}$}

$= \mathbf{CI0132} \circ \mathbf{R1} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$

{Definition of **CI0132** and Lemma L.4.1.3}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$ {Lemma L.3.6.2}

$= \mathbf{CITR}(P)$

□

Lemma L.1.1.2 ***Isabelle proof available: Section 9.5***

$$\mathbf{CITR} \circ \mathbf{R3}(P) = \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))$$

Proof.

$\mathbf{CITR} \circ \mathbf{R3}(P)$ {Lemma L.4.6.9}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

{Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

{Definition of $\mathbf{CI4_m}$, **CI4**, $\mathbf{R0_T}$ and $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

{Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

□

Lemma L.1.1.3 ***Isabelle proof available: Section 9.5***

$\mathbf{CITR} \circ \mathbf{R3}(P)$

$=$

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$$

Proof.

$$\begin{aligned}
&\mathbf{CITR} \circ \mathbf{R3}(P) && \{\text{Lemma L.3.6.2}\}\\
&= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P) && \{\text{Lemma L.4.6.1}\}\\
&= \mathbf{R0_T} \circ \mathbf{CI0132} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P) && \{\text{Lemma L.4.6.2}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0132} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P) && \{\text{Lemma L.4.6.3}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{CI4_m} \circ \mathbf{R3}(P) && \{\text{Lemma L.4.6.6}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P) && \{\text{Lemma L.2.2.12}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P) && \{\text{Lemma L.2.5.1}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P) && \{\text{Lemma L.4.7.16}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{R3}(P) && \{\text{Lemma L.4.6.10}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P)) && \{\text{Lemma L.4.7.17}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P)) && \{\text{Lemma L.2.5.1}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P)) && \{\text{Lemma L.2.2.12}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))
\end{aligned}$$

□

Lemma L.1.1.4 ***Isabelle proof available: Section 9.5***

$$\mathbf{CITR} \circ \mathbf{R}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))$$

Proof.

$$
\begin{aligned}
&\mathbf{CITR} \circ \mathbf{R}(P) && \{\text{Definition of } \mathbf{R}\}\\
&= \mathbf{CITR} \circ \mathbf{R1} \circ \mathbf{R2} \circ \mathbf{R3}(P) && \{\text{Lemma L.3.6.1}\}\\
&= \mathbf{CITR} \circ \mathbf{R2} \circ \mathbf{R3}(P) && \{\text{Commutativity of } \mathbf{R2} \text{ and } \mathbf{R3}\}\\
&= \mathbf{CITR} \circ \mathbf{R3} \circ \mathbf{R2}(P) && \{\text{Lemma L.4.6.8}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI0134} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))
\end{aligned}
$$

□

Lemma L.1.1.5 *Provided ok' and $wait$ are not free in P,*

$$
\begin{aligned}
&\mathbf{CITR} \circ \mathbf{R}(P \vdash Q)\\
&=\\
&\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P)\\ \vdash\\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T)\\ \wedge\\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&\mathbf{CITR} \circ \mathbf{R}(P \vdash Q) && \{\text{Lemma L.3.6.3}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \mathbf{R2}(P \vdash Q)\\ \wedge\\ \mathbf{CI2_m} \circ \mathbf{R2}(P \vdash Q) \end{array} \right)\\
& && \{\text{Property of designs}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q))\\ \wedge\\ \mathbf{CI2_m}(\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) \end{array} \right)\\
& && \{\text{Lemma L.4.5.5}\}
\end{aligned}
$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\ \mathbf{R2}(\neg\ P) \vdash \mathbf{R2}(Q)) \\ \wedge \\ \mathbf{CI2_m}(\neg\ \neg\ \mathbf{R2}(\neg\ P)) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\ \mathbf{R2}(\neg\ P) \vdash \mathbf{R2}(Q)) \\ \wedge \\ \mathbf{CI2_m} \circ \mathbf{R2}(\neg\ P) \end{array} \right)$$

{Assumption and Lemma L.4.5.5}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

□

Well, it turns out that the **R2** can actually be taken out of the design, so we create a new lemma for this.

Lemma L.1.1.6 ***Isabelle proof available: Section 9.5*** *Provided ok' and $wait$ are not free in P,*

$$\mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$$
$$=$$
$$\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

Proof.

$\mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$ {Lemma L.3.6.6}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} (wait' \vee tr' \neq tr \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{$\mathbf{R2}(tr' \neq tr) = tr' \neq tr$}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2}(tr' \neq tr) \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Distributivity of **R2**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \mathbf{R2} \left(\begin{array}{l} (wait' \vee tr' \neq tr \vee \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \mathbf{R2} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Definition of design and predicate calculus (Lemma L.4.6.11)}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

□

Lemma L.1.1.7 ***Isabelle proof available: Section 9.5***

$$\mathbf{CITR}(P) = \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$$

Proof.

$\mathbf{CITR}(P)$ {Definition of **CITR**}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR} \circ \mathbf{R2_{loc}}(P)$ {Definition of **TR**}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{TR4} \circ \mathbf{R2_{loc}}(P)$ {Lemma L.6.0.1}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T}(P)$ {Definition of $\mathbf{R0_T}$}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{R0_T} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T}(P)$ {Lemma L.1.1.8}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{TR3} \circ \mathbf{R2_T}(P)$ {Commutativity of conjunctive healthiness conditions (**TR3**, $\mathbf{CI4_m}$)}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$

□

Lemma L.1.1.8

$$\mathbf{CIB} \circ \mathbf{R0_T} \circ \mathbf{TR2}(P) = \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$$

Proof.

$\mathbf{CIB} \circ \mathbf{R0_T} \circ \mathbf{TR2}(P)$ {Definition of **CIB**, $\mathbf{R0_T}$ and **TR2** (Lemma L.3.5.2)}

$$= \left(\begin{array}{l} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T \end{array} \right)$$

{Definition of $\mathbf{R0_T}$}

$$= \mathbf{R0_T} \left(\begin{array}{l} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ front(tr_T) < tr'_T \end{array} \right)$$

$\{$Definition of $\mathbf{R1_T}\}$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \end{array} \right)$$

$\{$Definition of $\mathbf{CI4_m}\}$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$$

□

Applying **CITR** to **R**.

Lemma L.1.1.9

$$\mathbf{CITR} \circ \mathbf{R}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))$$

Proof.

$\mathbf{CITR} \circ \mathbf{R}(P)$ $\{$Definition of $\mathbf{R}\}$

$= \mathbf{CITR} \circ \mathbf{R1} \circ \mathbf{R2} \circ \mathbf{R3}(P)$ $\{$Lemma L.3.6.1$\}$

$= \mathbf{CITR} \circ \mathbf{R2} \circ \mathbf{R3}(P)$ $\{$Commutativity of **R2** and **R3**$\}$

$= \mathbf{CITR} \circ \mathbf{R3} \circ \mathbf{R2}(P)$ $\{$Lemma L.4.6.8$\}$

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))$

□

1.1.1 Results on TR

Lemma L.1.1.10

$$0 < \#s \wedge \#s \leq \#t \wedge \mathit{front}(s) \leq t = 0 < \#s \wedge \#s \leq \#t \wedge \mathit{front}(s) < t$$

Proof. Isabelle theorem: `front_a_lt_b__resultof__length_a_lt_length_b_and_¬front_a_lt_b`. □

Lemma L.1.1.11

$$\mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P) = P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge \mathit{front}(tr_T) < tr'_T$$

Proof.

$\mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P)$ {Definition of **TR0**, **TR1** and **TR2**}
$= P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge \mathit{front}(tr_T) \leq tr'_T$ {Lemma L.3.5.1}
$= P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge \mathit{front}(tr_T) < tr'_T$

□

Chapter 2

Healthiness Conditions of *Circus Time*

2.1 Results on $\mathbf{R0_T}$

Lemma L.2.1.1

$$\mathbf{R0_T}(P)^f_f = \mathbf{R0_T}(P^f_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{R0_T}(P)^f_f & \{\text{Definition of } \mathbf{R0_T}\} \\
= (\mathbf{TR0} \circ \mathbf{TR1}(P))^f_f & \{\text{Lemmas L.3.1.2 and L.3.2.2}\} \\
= \mathbf{TR0} \circ \mathbf{TR1}(P^f_f) &
\end{array}$$

□

Lemma L.2.1.2

$$\mathbf{R0_T}(P)^o_f = \mathbf{R0_T}(P^o_w)$$

Proof.

$$\begin{array}{ll}
\mathbf{R0_T}(P)^o_f & \{\text{Definition of } \mathbf{R0_T}\} \\
= (\mathbf{TR0} \circ \mathbf{TR1}(P))^o_f & \{\text{Lemmas L.3.1.3 and L.3.2.3}\} \\
= \mathbf{TR0} \circ \mathbf{TR1}(P^o_f) & \{\text{Definition of } \mathbf{R0_T}\} \\
= \mathbf{R0_T}(P^o_w) &
\end{array}$$

□

Lemma L.2.1.3 *Provided v is not tr_T nor tr'_T,*

$$\exists\, v \bullet \mathbf{R0_T}(P) = \mathbf{R0_T}(\exists\, v \bullet P)$$

Proof.

$\exists\, v \bullet \mathbf{R0_T}(P)$ {Definition of $\mathbf{R0_T}$}
$= \exists\, v \bullet P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T$
{Assumption: v is not tr_T nor tr'_T and predicate calculus}
$= (\exists\, v \bullet P) \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T$ {Definition of $\mathbf{R0_T}$}
$= \mathbf{R0_T}(\exists\, v \bullet P)$

□

2.2 Results on $\mathbf{R1_T}$

Definition 1

$$Expands_T(s, t) = (front(s) < t) \wedge fst \circ last(s) \leq fst \circ head(t - front(s))$$

Lemma L.2.2.1

$$tr'_A = tr_A \Rightarrow Expands_T(tr_A, tr'_A)$$

Proof.

$tr'_A = tr_A$ {Property of sequences}
$= tr'_A = tr_A \wedge front(tr_A) \leq tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head \circ last(tr'_A)$
{Property of sequences}
$= tr'_A = tr_A \wedge front(tr_A) \leq tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head \circ last(tr'_A - front(tr_A))$
{Definition of $Expands_T$}
$= tr'_A = tr_A \wedge Expands_T(tr_A, tr'_A)$ {Predicate calculus}

$\Rightarrow Expands_T(tr_A, tr'_A)$

□

Lemma L.2.2.2

$$fst \circ head \circ dif_T(tr'_A, tr_A)$$
$$=$$
$$fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A)$$

Proof.

$fst \circ head \circ dif_T(tr'_A, tr_A)$ {Definition of dif_T}

$$= fst \circ head \left(\begin{array}{l} \langle \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(tr_A)) \end{array} \right)$$

{Definition of $head$}

$$= fst \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right)$$

{Definition of fst}

$= fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A)$

□

Lemma L.2.2.3 $\mathbf{R1_T}(false) = false$

Proof.

$\mathbf{R1_T}(false)$ {Definition of $\mathbf{R1_T}$}

$= false \wedge Expands_T(tr_A, tr'_A)$ {Predicate calculus}

$= false$

□

Lemma L.2.2.4 $\mathbf{R1_T}(P \wedge Q) = \mathbf{R1_T}(P) \wedge \mathbf{R1_T}(Q)$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P \wedge Q) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= (P \wedge Q) \wedge \mathit{Expands}_A(tr_A, tr'_A) && \{\text{Predicate calculus}\}\\
&= (P \wedge \mathit{Expands}_A(tr_A, tr'_A)) \wedge (Q \wedge \mathit{Expands}_A(tr_A, tr'_A)) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R1_T}(P) \wedge \mathbf{R1_T}(Q)
\end{aligned}
$$

□

Lemma L.2.2.5 $\mathbf{R1_T}(P \wedge Q) = \mathbf{R1_T}(P) \wedge Q$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P \wedge Q) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= (P \wedge Q) \wedge \mathit{Expands}_T(tr_A, tr'_A) && \{\text{Predicate calculus}\}\\
&= (P \wedge \mathit{Expands}_T(tr_A, tr'_A)) \wedge Q && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R1_T}(P) \wedge Q
\end{aligned}
$$

□

Lemma L.2.2.6 $\mathbf{R1_T}(P \vee Q) = \mathbf{R1_T}(P) \vee \mathbf{R1_T}(Q)$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P \vee Q) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= (P \vee Q) \wedge \mathit{Expands}_A(tr_A, tr'_A) && \{\text{Predicate calculus}\}\\
&= (P \wedge \mathit{Expands}_A(tr_A, tr'_A)) \vee (Q \wedge \mathit{Expands}_A(tr_A, tr'_A)) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R1_T}(P) \vee \mathbf{R1_T}(Q)
\end{aligned}
$$

□

Lemma L.2.2.7 $\mathbf{R1_T}(\mathbf{R1_T}(P)\ ;\ \mathbf{R1_T}(Q)) =$

Proof.

□

Lemma L.2.2.8

$$\mathbf{R1_T}(P \lhd c \rhd Q)$$
$$=$$
$$\mathbf{R1_T}(P) \lhd c \rhd \mathbf{R1_T}(Q)$$

Proof.

$\mathbf{R1_T}(P \lhd c \rhd Q)$ {Definition of conditional}
$= \mathbf{R1_T}((c \wedge P) \vee (\neg\ c \wedge Q))$ {Lemma L.2.2.6}
$= \mathbf{R1_T}(c \wedge P) \vee \mathbf{R1_T}(\neg\ c \wedge Q)$ {Lemma L.2.2.5}
$= (c \wedge \mathbf{R1_T}(P)) \vee (\neg\ c \wedge \mathbf{R1_T}(Q))$ {Definition of conditional}
$= \mathbf{R1_T}(P) \lhd c \rhd \mathbf{R1_T}(Q)$

□

Lemma L.2.2.9

$$\mathbf{R1_T}(P \lhd c \rhd Q)$$
$$=$$
$$(c \wedge \mathbf{R1_T}(P)) \vee (\neg\ c \wedge \mathbf{R1_T}(Q))$$

Proof.

$\mathbf{R1_T}(P \lhd c \rhd Q)$ {Lemma L.2.2.8}
$= \mathbf{R1_T}(P) \lhd c \rhd \mathbf{R1_T}(Q)$ {Definition of conditional}
$= (c \wedge \mathbf{R1_T}(P)) \vee (\neg\ c \wedge \mathbf{R1_T}(Q))$

□

Lemma L.2.2.10

$$(P\ ;\ \mathbf{R1_T}(\mathit{true})) \vee (P\ ;\ \mathbf{R1_T}(Q))$$

$$=$$

$$(P \ ; \mathbf{R1_T}(true))$$

Proof.

$$\begin{aligned}
&(P \ ; \mathbf{R1_T}(true)) \vee (P \ ; \mathbf{R1_T}(Q)) && \{\text{Distributivity of sequential composition}\}\\
&= P \ ; (\mathbf{R1_T}(true) \vee \mathbf{R1_T}(Q)) && \{\text{Lemma L.2.2.6}\}\\
&= P \ ; \mathbf{R1_T}(true \vee Q) && \{\text{Predicate calculus}\}\\
&= P \ ; \mathbf{R1_T}(true)
\end{aligned}$$

□

Lemma L.2.2.11

$$Expands_T(tr_A, tr'_A)$$

$$=$$

$$\exists\, s : \underset{1}{\text{seq}} \bullet front(tr_A) \frown s = tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head(s)$$

Proof.

$$\begin{aligned}
&Expands_T(tr_A, tr'_A) && \{\text{Definition of } Expands_T\}\\
&= (front(tr_A) < tr'_A) \wedge fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\
& && \{\text{Definition of sequence prefixing}\}\\
&= (\exists\, s : \underset{1}{\text{seq}} \bullet front(tr_A) \frown s = tr'_A) \wedge fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\
& && \{\text{Transitivity of equality}\}\\
&= (\exists\, s : \underset{1}{\text{seq}} \bullet front(tr_A) \frown s = tr'_A) \wedge fst \circ last(tr_A) \leq fst \circ head(front(tr_A) \frown s - front(tr_A)) \\
& && \{\text{Property of sequences}\}\\
&= (\exists\, s : \underset{1}{\text{seq}} \bullet front(tr_A) \frown s = tr'_A) \wedge fst \circ last(tr_A) \leq fst \circ head(s)
\end{aligned}$$

□

Lemma L.2.2.12

$$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(P) = \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(P) && \{\text{Definition of } \mathbf{R1_T}\} \\
&= \mathbf{R2_T} \circ \mathbf{R1_T}(P) \land \mathit{Expands}_T(tr_A, tr'_A) && \{\text{Lemma L.2.3.14}\} \\
&= \mathbf{R2_T}(P) \land \mathit{Expands}_T(tr_A, tr'_A) && \{\text{Definition of } \mathbf{R1_T}\} \\
&= \mathbf{R1_T} \circ \mathbf{R2_T}(P)
\end{aligned}
$$

□

Lemma L.2.2.13

$$
\begin{aligned}
&\mathit{Expands}_T(tr_A, tr'_A) \\
&= \\
&\exists\, t : \operatorname{seq} \Sigma, r : \mathbb{P}\, \Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \land \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq t \end{array} \right)
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&\mathit{Expands}_T(tr_A, tr'_A) && \{\text{Definition of } \mathit{Expands}_T\} \\
&= (\mathit{front}(tr_A) < tr'_A) \land \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) \\
& && \{\text{Definition of sequence prefixing}\} \\
&= \exists\, t : \operatorname{seq} \Sigma, r : \mathbb{P}\, \Sigma, s : \operatorname{seq}(\operatorname{seq} \Sigma \times \mathbb{P}\, \Sigma) \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \land \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) \end{array} \right) \\
& && \{\text{Transitivity of equality}\} \\
&= \exists\, t : \operatorname{seq} \Sigma, r : \mathbb{P}\, \Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \land \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head} \left(\begin{array}{l} \mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s \\ - \\ \mathit{front}(tr_A) \end{array} \right) \end{array} \right) \\
& && \{\text{Property of sequences}\}
\end{aligned}
$$

$$= \exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown t = tr'_A) \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(\langle (t, r) \rangle \frown s) \end{array} \right)$$

{Definition of *head*}

$$= \exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst}((t, r)) \end{array} \right)$$

{Definition of *fst*}

$$= \exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq t \end{array} \right)$$

□

Lemma L.2.2.14

$$\mathit{Expands}_T(tr_A, tr'_A)$$
$$\Rightarrow$$
$$\exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A$$

Proof.

$\mathit{Expands}_T(tr_A, tr'_A)$

{Lemma L.2.2.13}

$$= \exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \left(\begin{array}{l} (\mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A) \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq t \end{array} \right)$$

{Predicate calculus}

$$\Rightarrow \exists\, t : \operatorname{seq}\Sigma, r : \mathbb{P}\,\Sigma, s \bullet \mathit{front}(tr_A) \frown \langle (t, r) \rangle \frown s = tr'_A$$

□

Lemma L.2.2.15

$$\mathbf{R1_T}(P)\ ;\ (\neg\ \mathit{wait}' \wedge tr'_A = tr_A)$$
$$=$$
$$\mathbf{R1_T}((P\ ;\ tr'_A = tr_A) \wedge \neg\ \mathit{wait}')$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P)\ ;\ (\neg\ wait' \wedge tr'_A = tr_A) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= (P \wedge Expands_T(tr_A, tr'_A))\ ;\ (\neg\ wait' \wedge tr'_A = tr_A)\\
& && \{\text{Definition of sequential composition, where } tr'_A \notin v\}\\
&= \exists\, v_0, tr_0 \bullet (P \wedge Expands_T(tr_A, tr'_A))[v_0, tr_0/v', tr'_A] \wedge (\neg\ wait' \wedge tr'_A = tr_A)[v_0, tr_0/v, tr_A]\\
& && \{\text{Substitution}\}\\
&= \exists\, v_0, tr_0 \bullet (P[v_0, tr_0/v', tr'_A] \wedge Expands_T(tr_A, tr_0)) \wedge (\neg\ wait' \wedge tr'_A = tr_0)\\
& && \{\text{One-point rule}\}\\
&= \exists\, v_0 \bullet (P[v_0/v'] \wedge Expands_T(tr_A, tr'_A)) \wedge \neg\ wait' && \{\text{Predicate calculus}\}\\
&= (\exists\, v_0 \bullet P[v_0/v']) \wedge Expands_T(tr_A, tr'_A) \wedge \neg\ wait' && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R1_T}((\exists\, v_0 \bullet P[v_0/v']) \wedge \neg\ wait') && \{\text{Definition of sequential composition}\}\\
&= \mathbf{R1_T}((\exists\, v_0, tr_0 \bullet P[v_0, tr_0/v', tr_A] \wedge tr'_A = tr_0) \wedge \neg\ wait')\\
& && \{\text{Definition of sequential composition}\}\\
&= \mathbf{R1_T}((P\ ;\ tr'_A = tr_A) \wedge \neg\ wait')
\end{aligned}
$$

□

Lemma L.2.2.16

$$
\begin{aligned}
&\mathbf{R1_T}(P)\ ;\ fst \circ head \circ dif_T(tr'_A, tr_A) \neq \langle\rangle\\
&=\\
&\exists\, s : \underset{1}{\text{seq}} \bullet \left(\begin{array}{l} (\exists\, v_0 \bullet P[v_0/v'])[front(tr_A) \frown s/tr'_A] \wedge fst \circ last(tr_A) \leq fst \circ head(s)\\ \wedge\\ fst \circ head(tr'_A - front(front(tr_A) \frown s)) - fst \circ last(s) \neq \langle\rangle \end{array} \right)
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P)\ ;\ fst \circ head \circ dif_T(tr'_A, tr_A) \neq \langle\rangle && \{\text{Lemma L.2.2.2}\}\\
&= \mathbf{R1_T}(P)\ ;\ fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) \neq \langle\rangle && \{\text{Definition of } \mathbf{R1_T}\}\\
&= (P \wedge Expands_T(tr_A, tr'_A))\ ;\ fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) \neq \langle\rangle\\
& && \{\text{Definition of } Expands_T \text{ (Lemma L.2.2.11)}\}
\end{aligned}
$$

$$= \left(\begin{array}{l} (P \wedge \exists\, s : \mathrm{seq}_1 \bullet front(tr_A) \frown s = tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head(s)) \\ \; ; \\ fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) \neq \langle\rangle \end{array} \right)$$

{Definition of sequential composition}

$$= \exists\, tr_0, v_0 \bullet \left(\begin{array}{l} \left(\begin{array}{l} P \\ \wedge \\ \exists\, s : \mathrm{seq}_1 \bullet front(tr_A) \frown s = tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head(s) \end{array} \right)[tr_0, v_0/tr'_A, v \\ \wedge \\ (fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) \neq \langle\rangle)[tr_0, v_0/tr_A, v] \end{array} \right)$$

{Substitution}

$$= \exists\, tr_0, v_0 \bullet \left(\begin{array}{l} P[tr_0, v_0/tr'_A, v'] \\ \wedge \\ \exists\, s : \mathrm{seq}_1 \bullet front(tr_A) \frown s = tr_0 \wedge fst \circ last(tr_A) \leq fst \circ head(s) \\ \wedge \\ fst \circ head(tr'_A - front(tr_0)) - fst \circ last(tr_0) \neq \langle\rangle \end{array} \right)$$

{One-point rule}

$$= \exists\, tr_0, v_0, s : \underset{1}{\mathrm{seq}} \bullet \left(\begin{array}{l} P[v_0/v'][front(tr_A) \frown s/tr'_A] \wedge fst \circ last(tr_A) \leq fst \circ head(s) \\ \wedge \\ fst \circ head(tr'_A - front(front(tr_A) \frown s)) - fst \circ last(front(tr_A) \frown s) \neq \langle\rangle \end{array} \right)$$

{Property of sequences}

$$= \exists\, v_0, s : \underset{1}{\mathrm{seq}} \bullet \left(\begin{array}{l} P[v_0/v'][front(tr_A \frown s/tr'_A] \wedge fst \circ last(tr_A) \leq fst \circ head(s) \\ \wedge \\ fst \circ head(tr'_A - front(front(tr_A) \frown s)) - fst \circ last(s) \neq \langle\rangle \end{array} \right)$$

{Predicate calculus}

$$= \exists\, s : \underset{1}{\mathrm{seq}} \bullet \left(\begin{array}{l} (\exists\, v_0 \bullet P[v_0/v'])[front(tr_A) \frown s/tr'_A] \wedge fst \circ last(tr_A) \leq fst \circ head(s) \\ \wedge \\ fst \circ head(tr'_A - front(front(tr_A) \frown s)) - fst \circ last(s) \neq \langle\rangle \end{array} \right)$$

□

Lemma L.2.2.17

$$\mathbf{R1_T}(tr'_A \uparrow_R = tr_A \uparrow_R) = tr'_A \uparrow_R = tr_A \uparrow_R$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(tr'_A \uparrow_R = tr_A \uparrow_R) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= tr'_A \uparrow_R = tr_A \uparrow_R \wedge Expands_T(tr_A, tr'_A) && \{\text{Lemma 19 and predicate calculus}\}\\
&= tr'_A \uparrow_R = tr_A \uparrow_R
\end{aligned}
$$

□

Lemma L.2.2.18 ***TODO: Can*** *Flat* ***also be defined on empty sequences?*** *Provided s is a non-empty sequence, $Flat(front(t) \frown s) - Flat(t) = fst \circ head(s) - fst \circ last(t) \frown Flat \circ tail(s)$*

Proof.

$$
\begin{aligned}
&Flat(front(t) \frown s) - Flat(t) && \{\text{Lemma 24}\}\\
&= Flat(dif_T(front(t) \frown s, t)) && \{\text{Definiton of } dif_T\}\\
&= Flat\left(\begin{array}{l} \langle \left(\begin{array}{l} fst \circ head(front(t) \frown s - front(t)) - fst \circ last(t), \\ snd \circ head(front(t) \frown s - front(t)) \end{array}\right) \rangle \\ \frown \\ tail(front(t) \frown s - front(t)) \end{array}\right) && \{\text{Property of sequences}\}\\
&= Flat(\langle (fst \circ head(s) - fst \circ last(t), snd \circ head(s)) \rangle \frown tail(s)) && \{\text{Assumption and property of } Flat\}\\
&= Flat(\langle (fst \circ head(s) - fst \circ last(t), snd \circ head(s)) \rangle) \frown Flat \circ tail(s) && \{\text{Definition of } Flat\}\\
&= fst \circ head(s) - fst \circ last(t) \frown Flat \circ tail(s)
\end{aligned}
$$

□

Lemma L.2.2.19

$$\mathbf{R1_T}(P) \ ; \mathbf{R1_T}(Q) \Rightarrow \mathbf{R1_T}(P) \ ; \mathbf{R1_T}(true)$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P) \ ; \mathbf{R1_T}(Q) && \{\text{Predicate calculus}\}
\end{aligned}
$$

$= \mathbf{R1_T}(P)\ ;\ \mathbf{R1_T}(Q \wedge \mathit{true})$ {Lemma L.2.2.5}

$= \mathbf{R1_T}(P)\ ;\ (\mathbf{R1_T}(Q) \wedge \mathbf{R1_T}(\mathit{true}))$ {Lemma L.8.0.11}

$\Rightarrow \mathbf{R1_T}(P)\ ;\ \mathbf{R1_T}(\mathit{true})$

□

Lemma L.2.2.20

$$\mathbf{R1_T}(P) = \mathbf{R1_T}(Q) \Leftrightarrow [\mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A) \Rightarrow (P \Leftrightarrow Q)]$$

Proof.

$\mathbf{R1_T}(P) = \mathbf{R1_T}(Q)$ {Definition of **R1**}

$\Leftrightarrow (P \wedge \mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A)) = (Q \wedge \mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A))$ {Equality}

$\Leftrightarrow [(P \wedge \mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A)) \Leftrightarrow (Q \wedge \mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A))]$ {Lemma L.8.0.9}

$\Leftrightarrow [\mathit{Expands}_T(\mathit{tr}_A, \mathit{tr}'_A) \Rightarrow (P \Leftrightarrow Q)]$

□

Lemma L.2.2.21

$$\mathbf{R1_T}(P) \wedge \mathit{tr} = \mathit{Flat}(\mathit{tr}_A) \wedge \mathit{tr}' = \mathit{Flat}(\mathit{tr}'_A) \Rightarrow \mathit{tr} \leq \mathit{tr}'$$

Proof.

$\mathbf{R1_T}(P) \wedge \mathit{tr} = \mathit{Flat}(\mathit{tr}_A) \wedge \mathit{tr}' = \mathit{Flat}(\mathit{tr}'_A)$ {Lemma L.2.2.22}

$\Rightarrow \mathit{Flat}(\mathit{tr}_A) \leq \mathit{Flat}(\mathit{tr}'_A) \wedge \mathit{tr} = \mathit{Flat}(\mathit{tr}_A) \wedge \mathit{tr}' = \mathit{Flat}(\mathit{tr}'_A)$

{Transitivity of equality}

$= \mathit{tr} \leq \mathit{tr}' \wedge \mathit{tr} = \mathit{Flat}(\mathit{tr}_A) \wedge \mathit{tr}' = \mathit{Flat}(\mathit{tr}'_A)$ {Predicate calculus}

$\Rightarrow \mathit{tr} \leq \mathit{tr}'$

□

Lemma L.2.2.22

$$\mathbf{R1_T}(P) \Rightarrow \mathit{Flat}(\mathit{tr}_A) \leq \mathit{Flat}(\mathit{tr}'_A)$$

Proof.

$$
\begin{aligned}
&\mathbf{R1_T}(P) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= P \wedge \mathit{Expands}_T(tr_A, tr'_A) && \{\text{Definition of } \mathit{Expands}_T\}\\
&= P \wedge \mathit{front}(tr_A) < tr'_A \wedge \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) && \{\text{Property of sequences}\}\\
&= P \wedge \exists\, s : \underset{1}{\mathrm{seq}} \bullet \mathit{front}(tr_A) \frown s = tr'_A \wedge \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) && \{\text{Predicate calculus}\}\\
&= \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A) \frown s) = \mathit{Flat}(tr'_A) \end{array} \right) && \{\text{Definition of } \mathit{Flat}\}\\
&= \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(s) = \mathit{Flat}(tr'_A) \end{array} \right) && \{\text{Definition of } \mathit{Flat}\}\\
&= \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A)) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(s) = \mathit{Flat}(tr'_A) \end{array} \right) && \{\text{Transitivity of equality}\}\\
&= \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(\mathit{front}(tr_A) \frown s - \mathit{front}(tr_A)) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(s) = \mathit{Flat}(tr'_A) \end{array} \right) && \{\text{Property of sequences}\}
\end{aligned}
$$

$$
\begin{array}{ll}
= & \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(s) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(s) = \mathit{Flat}(tr'_A) \end{array} \right) & \{\text{Predicate calculus}\} \\
\Rightarrow & \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(s) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(s) \leq \mathit{Flat}(tr'_A) \end{array} \right) & \{\text{Property of sequences}\} \\
= & \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(s) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(\mathit{head}(s) \frown \mathit{tail}(s)) \leq \mathit{Flat}(tr'_A) \end{array} \right) & \{\text{Definition of } \mathit{Flat}\} \\
= & \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat} \circ \mathit{head}(s) \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat}(\mathit{head}(s)) \frown \mathit{Flat}(\mathit{tail}(s)) \leq \mathit{Flat}(tr'_A) \end{array} \right) & \{\text{Lemma L.2.2.23}\} \\
\Rightarrow & \left(\begin{array}{l} P \wedge \exists\, s : \mathrm{seq}_1 \bullet \mathit{front}(tr_A) \frown s = tr'_A \\ \wedge \\ \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat}(tr'_A) \end{array} \right) & \{\text{Predicate calculus}\} \\
\Rightarrow & \mathit{Flat}(\mathit{front}(tr_A)) \frown \mathit{Flat} \circ \mathit{last}(tr_A) \leq \mathit{Flat}(tr'_A) & \{\text{Definition of } \mathit{Flat}\} \\
= & \mathit{Flat}(\mathit{front}(tr_A) \frown \mathit{last}(tr_A)) \leq \mathit{Flat}(tr'_A) & \{\text{Property of sequences}\}
\end{array}
$$

□

Lemma L.2.2.23

$$s \leq t \wedge e \frown t \leq u \Rightarrow e \frown s \leq u$$

Proof.

$s \leq t \wedge e \frown t \leq u$ {Property of sequence prefixing}
$= \exists\, o \bullet s \frown o = t \wedge e \frown t \leq u$ {Transitivity of equality}
$= \exists\, o \bullet s \frown o = t \wedge e \frown s \frown o \leq u$ {Predicate calculus}
$\Rightarrow \exists\, o \bullet e \frown s \frown o \leq u$ {Property of sequences}
$\Rightarrow e \frown s \leq u$

□

Lemma L.2.2.24

$$\mathbf{R1_T}(P)^f_f = \mathbf{R1_T}(P^f_f)$$

Proof.

$\mathbf{R1_T}(P)^f_f$ {Definition of $\mathbf{R1_T}$}
$= (P \wedge \mathit{Expands}_A(tr_T, tr'_T))^f_f$ {Substitution}
$= P^f_f \wedge \mathit{Expands}_A(tr_T, tr'_T)$ {Definition of $\mathbf{R1_T}$}
$= \mathbf{R1_T}(P^f_f)$

□

Lemma L.2.2.25

$$\mathbf{R1_T}(P)^o_f = \mathbf{R1_T}(P^o_f)$$

Proof.

$\mathbf{R1_T}(P)^o_f$ {Definition of $\mathbf{R1_T}$}
$= (P \wedge \mathit{Expands}_A(tr_T, tr'_T))^o_f$ {Substitution}
$= P^o_f \wedge \mathit{Expands}_A(tr_T, tr'_T)$ {Definition of $\mathbf{R1_T}$}
$= \mathbf{R1_T}(P^o_f)$

□

Lemma L.2.2.26 *Provided v is not tr_T nor tr'_T,*

$$\exists\, v \bullet \mathbf{R1_T}(P) = \mathbf{R1_T}(\exists\, v \bullet P)$$

Proof.

$$\begin{array}{ll}
\exists\, v \bullet \mathbf{R1_T}(P) & \{\text{Definition of } \mathbf{R1_T}\} \\
= \exists\, v \bullet P \wedge \mathit{front}(tr_T) < tr'_T \wedge \mathit{fst} \circ \mathit{last}(tr_T) \leq \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) & \\
\qquad \{\text{Assumption: } v \text{ is not } tr_T \text{ nor } tr'_T \text{ and predicate calculus}\} & \\
= (\exists\, v \bullet P) \wedge \mathit{front}(tr_T) < tr'_T \wedge \mathit{fst} \circ \mathit{last}(tr_T) \leq \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) & \\
\qquad \{\text{Definition of } \mathbf{R1_T}\} & \\
= \mathbf{R1_T}(\exists\, v \bullet P) &
\end{array}$$

□

2.3 Results on $\mathbf{R2_T}$

Lemma L.2.3.1

$$\mathbf{R2_T}(P)^f_f = \mathbf{R2_T}(P^f_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_T}(P)^f_f & \{\text{Definition of } \mathbf{R2_T}\} \\
= (P[\langle(\langle\rangle, \mathit{snd} \circ \mathit{last}(tr_T))\rangle, \mathit{dif}_T(tr'_T, tr_T)/tr_T, tr'_T])^f_f & \{\text{Substitution}\} \\
= P^f_f[\langle(\langle\rangle, \mathit{snd} \circ \mathit{last}(tr_T))\rangle, \mathit{dif}_T(tr'_T, tr_T)/tr_T, tr'_T] & \{\text{Definition of } \mathbf{R1_T}\} \\
= \mathbf{R2_T}(P^f_f) &
\end{array}$$

□

Lemma L.2.3.2

$$\mathbf{R2_T}(P)^o_f = \mathbf{R2_T}(P^o_f)$$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(P)^o_f & \{\text{Definition of } \mathbf{R2_T}\} \\
= (P[\langle(\langle\rangle, snd \circ last(tr_T))\rangle, dif_T(tr'_T, tr_T)/tr_T, tr'_T])^o_f & \{\text{Substitution}\} \\
= P^f_f[\langle(\langle\rangle, snd \circ last(tr_T))\rangle, dif_T(tr'_T, tr_T)/tr_T, tr'_T] & \{\text{Definition of } \mathbf{R1_T}\} \\
= \mathbf{R2_T}(P^o_f) &
\end{array}
$$

□

Lemma L.2.3.3

$$\mathbf{R2_T}(\#tr'_T = \#tr_T) = \#tr'_T = \#tr_T$$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(\#tr'_T = \#tr_T) & \{\text{Definition of } \mathbf{R2_T} \text{ and substitution}\} \\
= \#dif_T(tr'_T, tr_T) = 1 & \{\text{Lemma 26}\} \\
= (\#tr'_T - \#tr_T) + 1 = 1 & \{\text{Arithmetic}\} \\
= \#tr'_T = \#tr_T &
\end{array}
$$

□

Lemma L.2.3.4 *Provided v is not tr_T nor tr'_T,*

$$\exists\, v \bullet \mathbf{R2_T}(P) = \mathbf{R2_T}(\exists\, v \bullet P)$$

Proof.

$$
\begin{array}{ll}
\exists\, v \bullet \mathbf{R2_T}(P) & \{\text{Definition of } \mathbf{R2_T}\} \\
= \exists\, v \bullet P[\langle(\langle\rangle, snd \circ last(tr_T))\rangle, dif_T(tr'_T, tr_T)/tr_T, tr'_T] & \\
\qquad \{\text{Assumption: } v \text{ is not } tr_T \text{ nor } tr'_T \text{ and predicate calculus}\} & \\
= (\exists\, v \bullet P)[\langle(\langle\rangle, snd \circ last(tr_T))\rangle, dif_T(tr'_T, tr_T)/tr_T, tr'_T] & \{\text{Definition of } \mathbf{R2_T}\} \\
= \mathbf{R2_T}(\exists\, v \bullet P) &
\end{array}
$$

□

Lemma L.2.3.5

$$\mathbf{R2_T}(\#tr'_T = \#tr_T) = \#tr'_T = \#tr_T$$

Proof.

$$\begin{aligned}
&\mathbf{R2_T}(\#tr'_T = \#tr_T) && \{\text{Definition of } \mathbf{R2_T} \text{ and substitution}\}\\
&= \#dif_T(tr'_T, tr_T) = \#(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) && \{\text{Property of sequences}\}\\
&= \#dif_T(tr'_T, tr_T) = 1 && \{\text{Lemma 26}\}\\
&= ((\#tr'_T - \#tr_T) + 1) = 1 && \{\text{Arithmetic}\}\\
&= \#tr'_T = \#tr_T
\end{aligned}$$

□

Lemma L.2.3.6

$$\begin{aligned}
&dif_T(t, s) = \langle(u, v)\rangle\\
&=\\
&\left(\begin{array}{l}
(fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A)) = u\\
\wedge\\
snd \circ head(tr'_A - front(tr_A)) = v\\
\wedge\\
tail(tr'_A - front(tr_A)) = \langle\rangle
\end{array}\right)
\end{aligned}$$

Proof.

$$\begin{aligned}
&dif_T(t, s) = \langle(u, v)\rangle && \{\text{Definition of } dif_T\}\\
&= \left(\begin{array}{l}
\langle \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A), \\ snd \circ head(tr'_A - front(tr_A)) \end{array}\right) \rangle\\
\frown\\
tail(tr'_A - front(tr_A))
\end{array}\right) = \langle(u, v)\rangle\\
& && \{\text{Property of sequences}\}
\end{aligned}$$

$$= \left(\begin{array}{l} (fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A)) = u \\ \wedge \\ snd \circ head(tr'_A - front(tr_A)) = v \\ \wedge \\ tail(tr'_A - front(tr_A)) = \langle\rangle \end{array} \right)$$

□

Lemma L.2.3.7

$$dif_T(dif_T(tr'_A, tr_A), \langle(\langle\rangle, snd \circ last(tr_A))\rangle)$$
$$=$$
$$dif_T(tr'_A, tr_A)$$

Proof.

$dif_T(dif_T(tr'_A, tr_A), \langle(\langle\rangle, snd \circ last(tr_A))\rangle)$ {Definition of dif_T}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(dif_T(tr'_A, tr_A) - front(\langle(\langle\rangle, snd \circ last(tr_A))\rangle)) \\ - \\ fst \circ last(\langle(\langle\rangle, snd \circ last(tr_A))\rangle) \end{array} \right), \\ snd \circ head(dif_T(tr'_A, tr_A) - front(\langle(\langle\rangle, snd \circ last(tr_A))\rangle)) \end{array} \right) \rangle \\ \frown \\ tail(dif_T(tr'_A, tr_A) - front(\langle(\langle\rangle, snd \circ last(tr_A))\rangle)) \end{array} \right)$$

{Property of sequences: $front(\langle e\rangle) = \langle\rangle$}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(dif_T(tr'_A, tr_A) - \langle\rangle) \\ - \\ fst \circ last(\langle(\langle\rangle, snd \circ last(tr_A))\rangle) \end{array} \right), \\ snd \circ head(dif_T(tr'_A, tr_A) - \langle\rangle) \end{array} \right) \rangle \\ \frown \\ tail(dif_T(tr'_A, tr_A) - \langle\rangle) \end{array} \right)$$

{Property of sequences: $last(\langle e\rangle) = e$}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(dif_T(tr'_A, tr_A) - \langle\rangle) \\ - \\ fst(\langle\rangle, snd \circ last(tr_A)) \end{array} \right), \\ snd \circ head(dif_T(tr'_A, tr_A) - \langle\rangle) \end{array} \right) \rangle \\ \frown \\ tail(dif_T(tr'_A, tr_A) - \langle\rangle) \end{array} \right)$$

{Property of sequences and definition of *fst*}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} fst \circ head(dif_T(tr'_A, tr_A)), snd \circ head(dif_T(tr'_A, tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(dif_T(tr'_A, tr_A)) \end{array} \right)$$

{Property of sequences}

$= head(dif_T(tr'_A, tr_A)) \frown tail(dif_T(tr'_A, tr_A))$ {Property of sequences}

$= dif_T(tr'_A, tr_A)$

□

Lemma L.2.3.8 *Provided* $\#tr_A = \#tr'_A$ *and* $front(tr_A) < tr'_A$,

$$last \circ dif_T(tr'_A, tr_A) = (fst \circ last(tr'_A) - fst \circ last(tr_A), snd \circ last(tr'_A))$$

Proof.

$last \circ dif_T(tr'_A, tr_A)$ {Definition of dif_T}

$$= last \left(\left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \frown tail(tr'_A - front(tr_A)) \right)$$

{Assumption: $\#tr_A = \#tr'_A$ and property of sequences $tr'_A - front(tr_A)) = \langle last(tr'_A)\rangle$}

$$= last \left(\left(\begin{array}{l} fst \circ head(\langle last(tr'_A)\rangle) - fst \circ last(tr_A), \\ snd \circ head(\langle last(tr'_A)\rangle) \end{array} \right) \rangle \frown tail(\langle last(tr_A)\rangle) \right)$$

{Property of sequences}

$$= last \left(\langle \left(\begin{array}{l} fst \circ last(tr'_A) - fst \circ last(tr_A), \\ snd \circ last(tr'_A) \end{array} \right) \rangle \frown \langle\rangle \right)$$ {Property of sequences}

$= last(\langle(fst \circ last(tr'_A) - fst \circ last(tr_A), snd \circ last(tr_A))\rangle)$ {Definition of *last*}

$= (fst \circ last(tr'_A) - fst \circ last(tr_A), snd \circ last(tr'_A))$

□

Lemma L.2.3.9 *Provided* $\#tr_A < \#tr'_A$ *and* $front(tr_A) < tr'_A$,

$$last \circ dif_T(tr'_A, tr_A) = last(tr'_A)$$

Proof.

$last \circ dif_T(tr'_A, tr_A)$ {Definition of dif_T}

$$= last \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(tr_A)) \end{array} \right)$$

{Assumption: $\#tr_A < \#tr'_A$ and property of sequences $\#tail(tr'_A - front(tr_A)) > 1$}

$= last \circ tail(tr'_A - front(tr_A))$ {Property of sequences}

$= last \circ (tr'_A - front(tr_A))$ {Property of sequences}

$= last(tr'_A)$

□

Lemma L.2.3.10 *Provided* $\#tr_A \leq \#tr'_A$ *and* $front(tr_A) < tr'_A$,

$$snd \circ last \circ dif_T(tr'_A, tr_A) = snd \circ last(tr'_A)$$

Proof. (Case: $\#tr_A = \#tr'_A$)

$snd \circ last \circ dif_T(tr'_A, tr_A)$ {Lemma L.2.3.8}

$= snd(fst \circ last(tr'_A) - fst \circ last(tr_A), snd \circ last(tr'_A))$ {Definition of *snd*}

$= snd \circ last(tr'_A)$

□

Proof. (Case: $\#tr_A < \#tr'_A$)

$$
\begin{array}{ll}
snd \circ last \circ dif_T(tr'_A, tr_A) & \{\text{Lemma L.2.3.9}\} \\
= snd \circ last(tr'_A) &
\end{array}
$$

□

Lemma L.2.3.11

$$\mathbf{R2_T}(\neg\ P) = \neg\ \mathbf{R2_T}(P)$$

Proof.

$$
\begin{array}{ll}
\neg\ \mathbf{R2_T}(P) & \{\text{Definition of } \mathbf{R2_T}\} \\
= \neg\ (P)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \{\text{Property of substitution}\} \\
= (\neg\ P)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \{\text{Definition of } \mathbf{R2_T}\} \\
= \mathbf{R2_T}(\neg\ P) &
\end{array}
$$

□

Lemma L.2.3.12 *Provided tr'_A and tr_A are not free in P,*

$$\mathbf{R2_T}(P) = P$$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(P) & \{\text{Definition of } \mathbf{R2_T}\} \\
= (P)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \{\text{Assumption and substitution}\} \\
= P &
\end{array}
$$

□

Lemma L.2.3.13 $\mathbf{R2_T}(Expands_A(tr_A, tr'_A)) = true$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(Expands_A(tr_A, tr'_A)) & \{\text{Definition of } Expands_A\}
\end{array}
$$

$$= \mathbf{R2_T} \left(\begin{array}{l} front(tr_A) < tr'_A \\ \wedge \\ fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \end{array} \right)$$

{Definition of $\mathbf{R2_T}$ and substitution}

$$= \left(\begin{array}{l} front(\langle(\langle\rangle, fst \circ last(tr_A))\rangle) < dif_T(tr'_A, tr_A) \\ \wedge \\ fst \circ last(\langle(\langle\rangle, fst \circ last(tr_A))\rangle) \leq fst \circ head(dif_T(tr'_A, tr_A) - front(\langle(\langle\rangle, fst \circ last(tr_A))\rangle)) \end{array} \right)$$

{Property of sequences}

$$= \left(\begin{array}{l} \langle\rangle < dif_T(tr'_A, tr_A) \\ \wedge \\ fst(\langle\rangle, fst \circ last(tr_A)) \leq fst \circ head(dif_T(tr'_A, tr_A) - \langle\rangle) \end{array} \right)$$

{Property of sequences}

$= fst(\langle\rangle, fst \circ last(tr_A)) \leq fst \circ head(dif_T(tr'_A, tr_A))$

{Definition of *fst* and property of sequences}

$= true$

□

Lemma L.2.3.14 *Provided $Expands_A(tr'_A, tr_A)$ holds,*

$$\mathbf{R2_T} \circ \mathbf{R1_T}(P) = \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R2_T} \circ \mathbf{R1_T}(P)$ {Definition of $\mathbf{R1_T}$}

$= \mathbf{R2_T}(P \wedge Expands_A(tr'_A, tr_A))$ {Lemma L.2.3.18}

$= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(Expands_A(tr'_A, tr_A))$ {Lemma L.2.3.13 and predicate calculus}

$= \mathbf{R2_T}(P)$

□

Lemma L.2.3.15 *Provided tr_A is not free in P,*

$$\mathbf{R2_T}(P) = P[dif_T(tr'_A, tr_A)/tr'_A]$$

Proof.

$$\begin{aligned}
&\mathbf{R2_T}(P) && \{\text{Definition of } \mathbf{R2_T}\}\\
&= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] && \{\text{Assumption and substitution}\}\\
&= P[dif_T(tr'_A, tr_A)/tr'_A]
\end{aligned}$$

□

Lemma L.2.3.16 *Provided v is not tr'_A nor tr_A,*

$$\mathbf{R2_T}(\exists\, v \bullet P) = \exists\, v \bullet \mathbf{R2_T}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{R2_T}(\exists\, v \bullet P) && \{\text{Definition of } \mathbf{R2_T}\}\\
&= (\exists\, v \bullet P)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] \\
& && \{\text{Assumption and predicate calculus}\}\\
&= \exists\, v \bullet P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \exists\, v \bullet \mathbf{R2_T}(P)
\end{aligned}$$

□

Lemma L.2.3.17 *Provided tr_A is not free in P,*

$$\mathbf{R2_T}(P[dif_T(tr'_A, tr_A)/tr'_A]) = P[dif_T(tr'_A, tr_A)]$$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(P[dif_T(tr'_A, tr_A)/tr'_A]) & \{\text{Definition of } \mathbf{R2_T}\} \\
= P[dif_T(tr'_A, tr_A)/tr'_A][\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \\
& \{\text{Assumption and substitution}\} \\
= P[dif_T(dif_T(tr'_A, tr_A), \langle(\langle\rangle, snd \circ last(tr_A))\rangle)/tr'_A] & \{\text{Lemma L.2.3.7}\} \\
= P[dif_T(tr'_A, tr_A)/tr'_A] &
\end{array}
$$

□

Lemma L.2.3.18 $\mathbf{R2_T}(P \wedge Q) = \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(Q)$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(P \wedge Q) & \{\text{Definition of } \mathbf{R2_T}\} \\
= (P \wedge Q)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \{\text{Substitution}\} \\
= \left(\begin{array}{l} P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] \\ \wedge \\ Q[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] \end{array} \right) & \{\text{Definition of } \mathbf{R2_T}\} \\
= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(Q) &
\end{array}
$$

□

Lemma L.2.3.19 $\mathbf{R2_T}(P \vee Q) = \mathbf{R2_T}(P) \vee \mathbf{R2_T}(Q)$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T}(P \vee Q) & \{\text{Definition of } \mathbf{R2_T}\} \\
= (P \vee Q)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] & \{\text{Substitution}\} \\
= \left(\begin{array}{l} P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] \\ \vee \\ Q[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A] \end{array} \right) & \{\text{Definition of } \mathbf{R2_T}\} \\
= \mathbf{R2_T}(P) \vee \mathbf{R2_T}(Q) &
\end{array}
$$

□

Lemma L.2.3.20 *Provided tr'_A and tr_A are not free in Q,*

$$\mathbf{R2_T}(P \lhd Q \rhd R) = \mathbf{R2_T}(P) \lhd Q \rhd \mathbf{R2_T}(Q)$$

Proof.

$$\begin{aligned}
&\mathbf{R2_T}(P \lhd Q \rhd R) && \{\text{Definition of conditional}\}\\
&= \mathbf{R2_T}((Q \wedge P) \vee (\neg\, Q \wedge R)) && \{\text{Lemma L.2.3.19}\}\\
&= \mathbf{R2_T}(Q \wedge P) \vee \mathbf{R2_T}(\neg\, Q \wedge R) && \{\text{Lemma L.2.3.18}\}\\
&= (\mathbf{R2_T}(Q) \wedge \mathbf{R2_T}(P)) \vee (\mathbf{R2_T}(\neg\, Q) \wedge \mathbf{R2_T}(R)) \\
& && \{\text{Assumption and Lemma L.2.3.12}\}\\
&= (Q \wedge \mathbf{R2_T}(P)) \vee (\neg\, Q \wedge \mathbf{R2_T}(R)) && \{\text{Definition of conditional}\}\\
&= \mathbf{R2_T}(P) \lhd Q \rhd \mathbf{R2_T}(R)
\end{aligned}$$

□

Lemma L.2.3.21 *Provided v is not tr'_A nor tr_A, $\exists\, v \bullet \mathbf{R2_T}(P) = \mathbf{R2_T}(\exists\, v \bullet P)$*

Proof.

$$\begin{aligned}
&\exists\, v \bullet \mathbf{R2_T}(P) && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \exists\, v \bullet (P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A]) \\
& && \{\text{Assumption and predicate calculus}\}\\
&= (\exists\, v \bullet P)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A]
\end{aligned}$$

□

Lemma L.2.3.22

$$\begin{aligned}
&\mathbf{R2_T}(P)[front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle/tr_A]\\
&=\\
&P[\langle(\langle\rangle, ref)\rangle/tr_A, dif_T(tr'_A, tr_A)/tr'_A]
\end{aligned}$$

Proof.

$$\mathbf{R2_T}(P)[front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle/tr_A] \quad \{\text{Definition of } \mathbf{R2_T}\}$$

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A][front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle/tr_A]$

{Substitution}

$$= P \left[\begin{array}{l} \langle(\langle\rangle, snd \circ last(front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle))\rangle/tr_A \\ dif_T(tr'_A, front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)/tr'_A \end{array} \right]$$

{Property of sequences: $last(s \frown \langle e\rangle) = \langle e\rangle$}

$$= P \left[\begin{array}{l} \langle(\langle\rangle, ref)\rangle/tr_A \\ dif_T(tr'_A, front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)/tr'_A \end{array} \right]$$

{Lemma L.2.3.26}

$= P[\langle(\langle\rangle, ref)\rangle/tr_A, dif_T(tr'_A, tr_A)/tr'_A]$

□

Lemma L.2.3.23 *Provided tr'_A and tr_A are not free in P,*

$$\mathbf{R1_T} \circ \mathbf{R2_T}(P) = P \wedge \mathbf{R1_T}(true)$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Definition of $\mathbf{R1_T}$}

$= \mathbf{R2_T}(P) \wedge Expands_A(tr_A, tr'_A)$ {Assumption and definition of $\mathbf{R2_T}$}

$= P \wedge Expands_A(tr_A, tr'_A)$ {Definition of $\mathbf{R1_T}$ and predicate calculus}

$= P \wedge \mathbf{R1_T}(true)$

□

Lemma L.2.3.24 *Provided tr'_A and tr_A are not free in P,*

$$\mathbf{R1_T} \circ \mathbf{R2_T}(P \vee Q) = (P \wedge \mathbf{R1_T}(true)) \vee \mathbf{R1_T} \circ \mathbf{R2_T}(Q)$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R2_T}(P \vee Q)$ {Lemmas L.2.2.6 and L.2.3.19}

$= \mathbf{R1_T} \circ \mathbf{R2_T}(P) \vee \mathbf{R1_T} \circ \mathbf{R2_T}(Q)$ {Assumption and Lemma L.2.3.23}

$= (P \wedge \mathbf{R1_T}(true)) \vee \mathbf{R1_T} \circ \mathbf{R2_T}(Q)$

□

Lemma L.2.3.25

$$\mathbf{R2_T}(P) = \mathbf{R2_T}(P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle / tr_A])$$

Proof.

$\mathbf{R2_T}(P)$ {Definition of $\mathbf{R2_T}$}

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A]$ {Property of substitution}

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][dif_T(tr'_A, tr_A)/tr'_A]$ {Lemma L.2.3.27}

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][dif_T(tr'_A, tr_A)/tr'_A]$ {Property of substitution}

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A]$ {Definition of $\mathbf{R2_T}$}

$= \mathbf{R2_T}(P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A])$

□

Lemma L.2.3.26

$$dif_T(tr'_A, front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle) = dif_T(tr'_A, tr_A)$$

Proof.

$dif_T(tr'_A, front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)$ {Definition of dif_T}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(tr'_A - front(front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)) \\ - \\ fst \circ last(front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle) \end{array} \right), \\ snd \circ head(tr'_A - front(front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(front(tr_A) \frown \langle(fst \circ last(tr_A), ref)\rangle)) \end{array} \right)$$

{Property of sequences: $front(front(s) \frown \langle e\rangle) = front(s)$}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) \\ - \\ fst \circ last(front(tr_A) \frown \langle (fst \circ last(tr_A), ref) \rangle) \end{array} \right), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(tr_A)) \end{array} \right)$$

{Property of sequences: $last(front(s) \frown \langle e \rangle) = e$}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} (fst \circ head(tr'_A - front(tr_A)) - fst(fst \circ last(tr_A), ref)), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(tr_A)) \end{array} \right)$$

{Definition of *fst*}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} (fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A)), \\ snd \circ head(tr'_A - front(tr_A)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_A - front(tr_A)) \end{array} \right)$$

{Definition of dif_T}

$$= dif_T(tr'_A, tr_A)$$

□

Lemma L.2.3.27

$$P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A]$$
$$=$$
$$P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A]$$

Proof.

$P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A][\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A]$ {Substitution}

$= P[\langle(\langle\rangle, snd \circ last(\langle(\langle\rangle, snd \circ last(tr_A))\rangle))\rangle/tr_A]$ {Definition of *last*}

$= P[\langle(\langle\rangle, snd(\langle\rangle, snd \circ last(tr_A)))\rangle/tr_A]$ {Definition of *snd*}

$= P[\langle(\langle\rangle, snd \circ last(tr_A))\rangle/tr_A]$

□

Lemma L.2.3.28

$$\mathbf{R2_T}(P) \Leftrightarrow \mathbf{R2_T}(Q) = \mathbf{R2_T}(P \Leftrightarrow Q)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_T}(P) \Leftrightarrow \mathbf{R2_T}(Q) & \{\text{Predicate calculus}\} \\
= (\mathbf{R2_T}(P) \Rightarrow \mathbf{R2_T}(Q)) \wedge (\mathbf{R2_T}(Q) \Rightarrow \mathbf{R2_T}(P)) & \{\text{Predicate calculus}\} \\
= (\neg\, \mathbf{R2_T}(P) \vee \mathbf{R2_T}(Q)) \wedge (\neg\, \mathbf{R2_T}(Q) \vee \mathbf{R2_T}(P)) & \{\text{Lemma L.2.3.11}\} \\
= (\mathbf{R2_T}(\neg\, P) \vee \mathbf{R2_T}(Q)) \wedge (\mathbf{R2_T}(\neg\, Q) \vee \mathbf{R2_T}(P)) & \{\text{Lemma L.2.3.19}\} \\
= \mathbf{R2_T}(\neg\, P \vee Q) \wedge \mathbf{R2_T}(\neg\, Q \vee P) & \{\text{Lemma L.2.3.18}\} \\
= \mathbf{R2_T}((\neg\, P \vee Q) \wedge (\neg\, Q \vee P)) & \{\text{Predicate calculus}\} \\
= \mathbf{R2_T}((P \Rightarrow Q) \wedge (Q \Rightarrow P)) & \{\text{Predicate calculus}\} \\
= \mathbf{R2_T}(P \Leftrightarrow Q)
\end{array}$$

□

Lemma L.2.3.29

$$\begin{array}{l}
\mathbf{R2_T}(P) \\
= \\
\exists\, tr0_A, tr0'_A \bullet \left(\begin{array}{l} P[tr0_A, tr0'_A / tr_A, tr'_A] \\ \wedge \\ tr0_A = \langle (\langle\rangle, snd \circ last(tr_A)) \rangle \wedge tr0'_A = dif_T(tr'_A, tr_A) \end{array} \right)
\end{array}$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_T}(P) & \{\text{Definition of } \mathbf{R2_T}\} \\
= P[\langle (\langle\rangle, snd \circ last(tr_A)) \rangle, dif_T(tr'_A, tr_A) / tr_A, tr'_A] & \{\text{Predicate calculus}\} \\
= P[tr0_A, tr0'_A / tr_A, tr'_A][\langle (\langle\rangle, snd \circ last(tr_A)) \rangle, dif_T(tr'_A, tr_A) / tr0_A, tr0'_A] & \\
& \{\text{Predicate calculus}\}
\end{array}$$

$$= \exists\, tr0_A, tr0'_A \bullet \left(\begin{array}{l} P[tr0_A, tr0'_A / tr_A, tr'_A] \\ \wedge \\ tr0_A = \langle (\langle\rangle, snd \circ last(tr_A)) \rangle \wedge tr0'_A = dif_T(tr'_A, tr_A) \end{array} \right)$$

□

2.4 Results on $\mathbf{R3_T}$

Lemma L.2.4.1

$$\mathbf{R3_T}(P) = \mathbf{R3_T}(P_f)$$

Proof.

$\mathbf{R3_T}(P)$	{Definition of $\mathbf{R3_T}$}
$= \mathit{II}_A \lhd wait_T \rhd P$	{Property of conditional}
$= \mathit{II}_A \lhd wait_T \rhd (P \wedge \neg\, wait_T)$	{Leibiniz's substitution}
$= \mathit{II}_A \lhd wait_T \rhd (P[false/wait_T] \wedge \neg\, wait_T)$	{Property of conditional}
$= \mathit{II}_A \lhd wait_T \rhd P[false/wait_T]$	{Definition of $\mathbf{R3_T}$}
$= \mathbf{R3_T}(P_f)$	

□

Lemma L.2.4.2

$$\mathbf{R3_T}(P)^o_f = (\mathit{II}_A)^o_f \lhd wait_T \rhd P^o_f$$

Proof.

$\mathbf{R3_T}(P)^o_f$	{Definition of $\mathbf{R3_T}$}
$= (\mathit{II}_A \lhd wait_T \rhd P)^o_f$	{Substitution}
$= (\mathit{II}_A)^o_f \lhd wait_T \rhd P^o_f$	

□

Lemma L.2.4.3

$$(\mathit{II}_A)^o_f = \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T}(true)) \\ \vee \\ (o \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right)$$

Proof.

$(\mathit{II}_A)^o_f$ {Definition of II_A}

$= \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T}(true)) \\ \vee \\ (ok' \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right)^o_f$ {Substitution}

$= \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T}(true)) \\ \vee \\ (o \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right)$

□

Lemma L.2.4.4

$$\mathbf{R2} \circ \mathbf{R3_T}(P) = \mathbf{R3_T} \circ \mathbf{R2}(P)$$

Proof.

$\mathbf{R2} \circ \mathbf{R3_T}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R2}(\mathit{II}_A \lhd wait_T \rhd P)$ {Definition of conditional and distributivity of $\mathbf{R2}$}

$= \mathbf{R2}(\mathit{II}_A) \lhd wait_T \rhd \mathbf{R2}(P)$ {tr and tr' not free in II_A}

$= \mathit{II}_A \lhd wait_T \rhd \mathbf{R2}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R3_T} \circ \mathbf{R2}(P)$

□

Lemma L.2.4.5 $\mathbf{R3_T}(P \wedge Q) = \mathbf{R3_T}(P) \wedge \mathbf{R3_T}(Q)$

Proof.

$\mathbf{R3_T}(P \wedge Q)$ {Definition of $\mathbf{R3_T}$}

$= \mathit{II} \lhd \mathit{wait} \rhd (P \wedge Q)$ {Predicate calculus}

$= (\mathit{II} \wedge \mathit{II}) \lhd \mathit{wait} \rhd (P \wedge Q)$ {Property of conditional}

$= (\mathit{II} \lhd \mathit{wait} \rhd P) \wedge (\mathit{II} \lhd \mathit{wait} \rhd Q)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R3_T}(P) \wedge \mathbf{R3_T}(Q)$

□

Lemma L.2.4.6

$$\mathbf{R1_T} \circ \mathbf{R3_T}(P) = \mathbf{R3_T} \circ \mathbf{R1_T}(P)$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R3_T}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R1_T}(\mathit{II}_A \lhd \mathit{wait} \rhd P)$ {Lemma L.2.2.8}

$= \mathbf{R1_T}(\mathit{II}_A) \lhd \mathit{wait} \rhd \mathbf{R1_T}(P)$ {Lemma L.2.4.8}

$= \mathit{II}_A \lhd \mathit{wait} \rhd \mathbf{R1_T}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R3_T} \circ \mathbf{R1_T}(P)$

□

Lemma L.2.4.7

$$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R3_T}(P) = \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R3_T}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R1_T} \circ \mathbf{R2_T}(\mathit{II}_A \lhd \mathit{wait} \rhd P)$ {Lemma L.2.3.20}

$= \mathbf{R1_T}(\mathbf{R2_T}(\mathit{II}_A) \lhd \mathit{wait} \rhd \mathbf{R2_T}(P))$ {Lemma L.2.2.9}

$= \mathbf{R1_T} \circ \mathbf{R2_T}(\mathit{II}_A) \lhd \mathit{wait} \rhd \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Lemma L.2.4.9}

$= \mathit{II}_A \lhd \mathit{wait} \rhd \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$

□

Lemma L.2.4.8

$$\mathbf{R1_T}(I\!I_A) = I\!I_A$$

Proof.

$\mathbf{R1_T}(I\!I_A)$ {Definition of $I\!I_A$}

$= \mathbf{R1_T}((\neg\ ok \wedge Expands_A) \vee (ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state))$
{Lemma L.2.2.6}

$$= \left(\begin{array}{l} \mathbf{R1_T}(\neg\ ok \wedge Expands_A) \\ \vee \\ \mathbf{R1_T}(ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$
{Predicate calculus and Lemma L.2.2.5}

$$= \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T}(Expands_A)) \\ \vee \\ (\mathbf{R1_T}(true) \wedge ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$
{Definition of $\mathbf{R1_T}$ and predicate calculus}

$$= \left(\begin{array}{l} (\neg\ ok \wedge Expands_A) \\ \vee \\ (\mathbf{R1_T}(true) \wedge ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$
{Definition of $\mathbf{R1_T}$, Lemma L.2.2.1 and predicate calculus}

$$= \left(\begin{array}{l} (\neg\ ok \wedge Expands_A) \\ \vee \\ (ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

□

Lemma L.2.4.9

$$\mathbf{R1_T} \circ \mathbf{R2_T}(I\!I_A) = I\!I_A$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R2_T}(I\!I_A)$ {Definition of $I\!I_A$}

$$= \mathbf{R1_T} \circ \mathbf{R2_T} \left(\begin{array}{l} (\neg\ ok \wedge Expands_A(tr_A, tr'_A)) \\ \vee \\ (ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.3.19}

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(\neg\ ok \wedge Expands_A(tr_A, tr'_A)) \\ \vee \\ \mathbf{R2_T}(ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemmas L.2.3.12 and L.2.3.18}

$$= \mathbf{R1_T} \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R2_T}(Expands_A(tr_A, tr'_A))) \\ \vee \\ (ok' \wedge \mathbf{R2_T}(tr'_A = tr_A) \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Definition of $\mathbf{R1_T}$}

$$= \mathbf{R1_T} \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R2_T} \circ \mathbf{R1_T}(true)) \\ \vee \\ (ok' \wedge \mathbf{R2_T}(tr'_A = tr_A) \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.3.14}

$$= \mathbf{R1_T} \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R2_T}(true)) \\ \vee \\ (ok' \wedge \mathbf{R2_T}(tr'_A = tr_A) \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.2.6}

$$= \left(\begin{array}{l} \mathbf{R1_T}(\neg\ ok \wedge \mathbf{R2_T}(true)) \\ \vee \\ \mathbf{R1_T}(ok' \wedge \mathbf{R2_T}(tr'_A = tr_A) \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.2.5}

$$= \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T} \circ \mathbf{R2_T}(true)) \\ \vee \\ (ok' \wedge \mathbf{R1_T} \circ \mathbf{R2_T}(tr'_A = tr_A) \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.4.10}

$$= \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T} \circ \mathbf{R2_T}(true)) \\ \vee \\ (ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right)$$

{Lemma L.2.3.12 and definition of $\mathbf{R1_T}$}

$$= \left(\begin{array}{l} (\neg\ ok \wedge \mathit{Expands}_A(tr_A, tr'_A)) \\ \vee \\ (ok' \wedge tr'_A = tr_A \wedge wait' = wait \wedge state' = state) \end{array} \right) \quad \{\text{Definition of } \mathit{II}_A\}$$

$$= \mathit{II}_A$$

□

Lemma L.2.4.10

$$\mathbf{R1_T} \circ \mathbf{R2_T}(tr'_A = tr_A) = tr'_A = tr_A$$

Proof.

$\mathbf{R1_T} \circ \mathbf{R2_T}(tr'_A = tr_A)$ {Definition of $\mathbf{R2_T}$}

$= \mathbf{R1_T}((tr'_A = tr_A)[\langle(\langle\rangle, snd \circ last(tr_A))\rangle, dif_T(tr'_A, tr_A)/tr_A, tr'_A])$ {Substitution}

$= \mathbf{R1_T}(dif_T(tr'_A, tr_A) = \langle(\langle\rangle, snd \circ last(tr_A))\rangle))$

{Lemma L.2.3.6 and property of sequences}

$= \mathbf{R1_T}(tr'_A = tr_A)$ {Predicate calculus and Lemma L.2.2.5}

$= \mathbf{R1_T}(true) \wedge tr'_A = tr_A$ {Definition of $\mathbf{R1_T}$ and Lemma L.2.2.1}

$= tr'_A = tr_A$

□

Lemma L.2.4.11 $\mathbf{R3_T}(P) = \mathbf{R3_T}(Q) \Leftrightarrow P_f = Q_f$

Proof.

$\mathbf{R3_T}(P) = \mathbf{R3_T}(Q)$ {Definition of $\mathbf{R3_T}$}

$\Leftrightarrow (\mathit{II}_A \lhd wait \rhd P) = (\mathit{II}_A \lhd wait \rhd Q)$ {Definition of conditional}

$\Leftrightarrow ((wait \wedge \mathit{II}_A) \vee (\neg\ wait \wedge P)) = ((wait \wedge \mathit{II}_A) \vee (\neg\ wait \wedge Q))$ {Equality}

$\Leftrightarrow [((wait \wedge \mathit{II}_A) \vee (\neg\ wait \wedge P)) \Leftrightarrow ((wait \wedge \mathit{II}_A) \vee (\neg\ wait \wedge Q))]$

{Lemma L.8.0.7}

$\Leftrightarrow [(wait \wedge \mathit{II}_A) \vee ((\neg\ wait \wedge P) \Leftrightarrow (\neg\ wait \wedge Q))]$ {Lemma L.8.0.9}

$\Leftrightarrow [(wait \wedge \mathit{II}_A) \vee (\neg\ wait \Rightarrow (P \Leftrightarrow Q))]$ {Predicate calculus}

$\Leftrightarrow [(wait \wedge \mathit{II}_A) \vee wait \vee (P \Leftrightarrow Q)]$ {Predicate calculus: absorption law}

$\Leftrightarrow [wait \vee (P \Leftrightarrow Q)]$ {Predicate calculus}

$\Leftrightarrow [\neg\ wait \Rightarrow (P \Leftrightarrow Q)]$ {Universal quantification}

$\Leftrightarrow [\forall\ wait \bullet \neg\ wait \Rightarrow (P \Leftrightarrow Q)]$ {Instantiation}

$\Leftrightarrow [(\neg\ wait \Rightarrow (P \Leftrightarrow Q))[true/wait] \wedge (\neg\ wait \Rightarrow (P \Leftrightarrow Q))[false/wait]]$ {Substitution}

$\Leftrightarrow [(\neg\ true \Rightarrow (P \Leftrightarrow Q)[true/wait]) \wedge (\neg\ false \Rightarrow (P \Leftrightarrow Q)[false/wait])]$ {Predicate calculus}

$\Leftrightarrow [(P \Leftrightarrow Q)_f]$ {Substitution}

$\Leftrightarrow [P_f \Leftrightarrow Q_f]$ {Definition of equality}

$\Leftrightarrow P_f = Q_f$

□

2.5 Results on $\mathbf{R_T}$

Lemma L.2.5.1

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T}(P) = \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T}(P)$ {Definition of $\mathbf{R0_T}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P \wedge \#tr_T \leq \#tr'_T \wedge \#tr_T > 0)$ {Distributivity of $\mathbf{R2_T}$ (Lemma L.2.3.18)}

$= \mathbf{R0_T} \circ \mathbf{R1_T}(\mathbf{R2_T}(P) \wedge \mathbf{R2_T}(\#tr_T \leq \#tr'_T) \wedge \mathbf{R2_T}(\#tr_T > 0))$ {Definition of $\mathbf{R2_T}$ and substitution}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ \#(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) \leq \#(dif_T(tr'_T, tr_T)) \\ \wedge \\ \#(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) > 0 \end{array} \right)$$

{Property of sequences and Lemma 26}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ 1 \leq \#tr'_T - \#tr_T + 1 \\ \wedge \\ 1 > 0 \end{array} \right)$$

{Arithmetic under assumption of $\mathbf{R0_T}$ and $\mathbf{R1_T}$ ($\#tr_T \leq \#tr'_T$)}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

□

Lemma L.2.5.2 *Provided v is not tr_T nor tr'_T,*

$$\exists\, v \bullet \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(\exists\, v \bullet P)$$

Proof.

$\exists\, v \bullet \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Assumption and Lemma L.2.1.3}

$= \mathbf{R0_T}(\exists\, v \bullet \mathbf{R1_T} \circ \mathbf{R2_T}(P))$ {Assumption and Lemma L.2.2.26}

$= \mathbf{R0_T} \circ \mathbf{R1_T}(\exists\, v \bullet \mathbf{R2_T}(P))$ {Assumption and Lemma L.2.3.4}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(\exists\, v \bullet P)$

□

Lemma L.2.5.3

$$\mathbf{R_T}(\neg\, \mathbf{R1_T} \circ \mathbf{R2_T}(P) \wedge \neg\, Q \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T))$$
$$=$$

$$\mathbf{R_T}(\neg\ P \wedge \neg\ Q \vdash R \wedge (S \vee T))$$

Proof.

$$\begin{aligned}
&\mathbf{R_T}(\neg\ \mathbf{R1_T} \circ \mathbf{R2_T}(P) \wedge \neg\ Q \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) && \{\text{Definition of } \mathbf{R_T}\}\\
&= \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R3_T}(\neg\ \mathbf{R1_T} \circ \mathbf{R2_T}(P) \wedge \neg\ Q \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Commutativity of } \mathbf{R1_T}, \mathbf{R2_T} \text{ and } \mathbf{R3_T}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ \mathbf{R1_T} \circ \mathbf{R2_T}(P) \wedge \neg\ Q \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Predicate calculus}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ (\mathbf{R1_T} \circ \mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Lemma 4}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ \mathbf{R1_T}(\mathbf{R1_T} \circ \mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Lemma L.2.2.6 and } \mathbf{R1_T}\text{-idempotent}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ \mathbf{R1_T}(\mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Lemma 4}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ (\mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T)) && \{\text{Lemma 6}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ (\mathbf{R2_T}(P) \vee Q) \vdash \mathbf{RA1}(R \wedge (\mathbf{R1_T} \circ \mathbf{R2_T}(S) \vee T))) \\
& && \{\text{Lemmas L.2.2.4 and L.2.2.6 and } \mathbf{R1_T}\text{-idempotent}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ (\mathbf{R2_T}(P) \vee Q) \vdash \mathbf{RA1}(R \wedge (\mathbf{R2_T}(S) \vee T))) && \{\text{Lemma 6}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T}(\neg\ (\mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R2_T}(S) \vee T)) \\
& && \{\text{Commutativity of } \mathbf{R1_T} \text{ and } \mathbf{R2_T}\}\\
&= \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(\neg\ (\mathbf{R2_T}(P) \vee Q) \vdash R \wedge (\mathbf{R2_T}(S) \vee T)) && \{\text{Law 10}\}\\
&= \mathbf{R3_T} \circ \mathbf{R1_T}(\neg\ \mathbf{R2_T}(\mathbf{R2_T}(P) \vee Q) \vdash \mathbf{R2_T}(R \wedge (\mathbf{R2_T}(S) \vee T))) \\
& && \{\text{Lemmas L.2.3.18 and L.2.3.19 and } \mathbf{R2_T}\text{-idempotent}\}\\
&= \mathbf{R3_T} \circ \mathbf{R1_T}(\neg\ \mathbf{R2_T}(P \vee Q) \vdash \mathbf{R2_T}(R \wedge (S \vee T))) && \{\text{Law 10}\}\\
&= \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(\neg\ (P \vee Q) \vdash R \wedge (S \vee T)) \\
& && \{\text{Commutativity of } \mathbf{R1_T}, \mathbf{R2_T} \text{ and } \mathbf{R3_T} \text{ and definition of } \mathbf{R_T}\}\\
&= \mathbf{R_T}(\neg\ (P \vee Q) \vdash R \wedge (S \vee T)) && \{\text{Predicate calculus}\}\\
&= \mathbf{R_T}(\neg\ P \wedge \neg\ Q \vdash R \wedge (S \vee T))
\end{aligned}$$

□

Lemma L.2.5.4

$$\mathbf{R_T}(P \vdash Q) = \mathbf{R_T}(R \vdash S)$$
$$\Leftrightarrow$$
$$[Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))]$$

Proof.

$\mathbf{R_T}(P \vdash Q) = \mathbf{R_T}(R \vdash S)$

$$\{\text{Definition of } \mathbf{R_T} \text{ (and commutativity of healthiness conditions)}\}$$
$$\Leftrightarrow \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P \vdash Q) = \mathbf{R3_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(R \vdash S) \quad \{\text{Lemma L.2.4.11}\}$$
$$\Leftrightarrow (\mathbf{R1_T} \circ \mathbf{R2_T}(P \vdash Q))_f = (\mathbf{R1_T} \circ \mathbf{R2_T}(R \vdash S))_f$$
$$\{\text{Substitution and definition of } \mathbf{R1_T} \text{ and } \mathbf{R2_T}\}$$
$$\Leftrightarrow (\mathbf{R1_T} \circ \mathbf{R2_T}(P_f \vdash Q_f)) = (\mathbf{R1_T} \circ \mathbf{R2_T}(R_f \vdash S_f)) \quad \{\text{Lemma L.2.2.20}\}$$
$$\Leftrightarrow [Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}(P_f \vdash Q_f)) \Leftrightarrow \mathbf{R2_T}(R_f \vdash S_f))] \quad \{\text{Lemma L.2.3.28}\}$$
$$\Leftrightarrow [Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f))] \quad \{\text{Universal quantification}\}$$
$$\Leftrightarrow [\forall\, ok \bullet Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f))]$$
$$\{\text{Case-analysis on } ok\}$$
$$\Leftrightarrow \left[\begin{array}{l} (Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f)))[true/ok] \\ \wedge \\ (Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f)))[false/ok] \end{array}\right]$$
$$\{\text{Substitution: } ok \text{ not free in } Expands_A\}$$
$$\Leftrightarrow \left[\begin{array}{l} (Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f))[true/ok]) \\ \wedge \\ (Expands_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \vdash Q_f) \Leftrightarrow (R_f \vdash S_f))[false/ok]) \end{array}\right]$$
$$\{\text{Definition of design}\}$$

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} ((ok \wedge P_f) \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ ((ok \wedge R_f) \Rightarrow (S_f \wedge ok')) \end{array} \right) [true/ok] \right) \\ \wedge \\ \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} ((ok \wedge P_f) \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ ((ok \wedge R_f) \Rightarrow (S_f \wedge ok')) \end{array} \right) [false/ok] \right) \end{array} \right]$$

{Substitution: definition of $\mathbf{R2_T}$ and assumption}

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} ((true \wedge P_f) \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ ((true \wedge R_f) \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) \\ \wedge \\ \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} ((false \wedge P_f) \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ ((false \wedge R_f) \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) \end{array} \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) \\ \wedge \\ (\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}(true)) \end{array} \right]$$

{Lemma L.2.3.12 and predicate calculus}

$$\Leftrightarrow \left[\left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) \right]$$

{Universal quantification}

$$\Leftrightarrow \left[\forall\, ok' \bullet \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) \right]$$

{Case-analysis on ok'}

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) [\mathit{false}/ok'] \\ \wedge \\ \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) \right) [\mathit{true}/ok'] \end{array} \right]$$

{Substitution: ok' not free in $\mathit{Expands}_A$}

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) [\mathit{false}/ok'] \right) \\ \wedge \\ \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge ok')) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge ok')) \end{array} \right) [\mathit{true}/ok'] \right) \end{array} \right]$$

{Substitution: definition of $\mathbf{R2_T}$ and assumption}

$$\Leftrightarrow \left[\begin{array}{l} \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge \mathit{false})) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge \mathit{false})) \end{array} \right) \right) \\ \wedge \\ \left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Rightarrow (Q_f \wedge \mathit{true})) \\ \Leftrightarrow \\ (R_f \Rightarrow (S_f \wedge \mathit{true})) \end{array} \right) \right) \end{array} \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\begin{array}{l} (\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}(\neg\, P_f \Leftrightarrow \neg\, R_f)) \\ \wedge \\ (\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \Rightarrow Q_f) \Leftrightarrow (R_f \Rightarrow S_f))) \end{array} \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \left(\begin{array}{l} \mathbf{R2_T}(\neg\, P_f \Leftrightarrow \neg\, R_f) \\ \wedge \\ \mathbf{R2_T}((P_f \Rightarrow Q_f) \Leftrightarrow (R_f \Rightarrow S_f)) \end{array} \right) \right) \right]$$

{Lemma L.2.3.18}

$$\Leftrightarrow \left[\left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (\neg\ P_f \Leftrightarrow \neg\ R_f) \\ \wedge \\ ((P_f \Rightarrow Q_f) \Leftrightarrow (R_f \Rightarrow S_f)) \end{array} \right) \right) \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Leftrightarrow R_f) \\ \wedge \\ ((P_f \Rightarrow Q_f) \Leftrightarrow (R_f \Rightarrow S_f)) \end{array} \right) \right) \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\left(\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T} \left(\begin{array}{l} (P_f \Leftrightarrow R_f) \\ \wedge \\ ((P_f \Rightarrow Q_f) \Leftrightarrow (P_f \Rightarrow S_f)) \end{array} \right) \right) \right]$$

{Lemma L.8.0.8}

$$\Leftrightarrow [\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))]$$

□

Lemma L.2.5.5

$$\begin{array}{l} \mathbf{R_T}(P \vdash Q) = \mathbf{R_T}(R \vdash S) \\ \Leftrightarrow \\ \left[\forall\, s : \mathrm{seq}_1 \bullet \left(\begin{array}{l} \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(s) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))[\mathit{front}(tr_A) \frown s / tr'_A] \end{array} \right) \right] \end{array}$$

Proof.

$\mathbf{R_T}(P \vdash Q) = \mathbf{R_T}(R \vdash S)$ {Lemma L.2.5.4}

$\Leftrightarrow [\mathit{Expands}_A(tr_A, tr'_A) \Rightarrow \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))]$

{Definition of $\mathit{Expands}_A$}

$$\Leftrightarrow \left[\begin{array}{l} (\mathit{front}(tr_A) < tr'_A \wedge \mathit{fst} \circ \mathit{last}(tr_A) \leq \mathit{fst} \circ \mathit{head}(tr'_A - \mathit{front}(tr_A))) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right]$$

{Universal quantification}

$$\Leftrightarrow \left[\forall\, tr'_A \bullet \left(\begin{array}{l} (front(tr_A) < tr'_A \wedge fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A))) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right) \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\forall\, tr'_A \bullet \left(front(tr_A) < tr'_A \Rightarrow \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right) \right) \right]$$

{Property of sequences}

$$\Leftrightarrow \left[\forall\, tr'_A \bullet \left(\begin{array}{l} \exists\, s : \mathrm{seq}_1 \bullet front(tr_A) \frown s = tr'_A \\ \Rightarrow \\ \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right) \end{array} \right) \right]$$

{Predicate calculus}

$$\Leftrightarrow \left[\forall\, tr'_A;\ s : \mathrm{seq}_1 \bullet \left(\begin{array}{l} front(tr_A) \frown s = tr'_A \\ \Rightarrow \\ \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right) \end{array} \right) \right]$$

{One-point rule}

$$\Leftrightarrow \left[\forall\, s : \mathrm{seq}_1 \bullet \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f))) \end{array} \right) [front(tr_A) \frown s / tr'_A] \right]$$

{Substitution}

$$\Leftrightarrow \left[\forall\, s : \mathrm{seq}_1 \bullet \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(front(tr_A) \frown s - front(tr_A)) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))[front(tr_A) \frown s / tr'_A] \end{array} \right) \right]$$

{Property of sequences}

$$\Leftrightarrow \left[\forall\, s : \mathrm{seq}_1 \bullet \left(\begin{array}{l} fst \circ last(tr_A) \leq fst \circ head(s) \\ \Rightarrow \\ \mathbf{R2_T}((P_f \Leftrightarrow R_f) \wedge (P_f \Rightarrow (Q_f \Leftrightarrow S_f)))[front(tr_A) \frown s / tr'_A] \end{array} \right) \right]$$

□

Chapter 3

Healthiness Conditions of the Super-Theory

3.1 Results on TR0

Lemma L.3.1.1

$$\mathbf{TR0}(P) \sqsupseteq P$$

Proof.

$$\begin{array}{ll} \mathbf{TR0}(P) & \{\text{Definition of } \mathbf{TR0}\} \\ = P \wedge \#tr_T > 0 & \{\text{Predicate calculus}\} \\ \sqsupseteq P & \end{array}$$

□

Lemma L.3.1.2

$$\mathbf{TR0}(P)^f_f = \mathbf{TR0}(P^f_f)$$

Proof.

$$\begin{array}{ll} \mathbf{TR0}(P)^f_f & \{\text{Definition of } \mathbf{TR0}\} \\ = (P \wedge \#tr_T > 0)^f_f & \{\text{Substitution}\} \end{array}$$

$= P_f^f \wedge \#tr_T > 0$ {Definition of **TR0**}
$= \mathbf{TR0}(P_f^f)$

□

Lemma L.3.1.3

$$\mathbf{TR0}(P)_f^o = \mathbf{TR0}(P_f^o)$$

Proof.

$\mathbf{TR0}(P)_f^o$ {Definition of **TR0**}
$= (P \wedge \#tr_T > 0)_f^o$ {Substitution}
$= P_f^f \wedge \#tr_T > 0$ {Definition of **TR0**}
$= \mathbf{TR0}(P_f^o)$

□

3.2 Results on TR1

Lemma L.3.2.1

$$\mathbf{TR1}(P) \sqsupseteq P$$

Proof.

$\mathbf{TR1}(P)$ {Definition of **TR1**}
$= P \wedge \#tr_T \leq \#tr'_T$ {Predicate calculus}
$\sqsupseteq P$

□

Lemma L.3.2.2

$$\mathbf{TR1}(P)_f^f = \mathbf{TR1}(P_f^f)$$

Proof.

$$
\begin{aligned}
&\mathbf{TR1}(P)^f_f && \{\text{Definition of } \mathbf{TR1}\}\\
&= (P \wedge \#tr_T \leq \#tr'_T)^f_f && \{\text{Substitution}\}\\
&= P^f_f \wedge \#tr_T \leq \#tr'_T && \{\text{Definition of } \mathbf{TR1}\}\\
&= \mathbf{TR1}(P^f_f)
\end{aligned}
$$

□

Lemma L.3.2.3

$$\mathbf{TR1}(P)^o_f = \mathbf{TR1}(P^o_f)$$

Proof.

$$
\begin{aligned}
&\mathbf{TR1}(P)^o_f && \{\text{Definition of } \mathbf{TR1}\}\\
&= (P \wedge \#tr_T \leq \#tr'_T)^o_f && \{\text{Substitution}\}\\
&= P^o_f \wedge \#tr_T \leq \#tr'_T && \{\text{Definition of } \mathbf{TR1}\}\\
&= \mathbf{TR1}(P^o_f)
\end{aligned}
$$

□

3.3 Results on TR2

Lemma L.3.3.1

$$\mathbf{TR2}(P) \sqsupseteq P$$

Proof.

$$
\begin{aligned}
&\mathbf{TR2}(P) && \{\text{Definition of } \mathbf{TR2}\}\\
&= P \wedge \mathit{front}(tr_T) \leq tr'_T && \{\text{Predicate calculus}\}\\
&\sqsupseteq P
\end{aligned}
$$

□

3.4 Results on TR3

Lemma L.3.4.1

$$\mathbf{TR3}(P) \sqsupseteq P$$

Proof.

$$\begin{array}{ll}
\mathbf{TR3}(P) & \{\text{Definition of } \mathbf{TR3}\} \\
= P \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) & \{\text{Predicate calculus}\} \\
\sqsupseteq P &
\end{array}$$

□

Lemma L.3.4.2

$$\mathbf{TR3}(P)^f_f = \mathbf{TR3}(P^f_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{TR3}(P)^f_f & \{\text{Definition of } \mathbf{TR3}\} \\
= (P \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)))^f_f & \{\text{Substitution}\} \\
= P^f_f \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) & \{\text{Definition of } \mathbf{TR3}\} \\
= \mathbf{TR3}(P^f_f) &
\end{array}$$

□

Lemma L.3.4.3

$$\mathbf{TR3}(P \wedge ok \wedge wait) = P \wedge \#tr'_T = \#tr_T \wedge wait'_T$$

Proof.

$$\begin{array}{ll}
\mathbf{TR3}(P \wedge ok \wedge wait) & \{\text{Definition of } \mathbf{TR3}\} \\
= P \wedge (ok \wedge wait) \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) & \\
& \{\text{Predicate calculus}\}
\end{array}$$

$= P \wedge (ok \wedge wait) \wedge \#tr'_T = \#tr_T \wedge wait'_T$

□

Lemma L.3.4.4

$$\mathbf{TR3}(\neg\ ok) = \neg\ ok$$

Proof.

$\mathbf{TR3}(\neg\ ok)$ {Definition of **TR3**}
$= \neg\ ok \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T))$ {Predicate calculus}
$= \neg\ ok$

□

Lemma L.3.4.5

$$\mathbf{R2_T} \circ \mathbf{TR3}(P) = \mathbf{TR3} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R2_T} \circ \mathbf{TR3}(P)$ {Definition of **TR3**}
$= \mathbf{R2_T}(P \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)))$ {Lemma L.2.3.18}
$= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T))$ {Predicate calculus}
$= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(\neg\ (ok \wedge wait_T) \vee (\#tr'_T = \#tr_T \wedge wait'_T))$
{Lemmas L.2.3.12 and L.2.3.19}
$= \mathbf{R2_T}(P) \wedge (\neg\ (ok \wedge wait_T) \vee \mathbf{R2_T}(\#tr'_T = \#tr_T \wedge wait'_T))$
{Predicate calculus and Lemmas L.2.3.12 and L.2.3.18}
$= \mathbf{R2_T}(P) \wedge ((ok \wedge wait_T) \Rightarrow (\mathbf{R2_T}(\#tr'_T = \#tr_T) \wedge wait'_T))$
{Definition of $\mathbf{R2_T}$ and substitution}
$= \mathbf{R2_T}(P) \wedge ((ok \wedge wait_T) \Rightarrow (\#dif_T(tr'_T, tr_T) = \#\langle(\langle\rangle, snd \circ last(tr_T))\rangle \wedge wait'_T))$
{Property of sequences}
$= \mathbf{R2_T}(P) \wedge ((ok \wedge wait_T) \Rightarrow (\#dif_T(tr'_T, tr_T) = 1 \wedge wait'_T))$ {Lemma 26}
$= \mathbf{R2_T}(P) \wedge ((ok \wedge wait_T) \Rightarrow ((\#tr'_T - \#tr_T) + 1) = 1 \wedge wait'_T))$ {Arithmetic}

$= \mathbf{R2_T}(P) \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T))$ {Definition of **TR3**}
$= \mathbf{TR3} \circ \mathbf{R2_T}(P)$

□

Lemma L.3.4.6

$$\neg\, wait_T \wedge \mathbf{TR3}(P) = \neg\, wait_T \wedge P$$

Proof.

$\neg\, wait_T \wedge \mathbf{TR3}(P)$ {Definition of **TR3**}

$$= \begin{pmatrix} \neg\, wait_T \wedge P \\ \wedge \\ ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) \end{pmatrix}$$ {Predicate calculus}

$= \neg\, wait_T \wedge P$

□

3.5 Results on TR

Lemma L.3.5.1

$$0 < \#s \wedge \#s \leq \#t \wedge front(s) \leq t = 0 < \#s \wedge \#s \leq \#t \wedge front(s) < t$$

Proof. Isabelle theorem: `front_a_lt_b__resultof__length_a_lt_length_b_and_¬ front_a_lt_b`. □

Lemma L.3.5.2

$$\mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P) = P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T$$

Proof.

$\mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P)$ {Definition of **TR0**, **TR1** and **TR2**}
$= P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) \leq tr'_T$ {Lemma L.3.5.1}

$= P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T$

□

Lemma L.3.5.3

$$\mathbf{TR3} \circ \mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P)$$
$$=$$
$$P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T \wedge ((ok \wedge wait_T) \Rightarrow \#tr'_T = \#tr_T)$$

Proof.

$\mathbf{TR3} \circ \mathbf{TR2} \circ \mathbf{TR1} \circ \mathbf{TR0}(P)$ {Lemma L.3.5.2}
$= \mathbf{TR3}(P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T)$ {Definition of **TR3**}
$= P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T \wedge ((ok \wedge wait_T) \Rightarrow \#tr'_T = \#tr_T)$

□

I think the following properties should just be proved in terms of the results on conjunctive healthiness conditions and healthiness conditions similar to **R2**. They appear all the time, and their proofs are exactly the same. Perhaps can even develop a set of lemmas in Isabelle/UTP suitable for every conjunctive healthiness condition.

Lemma L.3.5.4

$$\mathbf{TR012}(P \wedge Q) = \mathbf{TR012}(P) \wedge Q$$

Proof.

$\mathbf{TR}(P \wedge Q)$ {**TR0**, **TR1** and **TR2** are conjunctive}
$= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2}(P) \wedge Q$ {Definition of **TR**}
$= \mathbf{TR}(P) \wedge Q$

□

Lemma L.3.5.5

$$\mathbf{TR}(P \wedge Q) = \mathbf{TR}(P) \wedge \mathbf{TR}(Q)$$

Proof.

$$\begin{aligned}
&\mathbf{TR}(P \wedge Q) && \{\text{Definition of } \mathbf{TR}\}\\
&= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P \wedge Q)\\
& && \{\text{Definition of } \mathbf{TR4} \text{ and property of substitution}\}\\
&= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2}(\mathbf{TR4}(P) \wedge \mathbf{TR4}(Q))\\
& && \{\mathbf{TR0}, \mathbf{TR1} \text{ and } \mathbf{TR2} \text{ are conjunctive}\}\\
&= \left(\begin{array}{l} \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P)\\ \wedge\\ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(Q) \end{array}\right) && \{\text{Definition of } \mathbf{TR}\}\\
&= \mathbf{TR}(P) \wedge \mathbf{TR}(Q)
\end{aligned}$$

□

Lemma L.3.5.6

$$\mathbf{TR}(P \vee Q) = \mathbf{TR}(P) \vee \mathbf{TR}(Q)$$

Proof.

$$\begin{aligned}
&\mathbf{TR}(P \vee Q) && \{\text{Definition of } \mathbf{TR}\}\\
&= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P \vee Q)\\
& && \{\text{Definition of } \mathbf{TR4} \text{ and property of substitution}\}\\
&= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2}(\mathbf{TR4}(P) \vee \mathbf{TR4}(Q))\\
& && \{\mathbf{TR0}, \mathbf{TR1} \text{ and } \mathbf{TR2} \text{ are conjunctive}\}\\
&= \left(\begin{array}{l} \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P)\\ \vee\\ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(Q) \end{array}\right) && \{\text{Definition of } \mathbf{TR}\}\\
&= \mathbf{TR}(P) \vee \mathbf{TR}(Q)
\end{aligned}$$

□

Lemma L.3.5.7

$$\mathbf{TR}(P \lhd Q \rhd R) = \mathbf{TR}(P) \lhd \mathbf{TR4}(Q) \rhd \mathbf{TR}(R)$$

Proof.

$$\begin{array}{ll}
\mathbf{TR}(P \lhd Q \rhd R) & \{\text{Definition of } \mathbf{TR}\} \\
= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P \lhd Q \rhd R) & \{\text{Definition of conditional}\} \\
= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}((Q \wedge P) \vee (\neg\, Q \wedge R)) & \\
& \{\text{Definition of } \mathbf{TR4} \text{ and distributivity}\} \\
= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \left(\begin{array}{l} (\mathbf{TR4}(Q) \wedge \mathbf{TR4}(P)) \\ \vee \\ (\mathbf{TR4}(\neg\, Q) \wedge \mathbf{TR4}(R)) \end{array} \right) & \\
& \{\text{Definition of } \mathbf{TR4} \text{ and property of substitution}\} \\
= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \left(\begin{array}{l} (\mathbf{TR4}(Q) \wedge \mathbf{TR4}(P)) \\ \vee \\ (\neg\, \mathbf{TR4}(Q) \wedge \mathbf{TR4}(R)) \end{array} \right) & \\
& \{\mathbf{TR0}, \mathbf{TR1} \text{ and } \mathbf{TR2} \text{ are conjunctive healthiness conditions}\} \\
= \left(\begin{array}{l} \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2}(\mathbf{TR4}(Q) \wedge \mathbf{TR4}(P)) \\ \vee \\ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2}(\neg\, \mathbf{TR4}(Q) \wedge \mathbf{TR4}(R)) \end{array} \right) & \{\text{Lemma L.3.5.4}\} \\
= \left(\begin{array}{l} (\mathbf{TR4}(Q) \wedge \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(P)) \\ \vee \\ (\neg\, \mathbf{TR4}(Q) \wedge \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR4}(R)) \end{array} \right) & \{\text{Definition of } \mathbf{TR}\} \\
= (\mathbf{TR4}(Q) \wedge \mathbf{TR}(P)) \vee (\neg\, \mathbf{TR4}(Q) \wedge \mathbf{TR}(R)) & \{\text{Definition of conditional}\} \\
= \mathbf{TR4}(P) \lhd \mathbf{TR}(Q) \rhd \mathbf{TR}(R) &
\end{array}$$

□

Lemma L.3.5.8 *Provided v is not tr_T, tr'_T, $wait_T$, $wait'_T$, $wait$, $wait'$, ok, ok', tr and tr',*

$$\mathbf{TR}(\exists\, v \bullet P) = \exists\, v \bullet \mathbf{TR}(P)$$

Proof.

$\mathbf{TR}(\exists\, v \bullet P)$ {Definition of **TR**, assumption and predicate calculus}
$= \exists\, v \bullet \mathbf{TR}(P)$

□

3.6 Results on CITR

In relation to the paper, the function **CITR** is the composition of **CI0132** after **TR**.

Lemma L.3.6.1

$$\mathbf{CITR} \circ \mathbf{R1}(P) = \mathbf{CITR}(P)$$

Proof.

$\mathbf{CITR} \circ \mathbf{R1}(P)$ {Lemma L.3.6.2}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{CI4_m}$ and **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R1} \circ \mathbf{CI4_m}(P)$
{tr_C and tr'_C are not free in **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R1} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{R1_C}$ and **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions **TR3** and **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{tr_T and tr'_T are not free in **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{R1_T}$ and **R1**}
$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Conjunctive healthiness conditions $\mathbf{R0_T}$ and **R1**}
$= \mathbf{CI0132} \circ \mathbf{R1} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$
{Definition of **CI0132** and Lemma L.4.1.3}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$

{Lemma L.3.6.2}

$= \mathbf{CITR}(P)$

□

Lemma L.3.6.2

$\mathbf{CITR}(P)$

$=$

$\mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$

Proof.

$\mathbf{CITR}(P)$ {Definition of **CITR**}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR} \circ \mathbf{R2_{loc}}(P)$ {Definition of **TR**}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{TR4} \circ \mathbf{R2_{loc}}(P)$

{Lemma L.6.0.1}

$= \mathbf{CI0132} \circ \mathbf{CIB} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T}(P)$ {Lemma L.4.7.6}

$= \mathbf{CI0132} \circ \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T}(P)$

{Lemma L.4.10.6}

$= \mathbf{CI0132} \circ \mathbf{CI4} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_C} \circ \mathbf{R2_T}(P)$

{Lemma L.4.10.8}

$= \mathbf{CI0132} \circ \mathbf{CI4} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T} \circ \mathbf{R2_C}(P)$

{Definition of $\mathbf{R0_T}$}

$= \mathbf{CI0132} \circ \mathbf{CI4} \circ \mathbf{R0_T} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_T} \circ \mathbf{R2_C}(P)$ {??}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{CI4_m} \circ \mathbf{TR3} \circ \mathbf{R2_T} \circ \mathbf{R2_C}(P)$

{Commutativity of conjunctive healthiness conditions (**TR3**, $\mathbf{CI4_m}$, $\mathbf{R1_C}$)}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{CI4_m} \circ \mathbf{R2_C}(P)$ {??}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$

□

Applying **CITR** to **R**.

Lemma L.3.6.3

$$\begin{array}{l}\mathbf{CITR} \circ \mathbf{R}(P) \\ = \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P))\end{array}$$

Proof.

$$\begin{array}{ll}\mathbf{CITR} \circ \mathbf{R}(P) & \{\text{Definition of } \mathbf{R}\} \\ = \mathbf{CITR} \circ \mathbf{R1} \circ \mathbf{R2} \circ \mathbf{R3}(P) & \{\text{Lemma L.3.6.1}\} \\ = \mathbf{CITR} \circ \mathbf{R2} \circ \mathbf{R3}(P) & \{\text{Commutativity of } \mathbf{R2} \text{ and } \mathbf{R3}\} \\ = \mathbf{CITR} \circ \mathbf{R3} \circ \mathbf{R2}(P) & \{\text{Lemma L.4.6.8}\} \\ = \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \wedge \mathbf{CI2_m} \circ \mathbf{R2}(P)) & \end{array}$$

□

Lemma L.3.6.4 *Provided ok' and $wait$ are not free in P and Q,*

$$\begin{array}{l}\mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array} \right) \\ = \\ \mathbf{S} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ \left(\begin{array}{l} ((R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S))) \\ \wedge \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \end{array} \right)\end{array}$$

Proof.

$$\mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array} \right)$$

{Lemma L.3.6.7 and definition of **S**}

$$= \mathbf{S} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ \left(\begin{array}{l} ((R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S))) \\ \wedge \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \end{array} \right)$$

□

Lemma L.3.6.5 *Provided ok' and $wait$ are not free in P and Q,*

$$\mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array} \right)$$

$$=$$

$$\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ (P \wedge Q)) \\ \vdash \\ \left(\begin{array}{l} (tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ (tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ \left(\begin{array}{l} \neg\ wait' \wedge \mathbf{R2}(R \vee S) \\ \wedge \\ (tr' = tr \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \end{array} \right) \end{array} \right)$$

Proof.

$$\mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array} \right) \qquad \{\text{Lemma L.3.6.6}\}$$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ (P \wedge Q)) \\ \vdash \\ \left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}((R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S)) \end{array} \right) \end{array} \right)$$

$\{\text{Distributivity of } \mathbf{R2} \text{ and } tr' = tr \text{ healthy}\}$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, (P \wedge Q)) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ (\mathbf{R2}(R \wedge S) \lhd tr' = tr \wedge wait' \rhd \mathbf{R2}(R \vee S)) \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, (P \wedge Q)) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \left(\begin{array}{l} (tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ (tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ (\neg\, wait' \wedge \mathbf{R2}(R \vee S)) \end{array} \right) \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, (P \wedge Q)) \\ \vdash \\ \left(\begin{array}{l} (tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ (tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ \left(\begin{array}{l} \neg\, wait' \wedge \mathbf{R2}(R \vee S) \\ \wedge \\ (tr' = tr \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \end{array} \right) \end{array} \right)$$

□

Lemma L.3.6.6 *Provided ok' and $wait$ are not free in P,*

$$\mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$$
$$=$$

$$\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

Proof.

$$\begin{aligned}
&\mathbf{CITR} \circ \mathbf{R}(P \vdash Q) && \{\text{Lemma L.3.6.3}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \mathbf{R2}(P \vdash Q) \\ \wedge \\ \mathbf{CI2_m} \circ \mathbf{R2}(P \vdash Q) \end{array} \right) \\
& && \{\text{Property of designs}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) \\ \wedge \\ \mathbf{CI2_m}(\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) \end{array} \right) \\
& && \{\text{Lemma L.4.5.5}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) \\ \wedge \\ \mathbf{CI2_m}(\neg\, \neg\, \mathbf{R2}(\neg\, P)) \end{array} \right) \\
& && \{\text{Predicate calculus}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) \\ \wedge \\ \mathbf{CI2_m} \circ \mathbf{R2}(\neg\, P) \end{array} \right) \\
& && \{\text{Assumption and Lemma L.4.5.5}\} \\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)
\end{aligned}$$

□

Well, it turns out that the **R2** can actually be taken out of the design, so we create

a new lemma for this.

Lemma L.3.6.7 *Provided ok' and wait are not free in P,*

$$\begin{array}{l}\mathbf{CITR} \circ \mathbf{R}(P \vdash Q) \\ = \\ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)\end{array}$$

Proof.

$$\mathbf{CITR} \circ \mathbf{R}(P \vdash Q) \qquad \{\text{Lemma L.3.6.6}\}$$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

$$\{\text{Predicate calculus}\}$$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ P) \\ \vdash \\ \left(\begin{array}{l} (wait' \vee tr' \neq tr \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

$$\{\mathbf{R2}(tr' \neq tr) = tr' \neq tr\}$$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\ \mathbf{R2}(\neg\ P) \\ \vdash \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2}(tr' \neq tr) \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

$$\{\text{Distributivity of } \mathbf{R2}\}$$

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \mathbf{R2} \left(\begin{array}{l} (wait' \vee tr' \neq tr \vee \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \mathbf{R2} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Definition of design and predicate calculus (Lemma L.4.6.11)}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

□

The following lemma is really the result I want to calculate.

Lemma L.3.6.8 *Provided tr_C and tr'_C are not free in P and Q, and ok' and $wait$ are not free in P,*

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$$

$$=$$

$$\mathbf{R_T} \left(\begin{array}{l} \neg\, (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

Proof.

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$$

$$= \exists\, \alpha \bullet \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Assumption and Lemma L.3.6.6}

$$= \mathbf{R_T} \left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \\ \mathbf{CI013} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right) \end{array} \right)$$

{Lemma L.4.6.17}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \\ \mathbf{CI013} \left(\begin{array}{l} \neg\, \mathbf{R2}(\neg\, P) \\ \vdash \\ \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\, \neg\, \mathbf{R2}(\neg\, P))) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Lemma L.4.6.16}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\, \neg\, \mathbf{R2}(\neg\, P))) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \left(\begin{array}{l} \neg\ (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ \neg\ \mathbf{R2}(\neg\ P))) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \left(\begin{array}{l} ((\neg\ wait' \land tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \land \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R_T} \left(\begin{array}{l} \neg\ (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\ P)) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \left(\begin{array}{l} ((\neg\ wait' \land tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \land \\ \mathbf{R2}(Q) \end{array} \right) \end{array} \right)$$

□

And it can be further simplified by the following.

Lemma L.3.6.9 *Provided tr_C and tr'_C are not free in P and Q, and ok' and $wait$ are not free in P,*

$$\begin{array}{l} \exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q) \\ = \\ \mathbf{R_T} \left(\begin{array}{l} (subsR2(P)[false/wait'] \lor wait'_T) \land subsR2(P)[true/wait'] \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \land \\ subsR2(Q)[false/wait'] \\ \land \\ \neg\ wait'_T \end{array} \right) \\ \lor \\ subsR2(Q)[true/wait'] \end{array} \right) \end{array} \right) \end{array}$$

Proof.

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q) \qquad \{\text{Lemma L.3.6.8}\}$$

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}\left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array}\right) \end{array}\right)$$

{Healthy predicate}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013}\left(\begin{array}{l} ((\neg\, wait' \wedge \mathbf{R2}(tr' = tr)) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013}\left(\begin{array}{l} (wait' \vee \neg\, \mathbf{R2}(tr' = tr) \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array}\right) \end{array}\right)$$

{Property $\neg\, \mathbf{R2}(P) = \mathbf{R2}(\neg\, P)$}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013}\left(\begin{array}{l} (wait' \vee \mathbf{R2}(tr' \neq tr) \vee \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}(Q) \end{array}\right) \end{array}\right)$$

{Distributivity of **R2**}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}\left(\begin{array}{l} (wait' \vee (tr' \neq tr) \vee \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R_T} \left(\begin{array}{l} \neg\, (\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, (\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \end{array} \right)$$

{Definition of design and conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, P)) \\ \vdash \\ \mathbf{R1_T} \left(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \end{array} \right) \right) \end{array} \right)$$

{Lemma L.4.6.20}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, \mathbf{R1_T}((subsR2(\neg\, P)[false/wait'] \wedge \neg\, wait'_T) \vee subsR2(\neg\, P)[true/wait']) \\ \vdash \\ \mathbf{R1_T} \left(\begin{array}{l} \left(\begin{array}{l} subsR2(((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[false/wait'] \\ \wedge \\ \neg\, wait'_T \end{array} \right) \\ \vee \\ subsR2(((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[true/wait'] \end{array} \right) \end{array} \right)$$

{Definition of design and conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\ ((subsR2(\neg\ P)[false/wait'] \wedge \neg\ wait'_T) \vee subsR2(\neg\ P)[true/wait']) \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[false/wait'] \\ \wedge \\ \neg\ wait'_T \end{array} \right) \\ \vee \\ subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[true/wait'] \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \left(\begin{array}{l} \neg\ ((subsR2(\neg\ P)[false/wait'] \wedge \neg\ wait'_T) \vee subsR2(\neg\ P)[true/wait']) \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[false/wait'] \\ \wedge \\ \neg\ wait'_T \end{array} \right) \\ \vee \\ subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[true/wait'] \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \mathbf{R_T} \left(\begin{array}{l} \left(\begin{array}{l} \neg\ (subsR2(\neg\ P)[false/wait'] \wedge \neg\ wait'_T) \\ \wedge \\ \neg\ subsR2(\neg\ P)[true/wait'] \end{array} \right) \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[false/wait'] \\ \wedge \\ \neg\ wait'_T \end{array} \right) \\ \vee \\ subsR2(((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q)[true/wait'] \end{array} \right) \end{array} \right)$$

{Definition of $subsR2$ and substitution}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\,(subsR2(\neg\,P)[false/wait'] \wedge \neg\,wait'_T) \\
\wedge \\
\neg\,subsR2(\neg\,P)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
((\neg\,false \wedge Flat(tr'_T) - Flat(tr_T) = \langle\rangle) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[false/wait']
\end{array}\right) \\
\wedge \\
\neg\,wait'_T
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
((\neg\,true \wedge Flat(tr'_T) - Flat(tr_T) = \langle\rangle) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[true/wait']
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Predicate calculus}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\,(subsR2(\neg\,P)[false/wait'] \wedge \neg\,wait'_T) \\
\wedge \\
\neg\,subsR2(\neg\,P)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
((Flat(tr'_T) - Flat(tr_T) = \langle\rangle) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[false/wait'] \\
\wedge \\
\neg\,wait'_T
\end{array}\right) \\
\vee \\
subsR2(Q)[true/wait']
\end{array}\right)
\end{array}\right)
$$

{Property of sequences}

$$= \mathbf{R_T}\left(\begin{array}{l}\left(\begin{array}{l}\neg\,(subsR2(\neg\,P)[false/wait'] \wedge \neg\,wait'_T)\\ \wedge\\ \neg\,subsR2(\neg\,P)[true/wait']\end{array}\right)\\ \vdash\\ \left(\begin{array}{l}\left(\begin{array}{l}(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T)\\ \wedge\\ subsR2(Q)[false/wait']\\ \wedge\\ \neg\,wait'_T\end{array}\right)\\ \vee\\ subsR2(Q)[true/wait']\end{array}\right)\end{array}\right)$$

{Property of substitution, definition of $subsR2$ and predicate calculus}

$$= \mathbf{R_T}\left(\begin{array}{l}(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait']\\ \vdash\\ \left(\begin{array}{l}\left(\begin{array}{l}(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T)\\ \wedge\\ subsR2(Q)[false/wait']\\ \wedge\\ \neg\,wait'_T\end{array}\right)\\ \vee\\ subsR2(Q)[true/wait']\end{array}\right)\end{array}\right)$$

□

Lemma L.3.6.10 *Provided tr_C and tr'_C are not free in P and Q, and ok' and $wait$ are not free in P and Q,*

$$\left(\begin{array}{l}\exists\, untimed\alpha, st\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash R)\\ \Box_T\\ \exists\, untimed\alpha, st\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(Q \vdash S)\end{array}\right)$$

$=$

$$
\mathbf{R_T}\left(\begin{array}{l}
(subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\
\vdash \\
\left(\begin{array}{l}
(subsR2(R)[true/wait'] \wedge subsR2(S)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\
\vee \\
subsR2(R \vee S)[true/wait']
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \\
\wedge \\
\left(\begin{array}{l}
subsR2(R)[false/wait'] \vee subsR2(S)[false/wait'] \\
\vee \\
subsR2(R)[true/wait'] \vee subsR2(S)[true/wait']
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

Proof.

$$
\left(\begin{array}{l}
\exists\, untimed\alpha, st\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash R) \\
\Box_T \\
\exists\, untimed\alpha, st\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(Q \vdash S)
\end{array}\right) \qquad \{\text{Lemma L.3.6.9}\}
$$

$$
= \left(\begin{array}{l}
\mathbf{R_T} \left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(R)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array} \right) \\
\vee \\
subsR2(R)[true/wait']
\end{array} \right)
\end{array} \right) \\
\Box_T \\
\mathbf{R_T} \left(\begin{array}{l}
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait'] \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(S)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array} \right) \\
\vee \\
subsR2(S)[true/wait']
\end{array} \right)
\end{array} \right)
\end{array} \right)
$$

{Definition of $\Box_T$}

$$
= \mathbf{R_T} \left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\wedge \\
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait']
\end{array} \right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(choice_{post}(R) \wedge choice_{post}(S)) \\
\lhd Flat(tr'_T) = Flat(tr_T) \rhd \\
(choice_{post}(R) \vee choice_{post}(S))
\end{array} \right) \\
\vee \\
\left(\begin{array}{l}
(choice_{post}(R) \vee choice_{post}(R)) \\
\wedge \\
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait'_T
\end{array} \right)
\end{array} \right)
\end{array} \right)
$$

{Definition of conditional}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\wedge \\
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
(choice_{post}(R) \wedge choice_{post}(S) \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
((choice_{post}(R) \vee choice_{post}(S)) \wedge Flat(tr'_T) \neq Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
(choice_{post}(R) \vee choice_{post}(S)) \\
\wedge \\
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait'_T
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.3.6.13}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\wedge \\
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
((choice_{post}(R) \vee choice_{post}(S)) \wedge Flat(tr'_T) \neq Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
(choice_{post}(R) \vee choice_{post}(S)) \\
\wedge \\
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait'_T
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.3.6.11}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\wedge \\
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
(subsR2(R)[true/wait'] \wedge subsR2(S)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\
\vee \\
subsR2(R \vee S)[true/wait']
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(choice_{post}(R) \vee choice_{post}(S)) \\
\wedge \\
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.3.6.14}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge subsR2(P)[true/wait'] \\
\wedge \\
(subsR2(Q)[false/wait'] \vee wait'_T) \wedge subsR2(Q)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
(subsR2(R)[true/wait'] \wedge subsR2(S)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\
\vee \\
subsR2(R \vee S)[true/wait']
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \\
\wedge \\
\left(\begin{array}{l}
subsR2(R)[false/wait'] \vee subsR2(S)[false/wait'] \\
\vee \\
subsR2(R)[true/wait'] \vee subsR2(S)[true/wait']
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of $subsR2$ and property of substitution}

$$
= \mathbf{R_T}\left(\begin{array}{l}
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \vee wait'_T) \wedge (subsR2(Q)[false/wait'] \vee wait'_T) \\
\wedge \\
subsR2(P \wedge Q)[true/wait']
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
(subsR2(R)[true/wait'] \wedge subsR2(S)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\
\vee \\
subsR2(R \vee S)[true/wait']
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \\
\wedge \\
\left(\begin{array}{l}
subsR2(R)[false/wait'] \vee subsR2(S)[false/wait'] \\
\vee \\
subsR2(R)[true/wait'] \vee subsR2(S)[true/wait']
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of $subsR2$, property of substitution and predicate calculus}

$$
= \mathbf{R_T}\left(\begin{array}{l}
(subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\
\vdash \\
\left(\begin{array}{l}
(subsR2(R)[true/wait'] \wedge subsR2(S)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\
\vee \\
subsR2(R \vee S)[true/wait']
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \\
\wedge \\
\left(\begin{array}{l}
subsR2(R)[false/wait'] \vee subsR2(S)[false/wait'] \\
\vee \\
subsR2(R)[true/wait'] \vee subsR2(S)[true/wait']
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

□

Lemma L.3.6.11

$$
\begin{array}{l}
(choice_{post}(P) \vee choice_{post}(Q)) \wedge Flat(tr'_T) \neq Flat(tr_T) \\
= \\
\left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(P \vee Q)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
subsR2(P \vee Q)[true/wait']
\end{array} \right)
\end{array} \right)
\end{array}
$$

Proof.

$$
\begin{array}{ll}
(choice_{post}(P) \vee choice_{post}(Q)) \wedge Flat(tr'_T) \neq Flat(tr_T) & \{\text{Lemma L.3.6.12}\} \\
= \left(\begin{array}{l}
(Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
(Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T)) \\
\vee \\
(subsR2(Q)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T))
\end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l}
Flat(tr'_T) \neq Flat(tr_T) \\
\wedge \\
\left(\begin{array}{l}
(subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(Q)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'])
\end{array} \right)
\end{array} \right) & \{\text{Predicate calculus}\}
\end{array}
$$

$$= \left(\begin{array}{l} Flat(tr'_T) \neq Flat(tr_T) \\ \wedge \\ \left(\begin{array}{l} ((subsR2(P)[false/wait'] \vee subsR2(Q)[false/wait']) \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait']) \end{array} \right) \end{array} \right)$$

{Definition of $subsR2$ and property of substitution}

$$= \left(\begin{array}{l} Flat(tr'_T) \neq Flat(tr_T) \\ \wedge \\ \left(\begin{array}{l} (subsR2(P \vee Q)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ subsR2(P \vee Q)[true/wait'] \end{array} \right) \end{array} \right)$$

□

Lemma L.3.6.12

$$\begin{array}{l} (choice_{post}(P) \vee choice_{post}(Q)) \wedge Flat(tr'_T) \neq Flat(tr_T) \\ = \\ \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T)) \\ \vee \\ (subsR2(Q)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T)) \end{array} \right) \end{array}$$

Proof.

$(choice_{post}(P) \vee choice_{post}(Q)) \wedge Flat(tr'_T) \neq Flat(tr_T)$ {Lemma L.3.6.17}

$$
= \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(Q)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array} \right) \\ \wedge \\ Flat(tr'_T) \neq Flat(tr_T) \end{array} \right) \quad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T)) \\ \vee \\ (subsR2(Q)[true/wait'] \wedge Flat(tr'_T) \neq Flat(tr_T)) \end{array} \right)
$$

□

Lemma L.3.6.13

$$
\left(\begin{array}{l} (choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T)) \end{array} \right)
$$

$$
=
$$

$$
\left(\begin{array}{l} (choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \end{array} \right)
$$

Proof.

$$
\left(\begin{array}{l} (choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T)) \end{array} \right) \quad \{\text{Lemma L.3.6.16}\}
$$

$$
= \left(\begin{array}{l}
(subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T))
\end{array} \right) \qquad \{\text{Lemma L.3.6.18}\}
$$

$$
= \left(\begin{array}{l}
(subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\
\vee \\
(choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
\left(\begin{array}{l}
\#tr'_T = \#tr_T \wedge \neg\, wait'_T \wedge Flat(tr'_T) = Flat(tr_T) \\
\wedge \\
subsR2(P)[false/wait'] \wedge subsR2(Q)[false/wait']
\end{array} \right) \\
\vee \\
\left(\begin{array}{l}
\#tr'_T = \#tr_T \wedge subsR2(P)[false/wait'] \wedge \neg\, wait'_T \\
\wedge \\
subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)
\end{array} \right) \\
\vee \\
\left(\begin{array}{l}
subsR2(P)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T) \\
\wedge \\
\#tr'_T = \#tr_T \wedge subsR2(Q)[false/wait'] \wedge \neg\, wait'_T
\end{array} \right) \\
\vee \\
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T))
\end{array} \right)
$$

{Predicate calculus: absorption law}

$$
= \left(\begin{array}{l}
(subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\
\vee \\
(choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T)) \\
\vee \\
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T))
\end{array} \right)
$$

{Lemma L.3.6.16}

$$
= \left(\begin{array}{l}
(choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\
\vee \\
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T))
\end{array} \right)
$$

□

Lemma L.3.6.14

$$
\begin{array}{l}
(choice_{post}(P) \vee choice_{post}(Q)) \wedge (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait'_T \\
= \\
\left(\begin{array}{l}
\left(\begin{array}{l}
subsR2(P)[false/wait'] \vee subsR2(Q)[false/wait'] \\
\vee \\
subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait']
\end{array} \right) \\
\wedge \\
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait'_T
\end{array} \right)
\end{array}
$$

Proof.

$$(choice_{post}(P) \vee choice_{post}(Q)) \wedge (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T$$

$$= \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(Q)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array} \right) \\ \wedge \\ (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \end{array} \right) \quad \{\text{Lemma L.3.6.17}\}$$

$$= \left(\begin{array}{l} \left(\begin{array}{l} subsR2(P)[false/wait'] \vee subsR2(Q)[false/wait'] \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array} \right) \\ \wedge \\ (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\, wait'_T \end{array} \right) \quad \{\text{Predicate calculus}\}$$

□

Lemma L.3.6.15

$$(choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T$$
$$=$$

$$\left(\begin{array}{l} \#tr_T = \#tr'_T \wedge \neg\ wait'_T \\ \wedge \\ \left(\begin{array}{l} subsR2(P)[false/wait'] \vee subsR2(Q)[false/wait'] \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array}\right) \end{array}\right)$$

Proof.

$$\begin{array}{ll} (choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T & \{\text{Lemma L.3.6.16}\} \\ = \left(\begin{array}{l} (subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \end{array}\right) & \{\text{Predicate calculus}\} \\ = \left(\begin{array}{l} \#tr_T = \#tr'_T \wedge \neg\ wait'_T \\ \wedge \\ \left(\begin{array}{l} subsR2(P)[false/wait'] \vee subsR2(Q)[false/wait'] \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array}\right) \end{array}\right) & \end{array}$$

□

Lemma L.3.6.16

$$\begin{array}{l} (choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T \\ = \\ \left(\begin{array}{l} (subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\ wait'_T) \end{array}\right) \end{array}$$

Proof.

$$
\begin{array}{ll}
(choice_{post}(P) \vee choice_{post}(Q)) \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T & \{\text{Lemma L.3.6.17}\} \\
= \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(Q)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array} \right) \\ \wedge \\ \#tr_T = \#tr'_T \wedge \neg\, wait'_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} (subsR2(P)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\ \vee \\ (subsR2(Q)[false/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\ \vee \\ (subsR2(P)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \\ \vee \\ (subsR2(Q)[true/wait'] \wedge \#tr_T = \#tr'_T \wedge \neg\, wait'_T) \end{array} \right) &
\end{array}
$$

□

Lemma L.3.6.17

$$
\begin{array}{l}
choice_{post}(P) \vee choice_{post}(Q) \\
=
\end{array}
$$

$$
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(P)[false/wait'] \wedge \neg\ wait'_T
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[false/wait'] \wedge \neg\ wait'_T
\end{array}\right) \\
\vee \\
subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait']
\end{array}\right)
$$

Proof.

$$
\begin{array}{ll}
choice_{post}(P) \vee choice_{post}(Q) & \{\text{Definition of } choice_{post}\} \\
= \left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(P)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array}\right) \\
\vee \\
subsR2(P)[true/wait']
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array}\right) \\
\vee \\
subsR2(Q)[true/wait']
\end{array}\right)
\end{array}\right) & \{\text{Predicate calculus}\}
\end{array}
$$

$$= \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(Q)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\ \vee \\ subsR2(P)[true/wait'] \vee subsR2(Q)[true/wait'] \end{array} \right)$$

□

Lemma L.3.6.18

$$\begin{array}{l} choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T) \\ = \\ \left(\begin{array}{l} \left(\begin{array}{l} \#tr'_T = \#tr_T \wedge \neg\ wait'_T \wedge Flat(tr'_T) = Flat(tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge subsR2(Q)[false/wait'] \end{array} \right) \\ \vee \\ \left(\begin{array}{l} \#tr'_T = \#tr_T \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T \\ \wedge \\ subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T) \end{array} \right) \\ \vee \\ \left(\begin{array}{l} subsR2(P)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T) \\ \wedge \\ \#tr'_T = \#tr_T \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\ \vee \\ (subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T)) \end{array} \right) \end{array}$$

Proof.

$choice_{post}(P) \wedge choice_{post}(Q) \wedge Flat(tr'_T) = Flat(tr_T)$ {Definition of $choice_{post}$}

$$
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(P)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array}\right) \\
\vee \\
subsR2(P)[true/wait']
\end{array}\right) \\
\wedge \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(Q)[false/wait'] \\
\wedge \\
\neg\ wait'_T
\end{array}\right) \\
\vee \\
subsR2(Q)[true/wait']
\end{array}\right) \\
\wedge \\
Flat(tr'_T) = Flat(tr_T)
\end{array}\right)
$$

$$
= \left(\begin{array}{l}
\left(\begin{array}{l}
(\#tr'_T = \#tr_T \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
subsR2(P)[true/wait']
\end{array}\right) \\
\wedge \\
\left(\begin{array}{l}
(\#tr'_T = \#tr_T \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T) \\
\vee \\
subsR2(Q)[true/wait']
\end{array}\right) \\
\wedge \\
Flat(tr'_T) = Flat(tr_T)
\end{array}\right)
$$

{Predicate calculus}

{Predicate calculus}

$$
= \left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} \#tr'_T = \#tr_T \wedge \neg\ wait'_T \\ \wedge \\ subsR2(P)[false/wait'] \wedge subsR2(Q)[false/wait'] \end{array} \right) \\
\vee \\
\left(\begin{array}{l} \#tr'_T = \#tr_T \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T \\ \wedge \\ subsR2(Q)[true/wait'] \end{array} \right) \\
\vee \\
\left(\begin{array}{l} subsR2(P)[true/wait'] \\ \wedge \\ \#tr'_T = \#tr_T \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\
\vee \\
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'])
\end{array} \right) \\
\wedge \\
Flat(tr'_T) = Flat(tr_T)
\end{array} \right)
$$

{Predicate calculus}

$$
= \left(\begin{array}{l}
\left(\begin{array}{l} \#tr'_T = \#tr_T \wedge \neg\ wait'_T \wedge Flat(tr'_T) = Flat(tr_T) \\ \wedge \\ subsR2(P)[false/wait'] \wedge subsR2(Q)[false/wait'] \end{array} \right) \\
\vee \\
\left(\begin{array}{l} \#tr'_T = \#tr_T \wedge subsR2(P)[false/wait'] \wedge \neg\ wait'_T \\ \wedge \\ subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T) \end{array} \right) \\
\vee \\
\left(\begin{array}{l} subsR2(P)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T) \\ \wedge \\ \#tr'_T = \#tr_T \wedge subsR2(Q)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \\
\vee \\
(subsR2(P)[true/wait'] \wedge subsR2(Q)[true/wait'] \wedge Flat(tr'_T) = Flat(tr_T))
\end{array} \right)
$$

□

Followed by interesting results on standard CSP operators, namely external choice.

Lemma L.3.6.19

$$\left(\begin{array}{l} \exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \\ \quad \mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array}\right) \end{array}\right)$$

$$=$$

$$\mathbf{R_T} \left(\begin{array}{l} (subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\ \vdash \\ \left(\begin{array}{l} subsR2(R \wedge S)[true/wait'] \\ \lhd Flat(tr'_T) = Flat(tr_T) \rhd \\ \left(\begin{array}{l} (subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (subsR2(R \vee S)[true/wait']) \end{array}\right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \end{array}\right) \end{array}\right) \end{array}\right)$$

Proof.

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CITR} \circ \mathbf{R} \left(\begin{array}{l} (P \wedge Q) \\ \vdash \\ (R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S) \end{array}\right)$$

{Lemma L.3.6.8}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \mathbf{R2}((R \wedge S) \lhd tr' = tr \wedge wait' \rhd (R \vee S)) \end{array}\right) \end{array}\right)$$

{Distributivity of **R2**}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ (\mathbf{R2}(R \wedge S) \lhd tr' = tr \wedge wait' \rhd \mathbf{R2}(R \vee S)) \end{array}\right) \end{array}\right)$$

{Definition of conditional}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \left(\begin{array}{l} (tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ (tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ (\neg\ wait' \wedge \mathbf{R2}(R \vee S)) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} (tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ (tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ ((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\ wait' \wedge \mathbf{R2}(R \vee S)) \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \exists\,\alpha \bullet \mathbf{CI013}(tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ \exists\,\alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} (tr' = tr \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \neg\ wait' \wedge \mathbf{R2}(R \vee S) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T}\left(\begin{array}{l} \neg\,(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \exists\,\alpha \bullet \mathbf{CI013}(tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)) \\ \vee \\ \exists\,\alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S)) \\ \vee \\ \exists\,\alpha \bullet \mathbf{CI013}\left(\begin{array}{l} (tr' = tr \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \neg\ wait' \wedge \mathbf{R2}(R \vee S) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R_T} \circ \mathbf{R1_T}\left(\begin{array}{l} \neg\ \mathbf{R1_T}(\exists\,\alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\,(P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \mathbf{R1_T}(\exists\,\alpha \bullet \mathbf{CI013}(tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S))) \\ \vee \\ \mathbf{R1_T}(\exists\,\alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S))) \\ \vee \\ \mathbf{R1_T}\left(\begin{array}{l} \exists\,\alpha \bullet \mathbf{CI013}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \neg\ wait' \wedge \mathbf{R2}(R \vee S)) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Lemma L.4.6.15}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, (P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\ \vee \\ \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S))) \\ \vee \\ \mathbf{R1_T} \left(\begin{array}{l} \exists\, \alpha \bullet \mathbf{CI013}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \neg\, wait' \wedge \mathbf{R2}(R \vee S)) \end{array} \right) \end{array} \right) \end{array} \right)$$

{Lemma L.4.6.14}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, (P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\ \vee \\ \mathbf{R1_T} \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\ \vee \\ \mathbf{R1_T} \left(\begin{array}{l} \exists\, \alpha \bullet \mathbf{CI013}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ \neg\, wait' \wedge \mathbf{R2}(R \vee S)) \end{array} \right) \end{array} \right) \end{array} \right)$$

{Lemma L.4.6.12}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \neg\, \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\neg\, (P \wedge Q))) \\ \vdash \\ \left(\begin{array}{l} \mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\ \vee \\ \mathbf{R1_T} \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\ \vee \\ \mathbf{R1_T}((Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \end{array} \right) \end{array} \right)$$

{Lemma L.4.6.20}

$$
= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l}
\neg\, \mathbf{R1_T} \left(\begin{array}{l} (subsR2(\neg\,(P \wedge Q))[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ subsR2(\neg\,(P \wedge Q))[true/wait'] \end{array} \right) \\
\vdash \\
\left(\begin{array}{l}
\mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\
\vee \\
\mathbf{R1_T} \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\
\vee \\
\mathbf{R1_T} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T \end{array} \right)
\end{array} \right)
\end{array} \right)
$$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$$
= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l}
\neg \left(\begin{array}{l} (subsR2(\neg\,(P \wedge Q))[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ subsR2(\neg\,(P \wedge Q))[true/wait'] \end{array} \right) \\
\vdash \\
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\
\vee \\
\left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\
\vee \\
\left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T \end{array} \right)
\end{array} \right)
\end{array} \right)
$$

{Predicate calculus, definition of $subsR2$ and property of substitution}

$$= \mathbf{R_T} \circ \mathbf{R1_T} \left(\begin{array}{l} (subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\ \vdash \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R_T}$ Lemma L.2.2.12 and commutativity with $\mathbf{R3_T}$}

$$= \mathbf{R_T} \left(\begin{array}{l} (subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\ \vdash \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait']) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait']) \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T \end{array} \right) \end{array} \right) \end{array} \right)$$

{Predicate calculus and definition of conditional}

$$= \mathbf{R_T} \left(\begin{array}{l} (subsR2(P \wedge Q)[false/wait'] \vee wait'_T) \wedge subsR2(P \wedge Q)[true/wait'] \\ \vdash \\ \left(\begin{array}{l} subsR2(R \wedge S)[true/wait'] \\ \lhd Flat(tr'_T) = Flat(tr_T) \rhd \\ \left(\begin{array}{l} (subsR2(R \vee S)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (subsR2(R \vee S)[true/wait']) \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(R \vee S)[false/wait'] \wedge \neg\ wait'_T \end{array} \right) \end{array} \right) \end{array} \right)$$

□

Lemma L.3.6.20 *Let*

$$\mathbf{ST_m}(P) = \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$$
$$CI2.0_m = (\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T$$

Then,

$$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^o_f = (\mathbf{ST_m}((ok \wedge P^o) \Rightarrow CI2.0_m \wedge Q^o \wedge o))_f$$

Proof.

$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^o_f$ {Lemma L.3.6.23}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \right)_f$$

{Lemmas L.4.1.1 and L.4.2.2}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \left(\mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \right)_f$$

{Lemma L.4.8.3}

$$= \mathbf{R012_T} \left(\mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \right)_f$$

{Lemmas L.2.1.2, L.2.2.25 and L.2.3.2}

$$= \left(\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \right)_f$$

□

Using this result can then derive the results for ^{f} and ^{t} as follows.

Lemma L.3.6.21

$$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^f_f = (\mathbf{ST_m} \circ \mathbf{H1}(\neg\ P^f))_f$$

Proof.

$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^f_f$ {Lemma L.3.6.20}

$= (\mathbf{ST_m}((ok \wedge P^f) \Rightarrow (CI2.0_m \wedge Q^f \wedge false)))_f$ {Predicate calculus}

$= (\mathbf{ST_m}(\neg\ (ok \wedge P^f)))_f$ {Predicate calculus}

$= (\mathbf{ST_m}(\neg\ ok \vee \neg\ P^f))_f$ {Definition of **H1**}

$= (\mathbf{ST_m} \circ \mathbf{H1}(\neg\ P^f))_f$

□

Lemma L.3.6.22

$$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^t_f = (\mathbf{ST_m}((ok \wedge P^t) \Rightarrow CI2.0_m \wedge Q^t))_f$$

Proof.

$$\begin{array}{ll}
(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^t_f & \{\text{Lemma L.3.6.20}\} \\
= (\mathbf{ST_m}((ok \wedge P^t) \Rightarrow CI2.0_m \wedge Q^t \wedge true))_f & \{\text{Predicate calculus}\} \\
= (\mathbf{ST_m}((ok \wedge P^t) \Rightarrow CI2.0_m \wedge Q^t))_f &
\end{array}$$

□

Lemma L.3.6.23 *Provided ok' and wait are not free in P,*

$$(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^o_f$$
$$=$$
$$\mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\begin{array}{l} \mathbf{CI3} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \end{array} \right)_f$$

Proof.

$$\begin{array}{ll}
(\mathbf{CITR} \circ \mathbf{R}(P \vdash Q))^o_f & \{\text{Lemma L.3.6.7}\} \\
= \left(\begin{array}{l} \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right) \end{array} \right)^o_f & \\
\{\text{Lemmas L.2.1.2, L.2.2.25 and L.2.3.2}\} &
\end{array}$$

$$= \left(\mathbf{R012_T} \left(\mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o_f \right) \right)$$

{Lemma L.4.8.3}

$$= \left(\mathbf{R012_T} \circ \mathbf{CIB} \left(\mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o_f \right) \right)$$

{Lemmas L.4.1.1 and L.4.2.2}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o_f \right)$$

{Lemma L.4.3.3}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o_f \right)$$

{Lemma L.4.3.4}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \left(\mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o \right)_f \right)$$

{Definition of **R2** and substitution}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \left(\mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ \left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q \end{array} \right)^o \right) \right) \right)_f$$

{Definition of design and substitution}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\mathbf{CI3} \circ \mathbf{R2} \left(\begin{array}{l} (ok \wedge P^o) \\ \Rightarrow \\ \left(\left(\begin{array}{l} (\neg\ wait' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \wedge Q^o \wedge o \right) \end{array} \right) \right)_f$$

□

Lemma L.3.6.24 *Let*

$$P = \mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)$$
$$Q = \mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)$$
$$\mathbf{S}(P) = \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$$

Then,

$$P\ \square_{STRP}\ Q$$
$$=$$
$$\mathbf{S} \left(\begin{array}{l} \neg\ P_f^f \wedge \neg\ Q_f^f \\ \vdash \\ (wait' \wedge P_f^t \wedge Q_f^t) \lhd tr' = tr \rhd (P_f^t \vee Q_f^t) \\ \vee \\ (\neg\ wait' \wedge (P_f^t \vee Q_f^t) \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T)) \end{array} \right)$$

Proof.

$P\ \square_{ST}\ Q$ {Assumption}

$= \mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)\ \square_{ST}\ \mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)$ {Definition of $\square_{STRP}$}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\ \mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^f \\
\wedge \\
\neg\ \mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^f
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\wedge \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\vee \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\vee \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
\qquad \{\text{Lemma L.3.6.21}\}
$$

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\ (\mathbf{ST_m} \circ \mathbf{H1}((P_f^f)^f))_f \\
\wedge \\
\neg\ (\mathbf{ST_m} \circ \mathbf{H1}((Q_f^f)^f))_f
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\wedge \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\vee \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{CITR} \circ \mathbf{R}(\neg\ P_f^f \vdash P_f^t)_f^t \\
\vee \\
\mathbf{CITR} \circ \mathbf{R}(\neg\ Q_f^f \vdash Q_f^t)_f^t
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
\qquad \{\text{Lemma L.3.6.22}\}
$$

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\,(\mathbf{ST_m} \circ \mathbf{H1}((P_f^f)^f))_f \\
\wedge \\
\neg\,(\mathbf{ST_m} \circ \mathbf{H1}((Q_f^f)^f))_f
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge (\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t))_f \\
\wedge \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))_f
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t))_f \\
\vee \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))_f
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\, wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
(\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t))_f \\
\vee \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))_f
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of $\mathbf{S}$, $\mathbf{R3_T}$, Lemma L.2.4.1 and substitution}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\,(\mathbf{ST_m} \circ \mathbf{H1}((P_f^f)^f)) \\
\wedge \\
\neg\,(\mathbf{ST_m} \circ \mathbf{H1}((Q_f^f)^f))
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge (\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t)) \\
\wedge \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t)) \\
\vee \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\, wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
(\mathbf{ST_m}((ok \wedge \neg\,(P_f^f)^t) \Rightarrow CI2.0_m \wedge (P_f^t)^t)) \\
\vee \\
(\mathbf{ST_m}((ok \wedge \neg\,(Q_f^f)^t) \Rightarrow CI2.0_m \wedge (Q_f^t)^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Substitution}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\, \mathbf{ST_m} \circ \mathbf{H1}(P_f^f) \\
\wedge \\
\neg\, \mathbf{ST_m} \circ \mathbf{H1}(Q_f^f)
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{ST_m}((ok \wedge \neg\, P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t)) \\
\wedge \\
\mathbf{ST_m}((ok \wedge \neg\, Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{ST_m}((ok \wedge \neg\, P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}((ok \wedge \neg\, Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\, wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{ST_m}((ok \wedge \neg\, P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}((ok \wedge \neg\, Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of design and Lemma L.5.0.7}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}\neg\ \mathbf{ST_m}(P_f^f)\\ \wedge\\ \neg\ \mathbf{ST_m}(Q_f^f)\end{array}\right)\\
\vdash\\
\left(\begin{array}{l}\left(\begin{array}{l}\left(\begin{array}{l}wait' \wedge \mathbf{ST_m}((ok \wedge \neg\ P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t))\\ \wedge\\ \mathbf{ST_m}((ok \wedge \neg\ Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))\end{array}\right)\\ \lhd tr' = tr \rhd\\ \left(\begin{array}{l}\mathbf{ST_m}((ok \wedge \neg\ P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t))\\ \vee\\ \mathbf{ST_m}((ok \wedge \neg\ Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)\\
\vee\\
\left(\begin{array}{l}\neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T)\\ \wedge\\ \left(\begin{array}{l}\mathbf{ST_m}((ok \wedge \neg\ P_f^f) \Rightarrow CI2.0_m \wedge (P_f^t))\\ \vee\\ \mathbf{ST_m}((ok \wedge \neg\ Q_f^f) \Rightarrow CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)
\end{array}\right)
$$

{Predicate calculus}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\ \mathbf{ST_m}(P_f^f) \\
\wedge \\
\neg\ \mathbf{ST_m}(Q_f^f)
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{ST_m}(\neg\ ok \vee P_f^f \vee (CI2.0_m \wedge (P_f^t))) \\
\wedge \\
\mathbf{ST_m}(\neg\ ok \vee Q_f^f \vee (CI2.0_m \wedge (Q_f^t)))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{ST_m}(\neg\ ok \vee P_f^f \vee (CI2.0_m \wedge (P_f^t))) \\
\vee \\
\mathbf{ST_m}(\neg\ ok \vee Q_f^f \vee (CI2.0_m \wedge (Q_f^t)))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{ST_m}(\neg\ ok \vee P_f^f \vee (CI2.0_m \wedge (P_f^t))) \\
\vee \\
\mathbf{ST_m}(\neg\ ok \vee Q_f^f \vee (CI2.0_m \wedge (Q_f^t)))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.5.0.12}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
\neg\ \mathbf{ST_m}(P_f^f) \\
\wedge \\
\neg\ \mathbf{ST_m}(Q_f^f)
\end{array}\right) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge (\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))) \\
\wedge \\
(\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))) \\
\vee \\
(\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
(\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))) \\
\vee \\
(\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of design and Lemma L.5.0.13}

$$
= \mathbf{S}\left(\begin{array}{l}
\left(\begin{array}{l}\neg\, \mathbf{ST_m}(P_f^f)\\ \wedge \\ \neg\, \mathbf{ST_m}(Q_f^f)\end{array}\right)\\
\vdash\\
\left(\begin{array}{l}\left(\begin{array}{l}\left(\begin{array}{l}wait' \wedge (\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)))\\ \wedge \\ (\mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)))\end{array}\right)\\ \lhd tr' = tr \rhd \\ \left(\begin{array}{l}\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))\\ \vee \\ \mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)\\
\vee \\
\left(\begin{array}{l}\neg\, wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T)\\ \wedge \\ \left(\begin{array}{l}\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))\\ \vee \\ \mathbf{ST_m}(Q_f^f) \vee \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)
\end{array}\right)
$$

{Definition of design and predicate calculus}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f) \wedge \neg\, \mathbf{ST_m}(Q_f^f)\\
\vdash\\
\left(\begin{array}{l}\left(\begin{array}{l}\left(\begin{array}{l}wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t))\\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))\end{array}\right)\\ \lhd tr' = tr \rhd \\ \left(\begin{array}{l}\mathbf{ST_m}(CI2.0_m \wedge (P_f^t))\\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)\\
\vee \\
\left(\begin{array}{l}\neg\, wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T)\\ \wedge \\ \left(\begin{array}{l}\mathbf{ST_m}(CI2.0_m \wedge (P_f^t))\\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)
\end{array}\right)
$$

{Predicate calculus}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\wedge \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\neg\ wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l}
wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\wedge \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
\mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
((\neg\ wait' \wedge tr' \neq tr) \vee (\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}
\mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\
\vee \\
\mathbf{ST_m}(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right) \qquad \{\text{Lemma L.2.3.12}\}
$$

$$= \mathbf{S}\left(\begin{array}{l} \neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\ \lhd tr' = tr \rhd \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right) \\ \vee \\ \left(\begin{array}{l} (\mathbf{R2_T}(\neg\ wait' \wedge tr' \neq tr) \vee (\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\ \wedge \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right) \end{array}\right) \end{array}\right)$$

$\{\mathbf{R2}(tr' \neq tr) = tr' \neq tr$ and distributivity of $\mathbf{R2}\}$

$$= \mathbf{S}\left(\begin{array}{l} \neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\ \lhd tr' = tr \rhd \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right) \\ \vee \\ \left(\begin{array}{l} (\mathbf{R2_T} \circ \mathbf{R2}(\neg\ wait' \wedge tr' \neq tr) \vee (\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\ \wedge \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Lemmas L.2.3.3 and L.2.3.18}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(\mathbf{R2_T} \circ \mathbf{R2}(\neg\ wait' \wedge tr' \neq tr) \vee \mathbf{R2_T}(\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

$\{tr$ and tr' not free and definition of $\mathbf{R2}\}$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(\mathbf{R2_T} \circ \mathbf{R2}(\neg\ wait' \wedge tr' \neq tr) \vee \mathbf{R2_T} \circ \mathbf{R2}(\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Distributivity of $\mathbf{R2_T}$ (Lemma L.2.3.18) and $\mathbf{R2}$}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{R2}((\neg\ wait' \wedge tr' \neq tr) \vee (\neg\ wait' \wedge \#tr_T = \#tr'_T)) \\ \wedge \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Predicate calculus}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(\mathbf{ST_m}(P_f^f) \vee \mathbf{ST_m}(Q_f^f)) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{R2}(\neg\ wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\ \wedge \\ \left(\begin{array}{l} \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.5.0.12}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\mathbf{ST_m}\left(\begin{array}{l} (CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
\mathbf{R2_T} \circ \mathbf{R2}(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\mathbf{ST_m}\left(\begin{array}{l} (CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.5.0.9}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\left(\begin{array}{l} wait' \wedge \mathbf{ST_m}(CI2.0_m \wedge (P_f^t)) \\ \wedge \\ \mathbf{ST_m}(CI2.0_m \wedge (Q_f^t)) \end{array}\right) \\
\lhd tr' = tr \rhd \\
\mathbf{ST_m}\left(\begin{array}{l} (CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right) \\
\vee \\
\mathbf{ST_m}\left(\begin{array}{l}
(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l} (CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t)) \end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemmas L.5.0.8 and L.5.0.11}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\mathbf{ST_m}(wait' \wedge CI2.0_m \wedge P_f^t \wedge Q_f^t) \\
\lhd tr' = tr \rhd \\
\mathbf{ST_m}\left(\begin{array}{l}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t))\end{array}\right)
\end{array}\right) \\
\vee \\
\mathbf{ST_m}\left(\begin{array}{l}
(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t))\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

$\{\mathbf{R2}(tr' = tr) = tr' = tr$ and also $\mathbf{R2_T}\}$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\mathbf{ST_m}(wait' \wedge CI2.0_m \wedge P_f^t \wedge Q_f^t) \\
\lhd \mathbf{R2_T} \circ \mathbf{R2}(tr' = tr) \rhd \\
\mathbf{ST_m}\left(\begin{array}{l}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t))\end{array}\right)
\end{array}\right) \\
\vee \\
\mathbf{ST_m}\left(\begin{array}{l}
(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t))\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

$\{$Lemma L.5.0.6$\}$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{ll}
\mathbf{ST_m} & \left(\begin{array}{l}
(wait' \wedge CI2.0_m \wedge P_f^t \wedge Q_f^t) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee & \\
\mathbf{ST_m} & \left(\begin{array}{l}
(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.5.0.12}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\, \mathbf{ST_m}(P_f^f \vee Q_f^f) \\
\vdash \\
\mathbf{ST_m}\left(\begin{array}{ll}
\left(\begin{array}{l}
(wait' \wedge CI2.0_m \wedge P_f^t \wedge Q_f^t) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) & \\
\vee & \\
\left(\begin{array}{l}
(\neg\, wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) &
\end{array}\right)
\end{array}\right)
$$

{Lemma L.4.6.25}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(wait' \wedge CI2.0_m \wedge P_f^t \wedge Q_f^t) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(\neg\ wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right) \qquad \{\text{Lemma L.4.6.24}\}
$$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(wait' \wedge P_f^t \wedge Q_f^t) \\
\lhd tr' = tr \rhd \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right) \\
\vee \\
\left(\begin{array}{l}
(\neg\ wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\
\wedge \\
\left(\begin{array}{l}
(CI2.0_m \wedge (P_f^t)) \\
\vee \\
(CI2.0_m \wedge (Q_f^t))
\end{array}\right)
\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Definition of conditional and Lemma L.4.6.23}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}(wait' \wedge P_f^t \wedge Q_f^t) \\ \lhd tr' = tr \rhd \\ (P_f^t \vee Q_f^t)\end{array}\right) \\
\vee \\
\left(\begin{array}{l}(\neg\ wait' \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\ \wedge \\ \left(\begin{array}{l}(CI2.0_m \wedge (P_f^t)) \\ \vee \\ (CI2.0_m \wedge (Q_f^t))\end{array}\right)\end{array}\right)
\end{array}\right)
\end{array}\right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\,(P_f^f \vee Q_f^f) \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}(wait' \wedge P_f^t \wedge Q_f^t) \\ \lhd tr' = tr \rhd \\ (P_f^t \vee Q_f^t)\end{array}\right) \\
\vee \\
\left(\begin{array}{l}(\neg\ wait' \wedge CI2.0_m \wedge (tr' \neq tr \vee \#tr_T = \#tr'_T)) \\ \wedge \\ (P_f^t \vee Q_f^t)\end{array}\right)
\end{array}\right)
\end{array}\right)
$$

{Lemma L.4.6.22 and predicate calculus}

$$
= \mathbf{S}\left(\begin{array}{l}
\neg\ P_f^f \wedge \neg\ Q_f^f \\
\vdash \\
\left(\begin{array}{l}((wait' \wedge P_f^t \wedge Q_f^t) \lhd tr' = tr \rhd (P_f^t \vee Q_f^t)) \\ \vee \\ \neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \wedge (P_f^t \vee Q_f^t)\end{array}\right)
\end{array}\right)
$$

□

Lemma L.3.6.25

$$\mathbf{S} \circ \mathbf{R012_T}(P) = \mathbf{S}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{S} \circ \mathbf{R012_T}(P) && \{\text{Definition of } \mathbf{S}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{R012_T}(P) && \{\text{Lemma L.4.6.26}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P) && \{\mathbf{R012_T}\text{-idempotent}\}\\
&= \mathbf{R012_T} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P) && \{\text{Lemma L.4.7.20}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P) && \{\text{Lemma L.4.6.27}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R012_T} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P)\\
& && \{\text{Definition of } \mathbf{R3_T}\text{, distributivity of } \mathbf{R012_T} \text{ and idempotent}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R012_T} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) && \{\text{Lemma L.4.6.27}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) && \{\text{Lemma L.4.7.20}\}\\
&= \mathbf{R012_T} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) && \{\mathbf{R012_T}\text{-idempotent}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) && \{\text{Definition of } \mathbf{S}\}\\
&= \mathbf{S}(P)
\end{aligned}
$$

□

Lemma L.3.6.26

$$\mathbf{S} \circ \mathbf{CI013}(P) = \mathbf{S}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{S} \circ \mathbf{CI013}(P) && \{\text{Definition of } \mathbf{S}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{CI013}(P)\\
&\{\text{Conjunctive healthiness conditions } \mathbf{CI0}\text{, } \mathbf{CI1} \text{ and } \mathbf{CI3} \text{ and } tr \text{ and } tr' \text{ not free in them}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{CI013} \circ \mathbf{R2}(P)\\
& && \{\text{Definition of } \mathbf{R3_T}\text{, distributivity of } \mathbf{CI013} \text{ and idempotent}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) && \{\text{Definition of } \mathbf{S}\}\\
&= \mathbf{S}(P)
\end{aligned}
$$

□

Lemma L.3.6.27

$$\mathbf{S} \circ \mathbf{CIB}(P) = \mathbf{S}(P)$$

Proof.

$\mathbf{S} \circ \mathbf{CIB}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{CIB}(P)$ {Lemma L.4.7.18}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{CIB} \circ \mathbf{R2}(P)$ {Lemma L.4.7.19}

$= \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{CIB} \circ \mathbf{R3_T} \circ \mathbf{CIB} \circ \mathbf{R2}(P)$

{Definition of $\mathbf{R3_T}$ and Lemma L.4.7.3 and **CIB**-idempotent}

$= \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{CIB} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Lemma L.4.7.19}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{S}(P)$

□

Lemma L.3.6.28

$$\mathbf{S} \circ \mathbf{ST_m}(P) = \mathbf{S}(P)$$

Proof.

$\mathbf{S} \circ \mathbf{ST_m}(P)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{S} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$ {Lemma L.5.0.1}

$= \mathbf{S} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$ {Lemma L.5.0.3}

$= \mathbf{S} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$ {Lemma L.5.0.2}

$= \mathbf{S} \circ \mathbf{R2}(P)$ {Definition of $\mathbf{S}$ and $\mathbf{R2}$ idempotent}

$= \mathbf{S}(P)$

□

Lemma L.3.6.29

$$\mathbf{S}(P \vee Q) = \mathbf{S}(P) \vee \mathbf{S}(Q)$$

Proof.

$$
\begin{array}{ll}
\mathbf{S}(P \vee Q) & \{\text{Definition of } \mathbf{S}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P \vee Q) & \{\text{Distributivity of } \mathbf{R2}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \vee \mathbf{R2}(Q)) & \{\text{Distributivity of } \mathbf{R3_T}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \left(\begin{array}{l} \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \\
\qquad \{\text{Distributivity of conjunctive healthiness conditions } \mathbf{CI0}, \mathbf{CI1} \text{ and } \mathbf{CI3}\} & \\
= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Lemma L.4.7.3}\} \\
= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \\
\qquad \{\text{Lemmas L.2.2.6 and L.2.3.19 and conjunctive healthiness condition } \mathbf{R0_T}\} & \\
= \left(\begin{array}{l} \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Definition of } \mathbf{S}\} \\
= \mathbf{S}(P) \vee \mathbf{S}(Q) &
\end{array}
$$

□

Lemma L.3.6.30 *Provided tr_C and tr'_C are not free in Q,*

$$\mathbf{ST_m}(P \lhd Q \rhd R) = \mathbf{ST_m}(P) \lhd \mathbf{R2_T} \circ \mathbf{R2}(Q) \rhd \mathbf{ST_m}(R)$$

Proof.

$$
\begin{array}{ll}
\mathbf{ST_m}(P) \lhd \mathbf{R2_T} \circ \mathbf{R2}(Q) \rhd \mathbf{ST_m}(R) & \{\text{Definition of conditional}\} \\
= \left(\begin{array}{l} (\mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(P)) \\ \vee \\ (\neg\, \mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(R)) \end{array} \right) & \{\text{Definition of } \mathbf{R2} \text{ and } \mathbf{R2_T}\}
\end{array}
$$

$$= \left(\begin{array}{l} (\mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(P)) \\ \vee \\ (\mathbf{R2_T} \circ \mathbf{R2}(\neg\ Q) \wedge \mathbf{ST_m}(R)) \end{array} \right) \quad \{\text{Assumption and Lemma L.5.0.9}\}$$

$$= \mathbf{ST_m}(Q \wedge P) \vee \mathbf{ST_m}(\neg\ Q \wedge R) \quad \{\text{Lemma L.5.0.12}\}$$

$$= \mathbf{ST_m}((Q \wedge P) \vee (\neg\ Q \wedge R)) \quad \{\text{Definition of conditional}\}$$

$$= \mathbf{ST_m}(P \lhd Q \rhd R)$$

□

Lemma L.3.6.31

$$ok \wedge \neg\ \mathbf{ST_m} \circ \mathbf{H1}(P) = ok \wedge \neg\ \mathbf{ST_m}(P)$$

Proof.

$ok \wedge \neg\ \mathbf{ST_m} \circ \mathbf{H1}(P)$ {Definition of **H1**}

$= ok \wedge \neg\ \mathbf{ST_m}(ok \Rightarrow P)$ {Predicate calculus}

$= ok \wedge \neg\ \mathbf{ST_m}(\neg\ ok \vee P)$ {Lemma L.5.0.12}

$= ok \wedge \neg\ (\mathbf{ST_m}(\neg\ ok) \vee \mathbf{ST_m}(P))$ {Predicate calculus}

$= ok \wedge \neg\ \mathbf{ST_m}(\neg\ ok) \wedge \neg\ \mathbf{ST_m}(P)$ {Lemma L.5.0.10}

$= ok \wedge \neg\ (\neg\ ok \wedge \mathbf{ST_m}(true)) \wedge \neg\ \mathbf{ST_m}(P)$ {Predicate calculus}

$= ok \wedge (ok \vee \neg\ \mathbf{ST_m}(true)) \wedge \neg\ \mathbf{ST_m}(P)$ {Predicate calculus: absorption law}

$= ok \wedge \neg\ \mathbf{ST_m}(P)$

□

Lemma L.3.6.32 *Provided tr, tr', tr_C, tr'_C, tr_T and tr'_T are not free in P,*

$$\mathbf{ST_m}(P \wedge Q) = P \wedge \mathbf{ST_m}(Q)$$

Proof.

$\mathbf{ST_m}(P \wedge Q)$ {Lemma L.5.0.11}

$= \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q)$ {Assumption and Lemma L.5.0.10}

$= P \wedge \mathbf{ST_m}(true) \wedge \mathbf{ST_m}(Q)$ {Lemma L.5.0.11}

$= P \wedge \mathbf{ST_m}(\mathit{true} \wedge Q)$ {Predicate calculus}

$= P \wedge \mathbf{ST_m}(Q)$

□

Lemma L.3.6.33 *Provided tr_C, tr'_C are not free in Q,*

$$\mathbf{ST_m}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q) = \mathbf{ST_m}(P \wedge Q)$$

Proof.

$\mathbf{ST_m}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q)$
{Conjunctive healthiness condition $\mathbf{R0_T}$}

$= \mathbf{R0_T}(\mathbf{R12_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q))$
{Conjunctive healthiness condition $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T}(\mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q))$ {Lemma L.2.3.18}

$= \mathbf{R012_T}(\mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$ {Assumption and Lemma L.4.7.5}

$= \mathbf{R012_T} \circ \mathbf{CIB}(\mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI0**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0}(\mathbf{CI12} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI1**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \circ \mathbf{CI1}(\mathbf{CI3} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI1**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \wedge \mathbf{R2}(Q))$ {Distributivity of **R2**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \wedge Q)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{ST_m}(P \wedge Q)$

□

Lemma L.3.6.34 *Provided tr, tr', tr_C, tr'_C, tr_T and tr'_T are not free in P,*

$$\mathbf{ST_m}(P) = P \wedge \mathbf{ST_m}(\mathit{true})$$

Proof.

$$
\begin{aligned}
&\mathbf{ST_m}(P) && \{\text{Definition of } \mathbf{ST_m}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) && \{\text{Assumption and definition of } \mathbf{R2}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(P) && \{\text{Predicate calculus}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(P \wedge \mathit{true}) && \{\text{Conjunctive healthiness condition } \mathbf{CI3}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01}(P \wedge \mathbf{CI3}(\mathit{true})) && \{\text{Conjunctive healthiness condition } \mathbf{CI1}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0}(P \wedge \mathbf{CI1} \circ \mathbf{CI3}(\mathit{true})) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{CI0}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB}(P \wedge \mathbf{CI013}(\mathit{true})) && \{\text{Assumption and Lemma L.4.7.2}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{CI013}(\mathit{true}) \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R1_C}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{R1_C}(\mathbf{CI013}(\mathit{true}) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true}))) \\
& && \{\text{Definition of } \mathbf{CI0}, \mathbf{CI1}, \mathbf{CI3} \text{ and } \mathbf{R2_C}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI013}(\mathit{true}) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true}))) \\
& && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI013}(\mathit{true}) \wedge \mathbf{CI4}(\mathit{true}))) \\
& && \{\text{Conjunctive healthiness conditions and predicate calculus}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4} \circ \mathbf{CI013}(\mathit{true})) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{R012_T}(P \wedge \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true})) && \{\text{Distributivity of } \mathbf{R2_T}\}\\
&= \mathbf{R01_T}(\mathbf{R2_T}(P) \wedge \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true})) \\
& && \{\text{Definition of } \mathbf{R2_T} \text{ and assumption}\}\\
&= \mathbf{R01_T}(P \wedge \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true})) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R1_T}\}\\
&= \mathbf{R0_T}(P \wedge \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true})) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R1_T}\}\\
&= P \wedge \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true}) && \{\text{Definition of } \mathbf{ST_m}\}\\
&= P \wedge \mathbf{ST_m}(\mathit{true})
\end{aligned}
$$

□

Lemma L.3.6.35

$$\mathbf{ST_m}(P \wedge Q) = \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q)$$

Proof.

$$\begin{array}{ll}
\mathbf{ST_m}(P \wedge Q) & \{\text{Definition of } \mathbf{ST_m}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \wedge Q) & \{\text{Distributivity of } \mathbf{R2}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \wedge \mathbf{R2}(Q)) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{CI3}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\begin{array}{l} \mathbf{CI3} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{CI1}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \left(\begin{array}{l} \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{CI0}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Lemma L.4.7.4}\} \\
= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Lemma L.2.3.18}\} \\
= \mathbf{R01_T} \left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Lemma L.2.2.4}\} \\
= \mathbf{R0_T} \left(\begin{array}{l} \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{R0_T}\}
\end{array}$$

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R0_T} \circ \mathbf{R2_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) \qquad \{\text{Definition of } \mathbf{ST_m}\}$$

$$= \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q)$$

□

Lemma L.3.6.36

$$\mathbf{ST_m}(P \vee Q) = \mathbf{ST_m}(P) \vee \mathbf{ST_m}(Q)$$

Proof.

$\mathbf{ST_m}(P \vee Q)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \vee Q)$ {Distributivitu of $\mathbf{R2}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \vee \mathbf{R2}(Q))$

{Conjunctive healthiness condition **CI3**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\begin{array}{l} \mathbf{CI3} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition **CI1**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \left(\begin{array}{l} \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition **CI0**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) \qquad \{\text{Lemma L.4.7.3}\}$$

$$= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) \qquad \{\text{Lemma L.2.3.19}\}$$

$$= \mathbf{R01_T} \left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) \qquad \{\text{Lemma L.2.2.6}\}$$

$$= \mathbf{R0_T} \left(\begin{array}{l} \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{R0_T}$}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Definition of $\mathbf{ST_m}$}

$$= \mathbf{ST_m}(P) \vee \mathbf{ST_m}(Q)$$

□

Lemma L.3.6.37

$$ok \wedge \mathbf{ST_m}(\neg\ ok) = false$$

Proof.

$ok \wedge \mathbf{ST_m}(\neg\ ok)$ {Lemma L.5.0.10}

$= ok \wedge \neg\ ok \wedge \mathbf{ST_m}(true)$ {Predicate calculus}

$= false$

□

This requires commutativity, or partial commutativity of healthiness conditions. Other calculations to be done include:

Lemma L.3.6.38

$$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{CI013}(P)$$

Proof.

$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{CI013}(P)$ {Definition of **CI013**}

$$= \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \\ \wedge \\ wait_T = wait \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right)$$

{One-point rule}

$$= \exists\, tr_T, tr'_T, wait'_T \bullet \left(\begin{array}{l} P[wait/wait_T] \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \\ \wedge \\ (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right)$$

□

Lemma L.3.6.39

$$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{R1_T} \circ \mathbf{R2_T}(P) \quad = \exists\, tr_T, wait_T, wait'_T \bullet \mathbf{R1_T} \left(\begin{array}{l} P \left[\begin{array}{l} \langle(\langle\rangle, snd \circ last \\ dif_T(front(tr_T \end{array} \right. \\ \wedge \\ fst \circ last(tr_T) \leq fs \end{array} \right.$$

Proof.

$$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{R1_T} \circ \mathbf{R2_T}(P) \qquad \{\text{Definition of } \mathbf{R2_T}\}$$

$$= \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{R1_T} \left(\; P[\langle(\langle\rangle, snd \circ last(tr_T))\rangle, dif_T(tr'_T, tr_T)/tr_T, tr'_T] \; \right)$$

{Lemma L.3.6.40}

$$= \exists\, tr_T, wait_T, wait'_T \bullet \mathbf{R1_T} \left(\begin{array}{l} P \left[\begin{array}{l} \langle (\langle\rangle, snd \circ last(tr_T)) \rangle / tr_T \\ dif_T(tr'_T, tr_T) / tr'_T \\ front(tr_T) \frown z / tr'_T \end{array} \right] \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(z) \end{array} \right)$$

{Substitution}

$$= \exists\, tr_T, wait_T, wait'_T \bullet \mathbf{R1_T} \left(\begin{array}{l} P \left[\begin{array}{l} \langle (\langle\rangle, snd \circ last(tr_T)) \rangle / tr_T \\ dif_T(front(tr_T) \frown z, tr_T) / tr'_T \end{array} \right] \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(z) \end{array} \right)$$

□

Lemma L.3.6.40

$$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{R1_T}(P)$$
$$=$$
$$\exists\, tr_T, wait_T, wait'_T, z \bullet \left(\begin{array}{l} P[front(tr_T) \frown z / tr'_T] \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(z) \end{array} \right)$$

Proof.

$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{R1_T}(P)$ {Definition of $\mathbf{R1_T}$}

$$= \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \left(\begin{array}{l} P \wedge front(tr_T) < tr'_T \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

{Property of sequences}

$$= \exists\, tr_T, tr'_T, wait_T, wait'_T, z \bullet \left(\begin{array}{l} P \wedge front(tr_T) \frown z = tr'_T \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

{One-point rule}

$$= \exists\, tr_T, wait_T, wait'_T, z \bullet \left(\begin{array}{l} P[front(tr_T) \frown z / tr'_T] \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(front(tr_T) \frown z - front(tr_T)) \end{array} \right)$$

{Property of sequences}

$$= \exists\, tr_T, wait_T, wait'_T, z \bullet \left(\begin{array}{l} P[front(tr_T) \frown z / tr'_T] \wedge z \neq \langle\rangle \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(z) \end{array} \right)$$

□

Lemma L.3.6.41 *Provided ok' and $wait$ are not free in P, and tr_C and tr'_C are not free in P and Q,*

$$\exists\, tr_T, tr'_T, wait_T, wait'_T, tr_C, tr'_C \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$$

Proof.

$\exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \mathbf{CITR} \circ \mathbf{R}(P \vdash Q)$ {Lemma L.3.6.7}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T, tr_C, tr'_C \bullet \\ \\ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R012_T}$ and Lemma L.2.2.12}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T, tr_C, tr'_C \bullet \\ \\ \mathbf{R012_T} \circ \mathbf{R1_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right. \end{array} \right.$$

{Predicate calculus}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \exists\, tr_C, tr'_C \bullet \\ \\ \mathbf{R012_T} \circ \mathbf{R1_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\, wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right. \end{array} \right.$$

{Lemmas L.2.1.3, L.2.2.26 and L.2.3.4}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \\ \mathbf{R012_T} \circ \mathbf{R1_T} \left(\exists\, tr_C, tr'_C \bullet \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \end{array} \right. \right. \end{array} \right.$$

{Assumption and Lemma L.4.7.8}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \\ \mathbf{R012_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right) \end{array} \right)$$

{Definition of $\mathbf{R012_T}$ and Lemma L.2.2.12}

$$= \left(\begin{array}{l} \exists\, tr_T, tr'_T, wait_T, wait'_T \bullet \\ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right) \end{array} \right)$$

{Definition of L_W and L_T}

$$= L_T \circ L_W \circ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

{Lemma L.3.6.47}

$$= L_T \circ \mathbf{R012_T} \circ L_W \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \left(\begin{array}{l} P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

□

And also the introduction of untimed variables when coming from the timed world, that is, applying **CITR** to a timed reactive design, where the untimed variables are originally not free.

We can split the mapping L_{TW} according to the variables which are being hidden, as follows.

Definition 2

$$L_T(P) = \exists\, tr_T, tr'_T \bullet P$$

$$L_W(P) = \exists\, wait_T, wait'_T \bullet P$$
$$L_{TW}(P) = L_T \circ L_W(P)$$

Lemma L.3.6.42

$$L_W \circ \mathbf{CI3}(P) = P[wait/wait_T][false/wait'_T] \vee (P[wait/wait_T][true/wait'_T] \wedge wait')$$

Proof.

$$\begin{array}{ll}
L_W \circ \mathbf{CI3}(P) & \{\text{Definition of } L_W \text{ and } \mathbf{CI3}\} \\
= \exists\, wait_T, wait'_T \bullet (P \wedge wait_T = wait \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)) & \{\text{One-point rule}\} \\
= \exists\, wait'_T \bullet (P[wait/wait_T] \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)) & \{\text{Case analysis of } wait'_T\} \\
= P[wait/wait_T][false/wait'_T] \vee (P[wait/wait_T][true/wait'_T] \wedge wait') &
\end{array}$$

□

Lemma L.3.6.43

$$L_W \circ \mathbf{CI3} \circ \mathbf{R3_T}(P)$$
$$=$$
$$L_W \circ \mathbf{CI3}(\mathit{II}_A) \lhd wait_T \rhd L_W \circ \mathbf{CI3}(P)$$

Proof.

$$\begin{array}{ll}
L_W \circ \mathbf{CI3} \circ \mathbf{R3_T}(P) & \{\text{Definition of } \mathbf{R3_T}\} \\
= L_W \circ \mathbf{CI3}(\mathit{II}_A \lhd wait_T \rhd P) & \{\text{Lemma L.3.6.42}\} \\
= \left(\begin{array}{l} (\mathit{II}_A \lhd wait_T \rhd P)[wait/wait_T][false/wait'_T] \\ \vee \\ ((\mathit{II}_A \lhd wait_T \rhd P)[wait/wait_T][true/wait'_T] \wedge wait') \end{array} \right) & \{\text{Substitution}\} \\
= \left(\begin{array}{l} (\mathit{II}_A[wait/wait_T][false/wait'_T] \lhd wait \rhd P[wait/wait_T][false/wait'_T]) \\ \vee \\ ((\mathit{II}_A[wait/wait_T][true/wait'_T] \lhd wait \rhd P[wait/wait_T][true/wait'_T]) \wedge wait') \end{array} \right) & \\
\{\text{Property of conditional}\} &
\end{array}$$

$$= \left(\begin{array}{l} (\mathit{II}_A[wait/wait_T][false/wait'_T] \lhd wait \rhd P[wait/wait_T][false/wait'_T]) \\ \vee \\ (\mathit{II}_A[wait/wait_T][true/wait'_T] \wedge wait') \lhd wait \rhd (P[wait/wait_T][true/wait'_T] \wedge wait') \end{array} \right)$$

{Property of conditional}

$$= \left(\begin{array}{l} (\mathit{II}_A[wait/wait_T][false/wait'_T] \vee (\mathit{II}_A[wait/wait_T][true/wait'_T] \wedge wait')) \\ \lhd wait \rhd \\ (P[wait/wait_T][false/wait'_T] \vee (P[wait/wait_T][true/wait'_T] \wedge wait')) \end{array} \right)$$

{Lemma L.3.6.42}

$$= L_W \circ \mathbf{CI3}(\mathit{II}_A) \lhd wait_T \rhd L_W \circ \mathbf{CI3}(P)$$

□

Lemma L.3.6.44

$$L_W \circ \mathbf{R0_T}(P) = \mathbf{R0_T} \circ L_W(P)$$

Proof.

$L_W \circ \mathbf{R0_T}(P)$ {Definition of L_W}

$= \exists\, wait_T, wait'_T \bullet \mathbf{R0_T}(P)$ {Lemma L.2.1.3}

$= \mathbf{R0_T}(\exists\, wait_T, wait'_T \bullet P)$ {Definition of L_W}

$= \mathbf{R0_T} \circ L_W(P)$

□

Lemma L.3.6.45

$$L_W \circ \mathbf{R1_T}(P) = \mathbf{R1_T} \circ L_W(P)$$

Proof.

$L_W \circ \mathbf{R1_T}(P)$ {Definition of L_W}

$= \exists\, wait_T, wait'_T \bullet \mathbf{R1_T}(P)$ {Lemma L.2.2.26}

$= \mathbf{R1_T}(\exists\, wait_T, wait'_T \bullet P)$ {Definition of L_W}

$= \mathbf{R1_T} \circ L_W(P)$

□

Lemma L.3.6.46

$$L_W \circ \mathbf{R2_T}(P) = \mathbf{R2_T} \circ L_W(P)$$

Proof.

$$\begin{aligned}
&L_W \circ \mathbf{R2_T}(P) && \{\text{Definition of } L_W\}\\
&= \exists\, wait_T, wait'_T \bullet \mathbf{R2_T}(P) && \{\text{Lemma L.2.3.4}\}\\
&= \mathbf{R2_T}(\exists\, wait_T, wait'_T \bullet P) && \{\text{Definition of } L_W\}\\
&= \mathbf{R2_T} \circ L_W(P)
\end{aligned}$$

□

Lemma L.3.6.47

$$L_W \circ \mathbf{R012_T}(P) = \mathbf{R012_T} \circ L_W(P)$$

Proof.

$$\begin{aligned}
&L_W \circ \mathbf{R012_T}(P) && \{\text{Definition of } \mathbf{R012_T}\}\\
&= L_W \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P) && \{\text{Lemma L.3.6.44}\}\\
&= \mathbf{R0_T} \circ L_W \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P) && \{\text{Lemma L.3.6.45}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ L_W \circ \mathbf{R2_T}(P) && \{\text{Lemma L.3.6.46}\}\\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ L_W(P) && \{\text{Definition of } \mathbf{R012_T}\}\\
&= \mathbf{R012_T} \circ L_W(P)
\end{aligned}$$

□

Lemma L.3.6.48

$$L_W \circ \mathbf{CI0}(P) = \mathbf{CI0} \circ L_W(P)$$

Proof.

$$
\begin{aligned}
& L_W \circ \mathbf{CI0}(P) && \{\text{Definition of } \mathbf{CI0}\} \\
& = L_W \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Lemma L.3.6.53}\} \\
& = \left(\begin{array}{l} L_W(P) \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Definition of } \mathbf{CI0}\} \\
& = \mathbf{CI0} \circ L_W(P)
\end{aligned}
$$

□

Lemma L.3.6.49

$$L_W \circ \mathbf{CI1}(P) = \mathbf{CI1} \circ L_W(P)$$

Proof.

$$
\begin{aligned}
& L_W \circ \mathbf{CI1}(P) && \{\text{Definition of } \mathbf{CI1}\} \\
& = L_W(P \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T)) && \{\text{Lemma L.3.6.53}\} \\
& = L_W(P) \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) && \{\text{Definition of } \mathbf{CI1}\} \\
& = \mathbf{CI1} \circ L_W(P)
\end{aligned}
$$

□

Lemma L.3.6.50

$$L_W \circ \mathbf{CI0} \circ \mathbf{CI1}(P) = \mathbf{CI0} \circ \mathbf{CI1} \circ L_W(P)$$

Proof.

$$
\begin{aligned}
& L_W \circ \mathbf{CI0} \circ \mathbf{CI1}(P) && \{\text{Lemma L.3.6.48}\} \\
& = \mathbf{CI0} \circ L_W \circ \mathbf{CI1}(P) && \{\text{Lemma L.3.6.49}\} \\
& = \mathbf{CI0} \circ \mathbf{CI1} \circ L_W(P)
\end{aligned}
$$

□

Lemma L.3.6.51 *Provided $wait_T$ and $wait'_T$ are not free in P,*

$$L_W(P) = P$$

Proof.

$L_W(P)$ {Definition of L_W}
$= \exists\, wait_T, wait'_T \bullet P$ {Assumption and predicate calculus}
$= P$

□

Lemma L.3.6.52 *Provided tr_T, tr'_T is not free in P,*

$$L_T(P \wedge Q) = P \wedge L_T(Q)$$

Proof.

$L_T(P \wedge Q)$ {Definition of L_T}
$= \exists\, tr_T, tr'_T \bullet P \wedge Q$ {Assumption and predicate calculus}
$= P \wedge \exists\, tr_T, tr'_T \bullet Q$ {Definition of L_T}
$= P \wedge L_T(Q)$

□

Lemma L.3.6.53 *Provided $wait_T$, $wait'_T$ is not free in P,*

$$L_W(P \wedge Q) = P \wedge L_W(Q)$$

Proof.

$L_W(P \wedge Q)$ {Definition of L_W}
$= \exists\, wait_T, wait'_T \bullet P \wedge Q$ {Assumption and predicate calculus}
$= P \wedge \exists\, wait_T, wait'_T \bullet Q$ {Definition of L_W}

$= P \wedge L_W(Q)$

□

Lemma L.3.6.54 *Provided tr_T, tr'_T, $wait_T$ and $wait'_T$ are not free in P,*

$$L_{TW} \circ \mathbf{R0_T}(P) = P$$

Proof.

$L_{TW} \circ \mathbf{R0_T}(P)$ {Definition of L_{TW}}
$= L_T \circ L_W \circ \mathbf{R0_T}(P)$ {Definition of $\mathbf{R0_T}$}
$= L_T \circ L_W \circ \mathbf{TR_0} \circ \mathbf{TR_1}(P)$ {Definition of $\mathbf{TR_0}$ and $\mathbf{TR_1}$}
$= L_T \circ L_W(P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T)$ {Assumption and Lemma L.3.6.51}
$= L_T(P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T)$ {Assumption and Lemma L.3.6.52}
$= P \wedge L_T(\#tr_T > 0 \wedge \#tr_T \leq \#tr'_T)$ {Definition of L_T}
$= P \wedge (\exists\, tr_T, tr'_T \bullet \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T)$ {Predicate calculus}
$= P$

□

Lemma L.3.6.55 *Provided tr_T, tr'_T, $wait_T$ and $wait'_T$ are not free in P,*

$$L_{TW} \circ \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{CI1}(P)$$

Proof.

$L_{TW} \circ \mathbf{R1_T}(P)$ {Definition of L_{TW}}
$= L_T \circ L_W \circ \mathbf{R1_T}(P)$ {Definition of $\mathbf{R1_T}$}
$= L_T \circ L_W(P \wedge \mathit{Expands}_A(tr_T, tr'_T))$ {Assumption and Lemma L.3.6.53}
$= L_T(P \wedge L_W(\mathit{Expands}_A(tr_T, tr'_T)))$ {Assumption and Lemma L.3.6.52}
$= P \wedge L_T \circ L_W(\mathit{Expands}_A(tr_T, tr'_T))$
{Definition of $\mathit{Expands}_A$ and Lemma L.3.6.51}
$= P \wedge L_T(\mathit{Expands}_A(tr_T, tr'_T))$

□

Lemma L.3.6.56

$$L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P)$$
$$=$$
$$\mathbf{R1} \circ \mathbf{L_T} \circ \mathbf{CI0} \circ \mathbf{CI1}(P)$$

Proof.

$$L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P) \qquad \{\text{Definition of } \mathbf{CI0} \text{ and } \mathbf{CI1}\}$$

$$= L_T \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \end{array} \right) \qquad \{\text{Definition of } L_T\}$$

$$= \exists\, tr_T, tr'_T \bullet \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \end{array} \right)$$

$$\{\text{Predicate calculus}\}$$

$$= \left(\begin{array}{l} tr \leq tr' \wedge \exists\, tr_T, tr'_T \bullet \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \end{array} \right) \end{array} \right)$$

$$\{\text{Definition of } \mathbf{R1}, \mathbf{CI0} \text{ and } \mathbf{CI1}\}$$

$$= \mathbf{R1} \circ \mathbf{L_T} \circ \mathbf{CI0} \circ \mathbf{CI1}(P)$$

□

Lemma L.3.6.57 *Provided tr_T and tr'_T are not free in P,*

$$L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P) = \mathbf{R1}(P)$$

Proof.

$$
\begin{array}{ll}
L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P) & \{\text{Predicate calculus}\} \\
= L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P \wedge \mathit{true}) & \{\text{Conjunctive healthiness conditions } \mathbf{CI0} \text{ and } \mathbf{CI1}\} \\
= L_T(P \wedge \mathbf{CI0} \circ \mathbf{CI1}(\mathit{true})) & \{\text{Assumption and Lemma L.3.6.52}\} \\
= P \wedge L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(\mathit{true}) & \{\text{Lemma L.3.6.58}\} \\
= P \wedge \mathbf{R1}(\mathit{true}) & \{\text{Conjunctive healthiness condition } \mathbf{R1}\} \\
= \mathbf{R1}(P) &
\end{array}
$$

□

Lemma L.3.6.58

$$L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(\mathit{true}) = \mathbf{R1}(\mathit{true})$$

Proof. (Implication)

$$
\begin{array}{ll}
L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(\mathit{true}) & \{\text{Definition of } \mathbf{CI0} \text{ and } \mathbf{CI1}\} \\
= L_T \left(\begin{array}{l} (tr' - tr) = \mathit{Flat}(tr'_T) - \mathit{Flat}(tr_T) \\ \wedge \\ tr \leq tr' \wedge \mathit{Flat}(tr_T) \leq \mathit{Flat}(tr'_T) \\ \wedge \\ \mathit{ref} = \mathit{snd} \circ \mathit{last}(tr_T) \wedge \mathit{ref}' = \mathit{snd} \circ \mathit{last}(tr'_T) \end{array} \right) & \{\text{Definition of } L_T\} \\
= \exists\, tr_T, tr'_T \bullet \left(\begin{array}{l} (tr' - tr) = \mathit{Flat}(tr'_T) - \mathit{Flat}(tr_T) \\ \wedge \\ tr \leq tr' \wedge \mathit{Flat}(tr_T) \leq \mathit{Flat}(tr'_T) \\ \wedge \\ \mathit{ref} = \mathit{snd} \circ \mathit{last}(tr_T) \wedge \mathit{ref}' = \mathit{snd} \circ \mathit{last}(tr'_T) \end{array} \right) & \\
& \{\text{Predicate calculus}\} \\
\Rightarrow tr \leq tr' & \{\text{Definition of } \mathbf{R1}\} \\
= \mathbf{R1}(\mathit{true}) &
\end{array}
$$

□

Proof. (Reverse implication)

$$
\begin{aligned}
&\mathbf{R1}(true) && \{\text{Definition of } \mathbf{R1}\}\\
&= tr \leq tr' && \{\text{Predicate calculus and definition of } Flat\}\\
&= \left(\begin{array}{l} tr \leq tr' \\ \wedge \\ tr = Flat(\langle (tr, ref) \rangle) \wedge tr' = Flat(\langle (tr', ref) \rangle) \\ \wedge \\ ref = snd \circ last(\langle (tr, ref) \rangle) \wedge ref = snd \circ last(\langle (tr', ref') \rangle) \end{array}\right) && \{\text{Property of sequences}\}\\
&= \left(\begin{array}{l} tr \leq tr' \wedge tr' - tr = Flat(\langle (tr', ref) \rangle) - Flat(\langle (tr, ref) \rangle) \\ \wedge \\ tr = Flat(\langle (tr, ref) \rangle) \wedge tr' = Flat(\langle (tr', ref) \rangle) \\ \wedge \\ ref = snd \circ last(\langle (tr, ref) \rangle) \wedge ref = snd \circ last(\langle (tr', ref') \rangle) \end{array}\right) && \{\text{Predicate calculus}\}\\
&= \exists\, tr_T, tr'_T \bullet \left(\begin{array}{l} tr \leq tr' \wedge tr' - tr = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr = Flat(tr_T) \wedge tr' = Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref = snd \circ last(tr'_T) \\ \wedge \\ tr'_T = \langle (tr', ref) \rangle \wedge tr_T = \langle (tr, ref) \rangle \end{array}\right) && \{\text{Predicate calculus}\}\\
&\Rightarrow \exists\, tr_T, tr'_T \bullet \left(\begin{array}{l} tr \leq tr' \wedge tr' - tr = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr = Flat(tr_T) \wedge tr' = Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref = snd \circ last(tr'_T) \end{array}\right) && \{\text{Definition of } \mathbf{CI0} \text{ and } \mathbf{CI1}\}\\
&= \exists\, tr_T, tr'_T \bullet \mathbf{CI0} \circ \mathbf{CI1}(P) && \{\text{Definition of } L_T\}\\
&= L_T \circ \mathbf{CI0} \circ \mathbf{CI1}(P)
\end{aligned}
$$

□

Chapter 4

Coupling Invariants

4.1 Results on CI0

Lemma L.4.1.1

$$\mathbf{CI0}(P)^o_f = \mathbf{CI0}(P^o_f)$$

Proof.

$$\begin{aligned}
&\mathbf{CI0}(P)^o_f && \{\text{Definition of } \mathbf{CI0}\}\\
&= \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right)^o_f && \{\text{Substitution}\}\\
&= \left(\begin{array}{l} P^o_f \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Definition of } \mathbf{CI0}\}\\
&= \mathbf{CI0}(P^o_f)
\end{aligned}$$

□

Lemma L.4.1.2

$$\mathbf{CI0}(P)^f_f = \mathbf{CI0}(P^f_f)$$

Proof.

$$
\begin{aligned}
&\mathbf{CI0}(P)^f_f && \{\text{Definition of } \mathbf{CI0}\}\\
&= \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right)^f_f && \{\text{Substitution}\}\\
&= \left(\begin{array}{l} P^f_f \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Definition of } \mathbf{CI0}\}\\
&= \mathbf{CI0}(P^f_f)
\end{aligned}
$$

□

Lemma L.4.1.3

$$\mathbf{CI0} \circ \mathbf{R1}(P) = \mathbf{CI0}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{CI0} \circ \mathbf{R1}(P) && \{\text{Definition of } \mathbf{CI0}\}\\
&= \left(\begin{array}{l} \mathbf{R1}(P) \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Definition of } \mathbf{R1}\}\\
&= \left(\begin{array}{l} P \wedge tr \leq tr' \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Predicate calculus}\}\\
&= \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) && \{\text{Definition of } \mathbf{CI0}\}\\
&= \mathbf{CI0}(P)
\end{aligned}
$$

□

Lemma L.4.1.4

$$\mathbf{CI0}(tr' = tr) = \mathbf{CI0}(Flat(tr'_T) = Flat(tr_T))$$

Proof.

$\mathbf{CI0}(tr' = tr)$ {Definition of **CI0**}

$= tr' - tr = Flat(tr'_T) - Flat(tr_T) \wedge Flat(tr_T) \leq Flat(tr'_T) \wedge tr' = tr \wedge tr \leq tr'$
{Property of sequences and transitivity of equality}

$= tr' - tr = Flat(tr'_T) - Flat(tr_T) \wedge Flat(tr'_T) = Flat(tr_T) \wedge Flat(tr_T) \leq Flat(tr'_T) \wedge tr \leq tr'$
{Definition of **CI0**}

$= \mathbf{CI0}(Flat(tr'_T) = Flat(tr_T))$

□

Lemma L.4.1.5

$$\mathbf{CI0}(tr \leq tr') = \mathbf{CI0}(true)$$

Proof.

$\mathbf{CI0}(tr \leq tr')$ {Definition of **CI0**}

$= tr \leq tr' \wedge tr' - tr = Flat(tr'_T) - Flat(tr_T) \wedge Flat(tr_T) \leq Flat(tr'_T) \wedge tr \leq tr'$
{Predicate calculus}

$= tr' - tr = Flat(tr'_T) - Flat(tr_T) \wedge Flat(tr_T) \leq Flat(tr'_T) \wedge tr \leq tr'$
{Definition of **CI0**}

$= \mathbf{CI0}(true)$

□

Lemma L.4.1.6

$$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0}(P) = \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{R2_T}(P)$$

Proof.

$$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0}(P) \qquad \{\text{Definition of } \mathbf{CI0}\}$$

$$= \mathbf{R1_T} \circ \mathbf{R2_T} \left(\begin{array}{l} P \wedge (tr' - tr) = \mathit{Flat}(tr'_T) - \mathit{Flat}(tr_T) \\ \wedge \\ tr \leq tr' \wedge \mathit{Flat}(tr_T) \leq \mathit{Flat}(tr'_T) \end{array} \right)$$

$$\{\text{Distributivity of } \mathbf{R2_T} \text{ (Lemma L.2.3.18)}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge \mathbf{R2_T}((tr' - tr) = \mathit{Flat}(tr'_T) - \mathit{Flat}(tr_T)) \\ \wedge \\ \mathbf{R2_T}(tr \leq tr') \wedge \mathbf{R2_T}(\mathit{Flat}(tr_T) \leq \mathit{Flat}(tr'_T)) \end{array} \right)$$

$$\{\text{Lemma L.2.3.12}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge \mathbf{R2_T}((tr' - tr) = \mathit{Flat}(tr'_T) - \mathit{Flat}(tr_T)) \\ \wedge \\ tr \leq tr' \wedge \mathbf{R2_T}(\mathit{Flat}(tr_T) \leq \mathit{Flat}(tr'_T)) \end{array} \right)$$

$$\{\text{Definition of } \mathbf{R2_T} \text{ and substitution}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge (tr' - tr) = \mathit{Flat}(\mathit{dif}_T(tr'_T, tr_T)) - \mathit{Flat}(\langle(\langle\rangle, \mathit{snd} \circ \mathit{last}(tr_T))\rangle) \\ \wedge \\ tr \leq tr' \\ \wedge \\ \mathit{Flat}(\langle(\langle\rangle, \mathit{snd} \circ \mathit{last}(tr_T))\rangle) \leq \mathit{Flat}(\mathit{dif}_T(tr'_T, tr_T)) \end{array} \right)$$

$$\{\text{Definition of } \mathit{Flat}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge (tr' - tr) = \mathit{Flat}(\mathit{dif}_T(tr'_T, tr_T)) - \langle\rangle \\ \wedge \\ tr \leq tr' \\ \wedge \\ \langle\rangle \leq \mathit{Flat}(\mathit{dif}_T(tr'_T, tr_T)) \end{array} \right)$$

$$\{\text{Property of sequences}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge (tr' - tr) = \mathit{Flat}(\mathit{dif}_T(tr'_T, tr_T)) \\ \wedge \\ tr \leq tr' \end{array} \right) \qquad \{\text{Lemma 24}\}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \end{array} \right)$$

{Definition of $\mathbf{R1_T}$ and Lemma L.2.2.22}

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right)$$

{Definition of **CI0**}

$$= \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{R2_T}(P)$$

□

4.1.0.1 CI013

Lemma L.4.1.7 ***Isabelle proof available: Section 9.5***

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013} \circ \mathbf{R3}(P)$ {Lemma L.4.1.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T}(\mathbf{TR3} \circ \mathbf{CI013}(\mathit{II}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P))$

{Property of conditional}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} (\mathbf{TR3} \circ \mathbf{CI013}(\mathit{II}_{rea}) \wedge wait_T) \\ \lhd wait_T \rhd \\ \mathbf{CI013}(P) \end{array} \right)$$

{Conjunctive healthiness condition **CI013** and **TR3** and predicate calculus}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} \mathbf{TR3} \circ \mathbf{CI013}(\mathit{II}_{rea} \wedge wait_T) \\ \lhd wait_T \rhd \\ \mathbf{CI013}(P) \end{array} \right)$$

{Property of conditional and distributivity of $\mathbf{R0_T}$ and $\mathbf{R1_T}$}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013}(\mathit{II}_{rea} \wedge wait_T) \\ \lhd wait_T \rhd \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(P) \end{array} \right) \qquad \{\text{Lemma L.4.1.10}\}$$

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \mathit{II}_A) \\ \lhd wait_T \rhd \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(P) \end{array} \right)$$

$\{$Property of conditional and distributivity of $\mathbf{R0_T}$, $\mathbf{R1_T}$ and $\mathbf{CI013}\}$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \left(\begin{array}{l} (wait_T \wedge \mathit{II}_A) \\ \lhd wait_T \rhd \\ P \end{array} \right) \qquad \{\text{Property of conditional}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(\mathit{II}_A \lhd wait_T \rhd P) \qquad \{\text{Definition of } \mathbf{R3_T}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$$

□

Lemma L.4.1.8

$$\mathbf{CI013}(\mathit{II}_{rea}) = \mathbf{CI013} \left(\neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait' = wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \right)$$

Proof.

$\mathbf{CI013}(\mathit{II}_{rea})$ $\{$Definition of $\mathit{II}_{rea}\}$

$$= \mathbf{CI013} \left(\begin{array}{l} (\neg\ ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge ref' = ref \wedge tr' = tr \wedge wait' = wait) \end{array} \right) \qquad \{\text{Definition of } \mathbf{CI013}\}$$

$$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \left(\begin{array}{l} (\neg\ ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge ref' = ref \wedge tr' = tr \wedge wait' = wait) \end{array} \right)$$

$\{$Commutativity of conjunctive healthiness conditions $\mathbf{CI0}$, $\mathbf{CI1}$ and $\mathbf{CI3}\}$

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{CI0}(tr \leq tr')) \\ \vee \\ (ok' \wedge ref' = ref \wedge \mathbf{CI0}(tr' = tr) \wedge wait' = wait) \end{array} \right)$$

{Lemmas L.4.1.4 and L.4.1.5}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{CI0}(true)) \\ \vee \\ \left(\begin{array}{l} ok' \wedge ref' = ref \\ \wedge \\ \mathbf{CI0}(Flat(tr'_T) = Flat(tr_T)) \wedge wait' = wait \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition **CI0**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge ref' = ref \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \wedge wait' = wait \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition **CI1**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge \mathbf{CI1}(ref' = ref) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \wedge wait' = wait \end{array} \right) \end{array} \right)$$

{Lemma L.4.2.4}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge \mathbf{CI1}(snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \wedge wait' = wait \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition **CI1**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge wait' = wait \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition **CI3**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge \mathbf{CI3}(wait' = wait) \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Definition of **CI3**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge \mathbf{CI3}(wait' = wait_T) \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition **CI3**}

$$= \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge wait' = wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Commutativity of conjunctive healthiness conditions **CI0**, **CI1** and **CI3**}

$$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \left(\begin{array}{l} (\neg\ ok) \\ \vee \\ \left(\begin{array}{l} ok' \wedge wait' = wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Definition of **CI013**}

$$= \mathbf{CI013} \left(\neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait' = wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T)) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \right)$$

□

Lemma L.4.1.9

$$\mathbf{TR3} \circ \mathbf{CI013}(\mathit{I\!I}_{rea} \wedge wait_T)$$
$$=$$
$$\mathbf{CI013} \left(wait_T \wedge \left(\neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait'_T = wait_T \wedge \#tr'_T = \#tr_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \right) \right)$$

Proof.

$\mathbf{TR3} \circ \mathbf{CI013}(\mathit{I\!I}_{rea} \wedge wait_T)$ {Conjunctive healthiness condition **CI013**}

$\mathbf{TR3}(\mathbf{CI013}(\mathit{I\!I}_{rea}) \wedge wait_T))$ {Lemma L.4.1.8}

$$= \mathbf{TR3}\left(\begin{array}{l} wait_T \wedge \mathbf{CI013}\left(\begin{array}{l} \neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait' = wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array}\right) \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{TR3} \circ \mathbf{CI013}\left(\begin{array}{l} (\neg\ ok \wedge wait_T) \\ \vee \\ \left(\begin{array}{l} ok' \wedge wait' = wait_T \wedge wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array}\right) \end{array}\right)$$

{Distributivity of **CI013** and **TR3**}

$$= \left(\begin{array}{l} \mathbf{TR3} \circ \mathbf{CI013}(\neg\ ok \wedge wait_T) \\ \vee \\ \mathbf{TR3} \circ \mathbf{CI013}\left(\begin{array}{l} ok' \wedge wait' = wait_T \wedge wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array}\right) \end{array}\right)$$

{Conjunctive healthiness condition **TR3**}

$$= \left(\begin{array}{l} \mathbf{CI013}(\mathbf{TR3}(\neg\ ok) \wedge wait_T) \\ \vee \\ \mathbf{TR3} \circ \mathbf{CI013}\left(\begin{array}{l} ok' \wedge wait' = wait_T \wedge wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array}\right) \end{array}\right)$$

{Lemma L.3.4.4}

$$= \left(\begin{array}{l} \mathbf{CI013}(\neg\ ok \wedge wait_T) \\ \vee \\ \mathbf{TR3} \circ \mathbf{CI013} \left(\begin{array}{l} ok' \wedge wait' = wait_T \wedge wait_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right) \quad \{\text{Predicate calculus}\}$$

$$= \left(\begin{array}{l} \mathbf{CI013}(\neg\ ok \wedge wait_T) \\ \vee \\ \mathbf{TR3} \circ \mathbf{CI013} \left(\begin{array}{l} ok' \wedge wait' = wait_T \wedge wait_T \wedge ok \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness conditions **CI013** and **TR3**}

$$= \left(\begin{array}{l} \mathbf{CI013}(\neg\ ok \wedge wait_T) \\ \vee \\ \left(\begin{array}{l} ok' \wedge \mathbf{CI013}(wait' = wait_T) \\ \wedge \\ \mathbf{TR3}(wait_T \wedge ok) \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Definition of **TR3** and predicate calculus}

$$
= \left(\begin{array}{l}
\mathbf{CI013}(\neg\ ok \wedge wait_T) \\
\vee \\
\left(\begin{array}{l}
ok' \wedge \mathbf{CI013}(wait' = wait_T) \\
\wedge \\
wait_T \wedge ok \\
\wedge \\
\#tr'_T = \#tr_T \wedge wait'_T \\
\wedge \\
snd \circ last(tr_T) = snd \circ last(tr'_T) \\
\wedge \\
Flat(tr'_T) = Flat(tr_T)
\end{array} \right)
\end{array} \right)
$$

{Definition of **CI3** and predicate calculus $wait'_T \Rightarrow wait'$}

$$
= \left(\begin{array}{l}
\mathbf{CI013}(\neg\ ok \wedge wait_T) \\
\vee \\
\left(\begin{array}{l}
ok' \wedge \mathbf{CI013}(wait'_T = wait_T) \\
\wedge \\
wait_T \wedge ok \\
\wedge \\
\#tr'_T = \#tr_T \wedge wait'_T \\
\wedge \\
snd \circ last(tr_T) = snd \circ last(tr'_T) \\
\wedge \\
Flat(tr'_T) = Flat(tr_T)
\end{array} \right)
\end{array} \right)
$$

{Conjunctive healthiness conditions **CI013**}

$$= \mathbf{CI013}\left(\begin{array}{l}(\neg\ ok \wedge wait_T)\\ \vee\\ \left(\begin{array}{l} ok' \wedge wait'_T = wait_T\\ \wedge\\ wait_T \wedge ok\\ \wedge\\ \#tr'_T = \#tr_T \wedge wait'_T\\ \wedge\\ snd \circ last(tr_T) = snd \circ last(tr'_T)\\ \wedge\\ Flat(tr'_T) = Flat(tr_T)\end{array}\right)\end{array}\right) \qquad \{\text{Predicate calculus}\}$$

$$= \mathbf{CI013}\left(wait_T \wedge \left(\neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait'_T = wait_T \wedge \#tr'_T = \#tr_T\\ \wedge\\ snd \circ last(tr_T) = snd \circ last(tr'_T)\\ \wedge\\ Flat(tr'_T) = Flat(tr_T)\end{array}\right)\right)\right)$$

□

Lemma L.4.1.10 ***Isabelle proof available: Section 9.5***

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013}(wait_T \wedge \mathit{II}_{rea})$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \mathit{II}_A)$$

Proof.

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013}(wait_T \wedge \mathit{II}_{rea}) \qquad \{\text{Lemma L.4.1.9}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}\left(wait_T \wedge \left(\neg\ ok \vee \left(\begin{array}{l} ok' \wedge wait'_T = wait_T \wedge \#tr'_T = \#tr_T\\ \wedge\\ snd \circ last(tr_T) = snd \circ last(tr'_T)\\ \wedge\\ Flat(tr'_T) = Flat(tr_T)\end{array}\right)\right)\right)$$

$$\{\text{Predicate calculus}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \left(\begin{array}{l} (wait_T \wedge \neg\ ok) \\ \vee \\ \left(\begin{array}{l} wait_T \wedge ok' \wedge wait'_T = wait_T \wedge \#tr'_T = \#tr_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Distributivity of **CI013**, $\mathbf{R1_T}$ and $\mathbf{R0_T}$}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \neg\ ok) \\ \vee \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \left(\begin{array}{l} wait_T \wedge ok' \wedge wait'_T = wait_T \wedge \#tr'_T = \#tr_T \\ \wedge \\ snd \circ last(tr_T) = snd \circ last(tr'_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \end{array} \right) \end{array} \right)$$

{Commutativity of healthiness conditions and Lemma L.5.1.6}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \neg\ ok) \\ \vee \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge ok' \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right)$$

{Commutativity of healthiness conditions and definition of $\mathbf{R1_T}$}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \neg\ ok \wedge \mathbf{R1_T}(true)) \\ \vee \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge ok' \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right)$$

{Distributivity of **CI013**, $\mathbf{R1_T}$ and $\mathbf{R0_T}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \left(wait_T \wedge \left(\begin{array}{l} (\neg\ ok \wedge \mathbf{R1_T}(true)) \\ \vee \\ (ok' \wedge wait'_T = wait_T \wedge tr'_T = tr_T) \end{array} \right) \right)$$

{Definition of II_A}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013}(wait_T \wedge \mathit{II}_A)$$

□

Lemma L.4.1.11

$$\mathbf{CI013} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P)$$

Proof.

$\mathbf{CI013} \circ \mathbf{R3}(P)$ {Definition of **R3**}
$= \mathbf{CI013}(\mathbb{I}_{rea} \lhd wait \rhd P)$ {Definition of conditional}
$= \mathbf{CI013}((wait \wedge \mathbb{I}_{rea}) \vee (\neg\, wait \wedge P))$ {Distributivity of **CI013**}
$= \mathbf{CI013}(wait \wedge \mathbb{I}_{rea}) \vee \mathbf{CI013}(\neg\, wait \wedge P)$
{Definition of **CI3** and commutativity}
$= \mathbf{CI013}(wait_T \wedge \mathbb{I}_{rea}) \vee \mathbf{CI013}(\neg\, wait_T \wedge P)$ {Definition of conditional}
$= \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P)$

□

Lemma L.4.1.12

$$\mathbf{TR3} \circ \mathbf{CI013} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{TR3} \circ \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P)$$

Proof.

$\mathbf{TR3} \circ \mathbf{CI013} \circ \mathbf{R3}(P)$ {Lemma L.4.1.11}
$= \mathbf{TR3}(\mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P))$
{Property of conditional and **TR3** is a conjunctive healthiness condition}
$= \mathbf{TR3} \circ \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{TR3} \circ \mathbf{CI013}(P)$ {Property of conditional}
$= \mathbf{TR3} \circ \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd (\neg\, wait_T \wedge \mathbf{TR3} \circ \mathbf{CI013}(P))$ {Lemma L.3.4.6}
$= \mathbf{TR3} \circ \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd (\neg\, wait_T \wedge \mathbf{CI013}(P))$
{Property of conditional}
$= \mathbf{TR3} \circ \mathbf{CI013}(\mathbb{I}_{rea}) \lhd wait_T \rhd \mathbf{CI013}(P)$

□

Lemma L.4.1.13

$$\begin{array}{l}\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) \\ = \\ \mathbf{R1_T}\left(\begin{array}{l}\exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait']\end{array}\right)\end{array}$$

Proof.

$$\begin{array}{ll}\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) & \{\text{Lemma L.4.1.14}\} \\ = \mathbf{R1_T}\left(\begin{array}{l}Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \left(\begin{array}{l}\exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait']\end{array}\right)\end{array}\right) & \\ & \{\text{Lemma L.2.2.22 and predicate calculus}\} \\ = \mathbf{R1_T}\left(\begin{array}{l}\exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait']\end{array}\right) & \end{array}$$

□

Lemma L.4.1.14

$$\begin{array}{l}\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P) \\ = \\ \left(\begin{array}{l}Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \left(\begin{array}{l}\exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait']\end{array}\right)\end{array}\right)\end{array}$$

Proof.

$$\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P) \qquad \{\text{Definition of } \mathbf{CI0}, \mathbf{CI1} \text{ and } \mathbf{CI3}\}$$

$$= \exists\, tr, tr', ref, ref', wait, wait' \bullet \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \\ \wedge \\ wait_T = wait \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right)$$

$$\{\text{One-point rule}\}$$

$$= \exists\, tr, tr', wait' \bullet \left(\begin{array}{l} P[snd \circ last(tr_T), snd \circ last(tr'_T)/ref, ref'][wait_T/wait] \\ \wedge \\ (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right)$$

$$\{\text{Property of sequences}\}$$

$$= \exists\, tr, tr', wait' \bullet \left(\begin{array}{l} P[snd \circ last(tr_T), snd \circ last(tr'_T)/ref, ref'][wait_T/wait] \\ \wedge \\ tr' = tr \frown Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right)$$

$$\{\text{One-point rule}\}$$

$$= \exists\, tr, wait' \bullet \left(\begin{array}{l} P \left[\begin{array}{l} snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown Flat(tr'_T) - Flat(tr_T)/tr' \end{array} \right] \\ \wedge \\ Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right) \qquad \{\text{Predicate calculus}\}$$

$$= \left(\begin{array}{l} Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \exists\, tr, wait' \bullet \left(\begin{array}{l} P \left[\begin{array}{l} snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown Flat(tr'_T) - Flat(tr_T)/tr' \end{array} \right] \\ \wedge \\ (\neg\, wait' \Rightarrow \neg\, wait'_T) \end{array} \right) \end{array} \right)$$

$\{\text{Case-analysis on } wait'\}$

$$= \left(\begin{array}{l} Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \exists\, tr \bullet \left(\begin{array}{l} \left(P \left[\begin{array}{l} snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown Flat(tr'_T) - Flat(tr_T)/tr' \\ false/wait' \end{array} \right] \wedge \neg\, wait'_T \right) \\ \vee \\ P \left[\begin{array}{l} snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown Flat(tr'_T) - Flat(tr_T)/tr' \\ true/wait' \end{array} \right] \end{array} \right) \end{array} \right)$$

$\{\text{Definition of } subs\}$

$$= \left(\begin{array}{l} Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \exists\, tr \bullet \left(\begin{array}{l} (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ subs(P)[true/wait'] \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}$$

$$= \left(\begin{array}{l} Flat(tr_T) \leq Flat(tr'_T) \\ \wedge \\ \left(\begin{array}{l} \exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait'] \end{array} \right) \end{array} \right)$$

□

4.2 Results on CI1

Lemma L.4.2.1

$$\mathbf{CI1}(P)^f_f = \mathbf{CI1}(P^f_f)$$

Proof.

$\mathbf{CI1}(P)^f_f$ {Definition of **CI1**}

$= (P \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T))^f_f$ {Substitution}

$= P^f_f \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T)$ {Definition of **CI1**}

$= \mathbf{CI1}(P^f_f)$

□

Lemma L.4.2.2

$$\mathbf{CI1}(P)^o_f = \mathbf{CI1}(P^o_f)$$

Proof.

$\mathbf{CI1}(P)^o_f$ {Definition of **CI1**}

$= (P \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T))^o_f$ {Substitution}

$= P^o_f \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T)$ {Definition of **CI1**}

$= \mathbf{CI1}(P^o_f)$

□

Lemma L.4.2.3

$$\mathbf{R2_T} \circ \mathbf{CI1}(P) = \mathbf{CI1} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R2_T} \circ \mathbf{CI1}(P)$ {Definition of **CI1**}

$= \mathbf{R2_T}(P \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T))$

{Distributivity of $\mathbf{R2_T}$ (Lemma L.2.3.18)}

$= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(ref = snd \circ last(tr_T)) \wedge \mathbf{R2_T}(ref' = snd \circ last(tr'_T))$

{Definition of $\mathbf{R2_T}$ and substitution}

$$= \left(\begin{array}{l} \mathbf{R2_T}(P) \wedge ref = snd \circ last(\langle(\langle\rangle, snd \circ last(tr_T))) \\ \wedge \\ ref' = snd \circ last(dif_T(tr'_T, tr_T)) \end{array} \right)$$

{Definition of *last* and *snd*}

$= \mathbf{R2_T}(P) \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(dif_T(tr'_T, tr_T))$

{Lemma L.2.3.10}

$= \mathbf{R2_T}(P) \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T)$ {Definition of **CI1**}

$= \mathbf{CI1} \circ \mathbf{R2_T}(P)$

□

Lemma L.4.2.4

$$\mathbf{CI1}(ref' = ref) = \mathbf{CI1}(snd \circ last(tr_T) = snd \circ last(tr'_T))$$

Proof.

$\mathbf{CI1}(ref' = ref)$ {Definition of **CI1**}

$= ref' = ref \wedge ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T)$

{Transitivity of equality}

$= ref = snd \circ last(tr_T) \wedge ref' = snd \circ last(tr'_T) \wedge snd \circ last(tr_T) = snd \circ last(tr'_T)$

{Definition of **CI1**}

$= \mathbf{CI1}(snd \circ last(tr_T) = snd \circ last(tr'_T))$

□

4.3 Results on CI3

Lemma L.4.3.1

$$\mathbf{CI3}(P)^f_f = P^f_f \wedge wait_T = false \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)$$

Proof.

$\mathbf{CI3}(P)^f_f$ {Definition of **CI3**}

$= (P \wedge wait_T = wait \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T))^f_f$ {Substitution}

$= P^f_f \wedge wait_T = false \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)$

□

Lemma L.4.3.2

$$\mathbf{CI3}(P)^o_f = P^o_f \wedge \neg\, wait_T \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)$$

Proof.

$\mathbf{CI3}(P)^o_f$ {Definition of **CI3**}

$= (P \wedge wait_T = wait \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T))^o_f$ {Substitution}

$= P^o_f \wedge wait_T = false \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)$ {Predicate calculus}

$= P^o_f \wedge \neg\, wait_T \wedge (\neg\, wait' \Rightarrow \neg\, wait'_T)$

□

Lemma L.4.3.3

$$(\mathbf{CI3} \circ \mathbf{R3_T}(P))^o_f = \mathbf{CI3}(P)^o_f$$

Proof.

$(\mathbf{CI3} \circ \mathbf{R3_T}(P))^o_f$ {Lemma L.4.3.2}
$= \mathbf{R3_T}(P)^o_f \wedge \neg\ wait_T \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T)$ {Lemma L.2.4.2}
$= ((\mathit{II}_A)^o_f \lhd wait_T \rhd P^o_f) \wedge \neg\ wait_T \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T)$
{Definition of conditional and predicate calculus}
$= P^o_f \wedge \neg\ wait_T \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T)$ {Lemma L.4.3.2}
$= \mathbf{CI3}(P)^o_f$

□

Lemma L.4.3.4

$$\mathbf{CI3}(P)^o = \mathbf{CI3}(P^o)$$

Proof.

$\mathbf{CI3}(P)^o$ {Definition of **CI3**}
$= (P \wedge wait_T = wait \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T))^o$ {Substitution}
$= (P^o \wedge wait_T = wait \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T))$ {Definition of **CI3**}
$= \mathbf{CI3}(P^o)$

□

Lemma L.4.3.5

$$\mathbf{R3_T}(\mathbf{CI3}(P)^o_f) = \mathbf{R3_T} \circ \mathbf{CI3}(P^o)$$

Proof.

$\mathbf{R3_T}(\mathbf{CI3}(P)^o_f)$ {Lemma L.2.4.1}
$= \mathbf{R3_T} \circ \mathbf{CI3}(P^o)$

□

Lemma L.4.3.6

$$\mathbf{R2_T} \circ \mathbf{CI3}(P) = \mathbf{CI3} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R2_T} \circ \mathbf{CI3}(P)$ {Definition of **CI3**}

$= \mathbf{R2_T}(P \wedge wait_T = wait \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T))$

{Distributivity of $\mathbf{R2_T}$ (Lemma L.2.3.18) and Lemma L.2.3.12}

$= \mathbf{R2_T}(P) \wedge wait_T = wait \wedge (\neg\ wait' \Rightarrow \neg\ wait'_T))$ {Definition of **CI3**}

$= \mathbf{CI3} \circ \mathbf{R2_T}(P)$

□

4.4 Results on CI2

Lemma L.4.4.1

$$\mathbf{CI2}(P \wedge Q) = \mathbf{CI2}(P) \wedge \mathbf{CI2}(Q)$$

Proof.

$\mathbf{CI2}(P) \wedge \mathbf{CI2}(Q)$ {Definition of **CI2**}

$$
= \left(\begin{array}{l} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \\ \wedge \\ \left(\begin{array}{l} Q \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ Q_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} P \wedge Q \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ Q_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} P \wedge Q \\ \wedge \\ \left(\begin{array}{l} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \vee \\ (\neg\ wait' \wedge \neg\ Q_f^f \wedge ok \wedge ok' \wedge tr' = tr) \end{array} \right) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} P \wedge Q \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge (\neg\ P_f^f \vee \neg\ Q_f^f) \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$= \left(\begin{array}{l} P \wedge Q \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (P_f^f \wedge Q_f^f) \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Property of substitution}

$$= \left(\begin{array}{l} P \wedge Q \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (P \wedge Q)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Definition of **CI2**}

$$= \mathbf{CI2}(P) \wedge \mathbf{CI2}(Q)$$

□

Lemma L.4.4.2

$$\mathbf{CI2} \circ \mathbf{R1_C}(P) = \mathbf{CI2} \circ \mathbf{R1_C}(P)$$

Proof.

$\mathbf{R1_C} \circ \mathbf{CI2}(P)$ {Definition of **CI2**}

$$= \mathbf{R1_C} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{R1_C}$}

$$= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \mathbf{R1_C} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$
= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \mathbf{R1_C} \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Conjunctive healthiness condition $\mathbf{R1_C}$}

$$
= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \mathbf{R1_C} \left(\begin{array}{l} (wait' \vee \mathbf{R1_C}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Lemma L.4.9.1}

$$
= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \mathbf{R1_C} \left(\begin{array}{l} (wait' \vee \mathbf{R1_C}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Predicate calculus}

$$
= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \mathbf{R1_C} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R1_C}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Conjunctive healthiness condition $\mathbf{R1_C}$}

$$
= \left(\begin{array}{l} \mathbf{R1_C}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R1_C}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Definition of **CI2**}

$$
= \mathbf{CI2} \circ \mathbf{R1_C}(P)
$$

□

Lemma L.4.4.3

$$\mathbf{CI2} \circ \mathbf{R2_C}(P) = \mathbf{R2_C} \circ \mathbf{CI2}(P)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_C} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{R2_C} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Distributivity of } \mathbf{R2_C}\} \\
= \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C} \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Distributivity of } \mathbf{R2_C} \text{ and } tr_C \text{ and } tr'_C \text{ not free}\} \\
= \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2_C}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Lemma L.4.10.1}\} \\
= \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2_C}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\}
\end{array}$$

$$
\begin{array}{ll}
 & \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R2_C}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \wedge \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI2} \circ \mathbf{R2_C}(P) &
\end{array}
$$

□

Lemma L.4.4.4

$$\mathbf{TR3} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{TR3}(P)$$

Proof.

$$
\begin{array}{ll}
\mathbf{TR3} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{TR3}\left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Predicate calculus}\} \\
= \mathbf{TR3}\left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (wait' \vee P^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \\
 & \{\text{Conjunctive and idempotent healthiness condition } \mathbf{TR3}\} \\
= \left(\begin{array}{l} \mathbf{TR3}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{TR3}(P^f_f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Lemma L.3.4.2}\}
\end{array}
$$

$$
= \left(\begin{array}{l} \mathbf{TR3}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{TR3}(P)^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} \mathbf{TR3}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{TR3}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Definition of } \mathbf{CI2}\}
$$

$$
= \mathbf{CI2} \circ \mathbf{TR3}(P)
$$

□

4.4.1 CI2 and R3

Lemma L.4.4.5

$$
\mathbf{CI2} \circ \mathbf{R3}(P) = \mathbf{R3}(P) \wedge \mathbf{CI2_m}(P)
$$

Proof.

$$
\mathbf{CI2} \circ \mathbf{R3}(P) \qquad \{\text{Definition of } \mathbf{CI2}\}
$$

$$
= \left(\begin{array}{l} \mathbf{R3}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R3}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

$$
\{\text{Definition of } \mathbf{R3} \text{ and substitution}\}
$$

$$
= \left(\begin{array}{l} \mathbf{R3}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Definition of } \mathbf{CI2_m}\}
$$

$= \mathbf{R3}(P) \wedge \mathbf{CI2_m}(P)$

□

Lemma L.4.4.6

$$\mathbf{R0_T} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{R0_T}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{CI2}(P)$ {Definition of **CI2**}

$$= \mathbf{R0_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \quad \{\text{Predicate calculus}\}$$

$$= \mathbf{R0_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Conjunctive idempotent healthiness condition ($\mathbf{R0_T}$)}

$$= \left(\begin{array}{l} \mathbf{R0_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R0_T}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \quad \{\text{Lemma L.2.1.1}\}$$

$$= \left(\begin{array}{l} \mathbf{R0_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R0_T}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \quad \{\text{Predicate calculus}\}$$

$$
\begin{array}{ll}
 & \left(\begin{array}{l} \mathbf{R0_T}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R0_T}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI2} \circ \mathbf{R0_T}(P) &
\end{array}
$$

□

Lemma L.4.4.7

$$\mathbf{R1_T} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{R1_T}(P)$$

Proof.

$$
\begin{array}{ll}
\mathbf{R1_T} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{R1_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Predicate calculus}\} \\
= \mathbf{R1_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (wait' \vee P^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \\
 & \{\text{Conjunctive idempotent healthiness condition } (\mathbf{R1_T})\} \\
= \left(\begin{array}{l} \mathbf{R1_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R1_T}(P^f_f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right) & \{\text{Lemma L.2.2.24}\}
\end{array}
$$

$$
\begin{array}{ll}
= \left(\begin{array}{l} \mathbf{R1_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R1_T}(P)^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \mathbf{R1_T}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R1_T}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI2} \circ \mathbf{R1_T}(P) &
\end{array}
$$

□

Lemma L.4.4.8

$$\mathbf{R2_T} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{R2_T}(P)$$

Proof.

$$
\begin{array}{ll}
\mathbf{R2_T} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{R2_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \mathbf{R2_T} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (wait' \vee P^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Distributivity of } \mathbf{R2_T}\}
\end{array}
$$

$$= \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ \left(\begin{array}{l} \mathbf{R2_T}(wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \mathbf{R2_T}(\#tr'_T = \#tr_T) \end{array} \right) \end{array} \right)$$

{Distributivity of $\mathbf{R2_T}$ and Lemma L.2.3.12}

$$= \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2_T}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ (\#tr'_T = \#tr_T) \end{array} \right) \end{array} \right) \quad \{\text{Lemma L.2.3.1}\}$$

$$= \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ \left(\begin{array}{l} (wait' \vee \mathbf{R2_T}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \quad \{\text{Predicate calculus}\}$$

$$= \left(\begin{array}{l} \mathbf{R2_T}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{R2_T}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \quad \{\text{Definition of } \mathbf{CI2}\}$$

$$= \mathbf{CI2} \circ \mathbf{R2_T}(P)$$

□

4.4.1.1 CI2 and CI.0

Lemma L.4.4.9

$$\mathbf{CI0} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{CI0}(P)$$

Proof.

$\mathbf{CI0} \circ \mathbf{CI2}(P)$ {Definition of **CI2**}

$$= \mathbf{CI0}\left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Conjunctive healthiness condition}

$$= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \mathbf{CI0}\left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \mathbf{CI0}\left(\begin{array}{l} \neg\ (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \mathbf{CI0}\left(\begin{array}{l} wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Conjunctive healthiness condition}

$$= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \mathbf{CI0}\left(\begin{array}{l} wait' \vee \mathbf{CI0}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Lemma L.4.1.2}

$$= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \mathbf{CI0}\left(\begin{array}{l} wait' \vee \mathbf{CI0}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr \\ \vee \\ \#tr'_T = \#tr_T \end{array}\right) \end{array}\right)$$

{Conjunctive healthiness condition}

$$
\begin{array}{ll}
= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \left(\begin{array}{l} wait' \vee \mathbf{CI0}(P)^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \mathbf{CI0}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI0}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI2} \circ \mathbf{CI0}(P) &
\end{array}
$$

□

Lemma L.4.4.10

$$\mathbf{CI1} \circ \mathbf{CI2}(P) = \mathbf{CI2} \circ \mathbf{CI1}(P)$$

Proof.

$$
\begin{array}{ll}
\mathbf{CI1} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI1} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition}\} \\
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \mathbf{CI1} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\}
\end{array}
$$

$$
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \mathbf{CI1} \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Conjunctive and idempotent **CI1**}

$$
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \mathbf{CI1} \left(\begin{array}{l} (wait' \vee \mathbf{CI1}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Lemma L.4.2.1}\}
$$

$$
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \mathbf{CI1} \left(\begin{array}{l} (wait' \vee \mathbf{CI1}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Predicate calculus}

$$
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \mathbf{CI1} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI1}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Conjunctive and idempotent **CI1**}

$$
= \left(\begin{array}{l} \mathbf{CI1}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI1}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) \qquad \{\text{Definition of } \mathbf{CI2}\}
$$

$$
= \mathbf{CI2} \circ \mathbf{CI1}(P)
$$

□

Lemma L.4.4.11

$$\mathbf{CI2} \circ \mathbf{CI4_m}(P) = \mathbf{CI4_m} \circ \mathbf{CI2}(P)$$

Proof.

$$\begin{array}{ll}
\mathbf{CI4_m} \circ \mathbf{CI2}(P) & \{\text{Definition of } \mathbf{CI2}\} \\
= \mathbf{CI4_m} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{CI4_m}\} \\
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{CI4_m}\} \\
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee \mathbf{CI4_m}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \\
& \{\text{Lemma L.4.8.1}\}
\end{array}$$

$$
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee \mathbf{CI4_m}(P)^f_f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Predicate calculus}

$$
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI4_m}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Conjunctive healthiness condition $\mathbf{CI4_m}$}

$$
= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI4_m}(P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)
$$

{Definition of **CI2**}

$$
= \mathbf{CI2} \circ \mathbf{CI4_m}(P)
$$

□

Lemma L.4.4.12

$$
\mathbf{CI2} \circ \mathbf{R3}(P) = \mathbf{R3}(P \wedge \mathbf{CI2_m}(P))
$$

Proof.

$$
\begin{array}{ll}
\mathbf{CI2} \circ \mathbf{R3}(P) & \{\text{Lemma L.4.4.5}\} \\
= \mathbf{R3}(P) \wedge \mathbf{CI2_m}(P) & \{\text{Definition of } \mathbf{R3}\} \\
= (\mathbb{I}_{rea} \lhd wait \rhd P) \wedge \mathbf{CI2_m}(P) & \{\text{Property of conditional}\} \\
= (\mathbb{I}_{rea} \wedge \mathbf{CI2_m}(P)) \lhd wait \rhd (P \wedge \mathbf{CI2_m}(P)) & \\
\qquad \{\text{Property of conditional and Lemma L.4.5.1}\} & \\
= \mathbb{I}_{rea} \lhd wait \rhd (P \wedge \mathbf{CI2_m}(P)) & \{\text{Definition of } \mathbf{R3}\} \\
= \mathbf{R3}(P \wedge \mathbf{CI2_m}(P)) &
\end{array}
$$

□

4.5 Results on CI2m

Lemma L.4.5.1

$$\mathbb{\Pi}_{rea} \wedge wait \wedge \mathbf{CI2_m}(P) = \mathbb{\Pi}_{rea} \wedge wait$$

Proof.

$$\begin{array}{ll}
\mathbb{\Pi}_{rea} \wedge \mathbf{CI2_m}(P) & \{\text{Definition of } \mathbb{\Pi}_{rea}\} \\
= (\mathbf{R1}(\neg\, ok) \vee (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref)) \wedge wait \wedge \mathbf{CI2_m}(P) & \\
& \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} (\mathbf{R1}(\neg\, ok) \wedge wait \wedge \mathbf{CI2_m}(P)) \\ \vee \\ (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref \wedge wait \wedge \mathbf{CI2_m}(P)) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{R1}\} \\
= \left(\begin{array}{l} \mathbf{R1}(\neg\, ok \wedge wait \wedge \mathbf{CI2_m}(P)) \\ \vee \\ (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref \wedge wait \wedge \mathbf{CI2_m}(P)) \end{array} \right) & \\
& \{\text{Lemma L.4.5.4}\} \\
= \left(\begin{array}{l} \mathbf{R1}(\neg\, ok \wedge wait) \\ \vee \\ (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref \wedge wait \wedge \mathbf{CI2_m}(P)) \end{array} \right) & \\
& \{\text{Lemma L.4.5.2}\} \\
= \left(\begin{array}{l} \mathbf{R1}(\neg\, ok \wedge wait) \\ \vee \\ (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref \wedge wait) \end{array} \right) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{R1} \text{ and predicate calculus}\} \\
= \left(\left(\begin{array}{l} \mathbf{R1}(\neg\, ok) \\ \vee \\ (ok' \wedge wait' = wait \wedge tr' = tr \wedge ref' = ref) \end{array} \right) \wedge wait \right) & \{\text{Definition of } \mathbb{\Pi}_{rea}\} \\
= \mathbb{\Pi}_{rea} \wedge wait &
\end{array}$$

□

Lemma L.4.5.2

$$(wait' = wait \wedge wait \wedge \mathbf{CI2_m}(P)) = wait' = wait \wedge wait$$

Proof.

$$
\begin{array}{ll}
(wait' = wait \wedge wait \wedge \mathbf{CI2_m}(P)) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \left(\begin{array}{l} wait' = wait \wedge wait \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} wait' = wait \wedge wait \wedge wait' \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\} \\
= wait' = wait \wedge wait \wedge wait' & \{\text{Predicate calculus}\} \\
= wait' = wait \wedge wait &
\end{array}
$$

□

Lemma L.4.5.3

$$\mathbf{CI2_m}(P \wedge Q) = \mathbf{CI2_m}(P) \wedge \mathbf{CI2_m}(Q)$$

Proof.

$$
\begin{array}{ll}
\mathbf{CI2_m}(P \wedge Q) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (P \wedge Q)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Substitution}\}
\end{array}
$$

$$
\begin{array}{ll}
= \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (P_f^f \wedge Q_f^f) \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge (\neg\ P_f^f \vee \neg\ Q_f^f) \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \vee \\ (\neg\ wait' \wedge \neg\ Q_f^f \wedge ok \wedge ok' \wedge tr' = tr) \end{array} \right) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \\ \vee \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ Q_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \mathbf{CI2_m}(P) \wedge \mathbf{CI2_m}(Q) &
\end{array}
$$

□

Lemma L.4.5.4

$$\neg\ ok \wedge \mathbf{CI2_m}(P) = \neg\ ok$$

Proof.

$$
\begin{array}{ll}
\neg\ ok \wedge \mathbf{CI2_m}(P) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \left(\begin{array}{l} \neg\ ok \wedge (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \neg\ ok &
\end{array}
$$

□

Lemma L.4.5.5

$$\mathbf{CI2_m}(P \vdash Q) = \mathbf{CI2_m}(\neg\ P)$$

Proof.

$$\begin{array}{ll}
\mathbf{CI2_m}(P \vdash Q) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (P \vdash Q)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \\
& \{\text{Substitution and definition of design}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \neg\ (ok \wedge P^f_f) \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge P^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Predicate calculus}\} \\
= \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ (\neg\ P)^f_f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \mathbf{CI2_m}(\neg\ P) &
\end{array}$$

□

Lemma L.4.5.6 *Provided ok' and $wait$ are not free in P,*

$$\begin{array}{l}
\mathbf{CI2_m}(P) \wedge (\neg\ P \vdash Q) \\
= \\
\left(\begin{array}{l} \neg\ P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)
\end{array}$$

Proof.

$$
\begin{array}{ll}
\mathbf{CI2_m}(P) \wedge (\neg\, P \vdash Q) & \{\text{Definition of } \mathbf{CI2_m}\} \\
= \left(\left(\begin{array}{l} (\neg\, wait' \wedge \neg\, P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \right) & \{\text{Definition of design}\} \\
\quad \wedge \\
\quad (\neg\, P \vdash Q) & \\
= \left(\left(\begin{array}{l} (\neg\, wait' \wedge \neg\, P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \right) & \{\text{Predicate calculus}\} \\
\quad \wedge \\
\quad (ok \wedge \neg\, P) \Rightarrow (Q \wedge ok') & \\
= \left(\left(\begin{array}{l} (ok \wedge \neg\, P_f^f) \\ \Rightarrow \\ ((\neg\, wait' \wedge ok' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \right) & \\
\quad \wedge \\
\quad (ok \wedge \neg\, P) \Rightarrow (Q \wedge ok') & \\
& \{\text{Assumption: } ok' \text{ and } wait \text{ are not free in } P\} \\
= \left(\left(\begin{array}{l} (ok \wedge \neg\, P) \\ \Rightarrow \\ ((\neg\, wait' \wedge ok' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \end{array} \right) \right) & \{\text{Predicate calculus}\} \\
\quad \wedge \\
\quad (ok \wedge \neg\, P) \Rightarrow (Q \wedge ok') & \\
= \left(\begin{array}{l} (ok \wedge \neg\, P) \\ \Rightarrow \\ \left(\begin{array}{l} ((\neg\, wait' \wedge ok' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ (Q \wedge ok') \end{array} \right) \end{array} \right) & \{\text{Predicate calculus}\}
\end{array}
$$

$$= \left(\begin{array}{l} (ok \wedge \neg\ P) \\ \Rightarrow \\ \left(\begin{array}{l} ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Q \wedge ok' \end{array} \right) \end{array} \right) \qquad \{\text{Definition of design}\}$$

$$= \left(\begin{array}{l} \neg\ P \\ \vdash \\ ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr'_T = \#tr_T) \wedge Q \end{array} \right)$$

□

4.6 Results on CI0132

Lemma L.4.6.1

$$\mathbf{CI0132} \circ \mathbf{R0T}(P) = \mathbf{R0T} \circ \mathbf{CI0132}(P)$$

Proof.

$\mathbf{CI0132} \circ \mathbf{R0_T}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{R0_T}(P)$ {Lemma L.4.4.6}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R0_T} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI3** and $\mathbf{R0_T}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R0_T} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI1** and $\mathbf{R0_T}$}

$= \mathbf{CI0} \circ \mathbf{R0_T} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI0** and $\mathbf{R0_T}$}

$= \mathbf{R0_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI0132**}

$= \mathbf{R0_T} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.2

$$\mathbf{CI0132} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P) = \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0132}(P)$$

Proof.

$= \mathbf{CI0132} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Lemma L.4.4.7}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R1_T} \circ \mathbf{CI2} \circ \mathbf{R2_T}(P)$ {Lemma L.4.4.8}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI3** and $\mathbf{R1_T}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R1_T} \circ \mathbf{CI3} \circ \mathbf{R2_T} \circ \mathbf{CI2}(P)$ {Lemma L.4.3.6}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI1** and $\mathbf{R1_T}$}

$= \mathbf{CI0} \circ \mathbf{R1_T} \circ \mathbf{CI1} \circ \mathbf{R2_T} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Lemma L.4.2.3}

$= \mathbf{CI0} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI0** and $\mathbf{R1_T}$}

$= \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{R2_T} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Lemma L.4.1.6}

$= \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI0132**}

$= \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.3

$$\mathbf{CI0132} \circ \mathbf{TR3}(P) = \mathbf{TR3} \circ \mathbf{CI0132}(P)$$

Proof.

CI0132 $\circ$ **TR3**(P) {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{TR3}(P)$ {Lemma L.4.4.4}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{TR3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI3** and **TR3**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{TR3} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI1** and **TR3**}

$= \mathbf{CI0} \circ \mathbf{TR3} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$

{Conjunctive healthiness conditions **CI0** and **TR3**}

$= \mathbf{TR3} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI0132**}

$= \mathbf{TR3} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.4

$$\mathbf{CI0132} \circ \mathbf{R1_C}(P) = \mathbf{R1_C} \circ \mathbf{CI0132}(P)$$

Proof.

$\mathbf{CI0132} \circ \mathbf{R1_C}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{R1_C}(P)$ {Lemma L.4.4.2}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R1_C} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI3** and $\mathbf{R1_C}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R1_C} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI1** and $\mathbf{R1_C}$}

$= \mathbf{CI0} \circ \mathbf{R1_C} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI0** and $\mathbf{R1_C}$}

$= \mathbf{R1_C} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI0132**}

$= \mathbf{R1_C} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.5

$$\mathbf{CI0132} \circ \mathbf{R2_C}(P) = \mathbf{R2_C} \circ \mathbf{CI0132}(P)$$

Proof.

$\mathbf{CI0132} \circ \mathbf{R2_C}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{R2_C}(P)$ {Lemma L.4.4.3}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2_C} \circ \mathbf{CI2}(P)$ {Lemma L.4.10.5}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R2_C} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Lemma L.4.10.4}

$= \mathbf{CI0} \circ \mathbf{R2_C} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Lemma L.4.10.3}

$= \mathbf{R2_C} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI0132**}

$= \mathbf{R2_C} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.6

$$\mathbf{CI0132} \circ \mathbf{CI4_m}(P) = \mathbf{CI4_m} \circ \mathbf{CI0132}(P)$$

Proof.

$\mathbf{CI0132} \circ \mathbf{CI4_m}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{CI4_m}(P)$ {Lemma L.4.4.11}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI4_m} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI3** and $\mathbf{CI4_m}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI4_m} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI1** and $\mathbf{CI4_m}$}

$= \mathbf{CI0} \circ \mathbf{CI4_m} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$
{Conjunctive healthiness conditions **CI0** and $\mathbf{CI4_m}$}

$= \mathbf{CI4_m} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI.1**}

$= \mathbf{CI4_m} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.6.7

$$\mathbf{CI0132} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R3}(P \wedge \mathbf{CI2_m}(P))$$

Proof.

$\mathbf{CI0132} \circ \mathbf{R3}(P)$ {Definition of **CI0132**}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{R3}(P)$ {Lemma L.4.4.12}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R3}(P \wedge \mathbf{CI2_m}(P))$

□

Lemma L.4.6.8

$$\mathbf{CITR} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$$

Proof.

$\mathbf{CITR} \circ \mathbf{R3}(P)$ {Lemma L.4.6.9}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$
{Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$
{Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$
{Lemma L.4.7.21}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$
{Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$
{Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

□

Lemma L.4.6.9

$$\mathbf{CITR} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$$

Proof.

$\mathbf{CITR} \circ \mathbf{R3}(P)$ {Lemma L.3.6.2}

$= \mathbf{CI0132} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{R3}(P)$ {Lemma L.4.6.1}

$= \mathbf{R0_T} \circ \mathbf{CI0132} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{R3}(P)$ {Lemma L.4.6.2}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0132} \circ \mathbf{TR3} \circ \mathbf{CI4_m} \circ \mathbf{R3}(P)$ {Lemma L.4.6.3}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{CI4_m} \circ \mathbf{R3}(P)$ {Lemma L.4.8.4}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$ {Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$ {Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$ {Lemma L.4.8.6}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$ {Lemma L.4.6.10}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.4.8.7}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.2.5.1}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.2.2.12}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

□

Lemma L.4.6.10 ***Isabelle proof available: Section 9.5***

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI0132} \circ \mathbf{R3}(P)$ {Lemma L.4.6.7}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI013} \circ \mathbf{R3}(P \wedge \mathbf{CI2_m}(P))$ {Lemma L.4.1.7}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P \wedge \mathbf{CI2_m}(P))$

□

Lemma L.4.6.11

$$\mathbf{R2}(P \vdash Q) = (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q))$$

Proof.

$$\begin{array}{ll}
\mathbf{R2}(P \vdash Q) & \{\text{Definition of design}\} \\
= \mathbf{R2}((ok \wedge P) \Rightarrow (Q \wedge ok')) & \{\text{Predicate calculus}\} \\
= \mathbf{R2}((\neg\, ok \vee \neg\, P) \vee (Q \wedge ok')) & \{\text{Distributivity of } \mathbf{R2}\} \\
= \mathbf{R2}(\neg\, ok) \vee \mathbf{R2}(\neg\, P) \vee (\mathbf{R2}(Q) \wedge \mathbf{R2}(ok')) & \{\text{Definition of } \mathbf{R2}\} \\
= \neg\, ok \vee \mathbf{R2}(\neg\, P) \vee (\mathbf{R2}(Q) \wedge ok') & \{\text{Predicate calculus}\} \\
= (ok \wedge \neg\, \mathbf{R2}(\neg\, P)) \Rightarrow (\mathbf{R2}(Q) \wedge ok') & \{\text{Definition of design}\} \\
= (\neg\, \mathbf{R2}(\neg\, P) \vdash \mathbf{R2}(Q)) &
\end{array}$$

□

Lemma L.4.6.12

$$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, wait' \wedge \mathbf{R2}(R \vee S)))$$

$$= \mathbf{R1_T}(\#tr'_T = \#tr_T \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T)$$

Proof.

$$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, wait' \wedge \mathbf{R2}(R \vee S)))$$

$$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, wait' \wedge (R \vee S)))$$

{Distributivity of $\mathbf{R2}$}

$$= \mathbf{R1_T} \left(\begin{array}{l} (subsR2((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, wait' \wedge (R \vee S))[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ subsR2((tr' = tr \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, wait' \wedge (R \vee S))[true/wait'] \end{array} \right)$$

{Lemma L.4.6.20}

{Definition of $subsR2$ and substitution}

$$= \mathbf{R1_T}\left(\begin{array}{l}((Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, false \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, u \\ \vee \\ ((Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \wedge \neg\, true \wedge subsR2(R \vee S)[true/wait'])\end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R1_T}((Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T)$$

□

Lemma L.4.6.13

$$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(\#tr'_T = \#tr_T \wedge \neg\, wait' \wedge \mathbf{R2}(R \vee S)))$$

$$= \mathbf{R1_T}(\#tr'_T = \#tr_T \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T)$$

Proof.

$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(\#tr'_T = \#tr_T \wedge \neg\, wait' \wedge \mathbf{R2}(R \vee S)))$ {Distributivity of **R2**}

$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(\#tr'_T = \#tr_T \wedge \neg\, wait' \wedge (R \vee S)))$ {Lemma L.4.6.20}

$$= \mathbf{R1_T}\left(\begin{array}{l}(subsR2(\#tr'_T = \#tr_T \wedge \neg\, wait' \wedge (R \vee S))[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ subsR2(\#tr'_T = \#tr_T \wedge \neg\, wait' \wedge (R \vee S))[true/wait']\end{array}\right)$$

{Definition of $subsR2$ and substitution}

$$= \mathbf{R1_T}\left(\begin{array}{l}(\#tr'_T = \#tr_T \wedge \neg\, false \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ (\#tr'_T = \#tr_T \wedge \neg\, true \wedge subsR2(R \vee S)[true/wait'])\end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R1_T}(\#tr'_T = \#tr_T \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\, wait'_T)$$

□

Lemma L.4.6.14

$$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S)))$$

$=$

$$\mathbf{R1_T}\left(\begin{array}{l}(Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\ wait'_T)\\ \vee\\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait'])\end{array}\right)$$

Proof.

$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(tr' \neq tr \wedge \mathbf{R2}(R \vee S)))$ {Predicate calculus}

$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(\neg\ tr' = tr \wedge \mathbf{R2}(R \vee S)))$ {Healthy predicate}

$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(\neg\ \mathbf{R2}(tr' = tr) \wedge \mathbf{R2}(R \vee S)))$ {Property of **R2**}

$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(\mathbf{R2}(\neg\ (tr' = tr)) \wedge \mathbf{R2}(R \vee S)))$ {Distributivity of **R2**}

$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(tr' \neq tr \wedge (R \vee S)))$ {Lemma L.4.6.20}

$$= \mathbf{R1_T}\left(\begin{array}{l}(subsR2(tr' \neq tr \wedge (R \vee S))[false/wait'] \wedge \neg\ wait'_T)\\ \vee\\ subsR2(tr' \neq tr \wedge (R \vee S))[true/wait']\end{array}\right)$$

{Definition of $subsR2$ and substitution}

$$= \mathbf{R1_T}\left(\begin{array}{l}(Flat(tr'_T) - Flat(tr_T) \neq \langle\rangle \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\ wait'_T)\\ \vee\\ (Flat(tr'_T) - Flat(tr_T) \neq \langle\rangle \wedge subsR2(R \vee S)[true/wait'])\end{array}\right)$$

{Property of sequences}

$$= \mathbf{R1_T}\left(\begin{array}{l}(Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[false/wait'] \wedge \neg\ wait'_T)\\ \vee\\ (Flat(tr'_T) \neq Flat(tr_T) \wedge subsR2(R \vee S)[true/wait'])\end{array}\right)$$

□

Lemma L.4.6.15

$$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)))$$
$$=$$
$$\mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait'])$$

Proof.

$\mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013}(tr' = tr \wedge wait' \wedge \mathbf{R2}(R \wedge S)))$ {Distributivity of **R2**}

$$= \mathbf{R1_T}(\exists\, \alpha \bullet \mathbf{CI013} \circ \mathbf{R2}(tr' = tr \wedge wait' \wedge (R \wedge S))) \qquad \{\text{Lemma L.4.6.20}\}$$

$$= \mathbf{R1_T}\left(\begin{array}{l} (subsR2(tr' = tr \wedge wait' \wedge (R \wedge S))[false/wait'] \wedge \neg\, wait'_T) \\ \vee \\ subsR2(tr' = tr \wedge wait' \wedge (R \wedge S))[true/wait'] \end{array}\right)$$

$$\{\text{Definition of } subsR2 \text{ and substitution}\}$$

$$= \mathbf{R1_T}\left(\begin{array}{l} \left(\begin{array}{l} Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge false \wedge subsR2(R \wedge S)[false/wait'] \\ \wedge \\ \neg\, wait'_T \end{array}\right) \\ \vee \\ (Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge true \wedge subsR2(R \wedge S)[true/wait']) \end{array}\right)$$

$$\{\text{Predicate calculus}\}$$

$$= \mathbf{R1_T}(Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge subsR2(R \wedge S)[true/wait'])$$

$$\{\text{Property of sequences}\}$$

$$= \mathbf{R1_T}(Flat(tr'_T) = Flat(tr_T) \wedge subsR2(R \wedge S)[true/wait'])$$

□

Lemma L.4.6.16

$$\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P \vdash Q))$$

$$=$$

$$\mathbf{R1_T}\left(\begin{array}{l} \neg\, (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\, P)) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q) \end{array}\right)$$

Proof.

$$\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P \vdash Q)) \qquad \{\text{Definition of design}\}$$

$$= \mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}((ok \wedge P) \Rightarrow (Q \wedge ok')))$$

$$\{\text{Predicate calculus}\}$$

$$= \mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\, ok \vee \neg\, P \vee (Q \wedge ok')))$$

$$\{\text{Conjuntive healthiness conditions } \mathbf{CI0}, \mathbf{CI1}, \mathbf{CI3}\}$$

$$= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \left(\begin{array}{l} \mathbf{CI013}(\neg\ ok) \\ \vee \\ \mathbf{CI013}(\neg\ P) \\ \vee \\ \mathbf{CI013}(Q \wedge ok') \end{array}\right) \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ ok) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q \wedge ok') \end{array}\right)$$

{Conjuntive healthiness conditions **CI0**, **CI1**, **CI3**}

$$= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ ok) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet (\mathbf{CI013}(Q) \wedge ok') \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ ok) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ ((\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q)) \wedge ok') \end{array}\right)$$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R1_T}\left(\begin{array}{l} \mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ ok)) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ ((\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q)) \wedge ok') \end{array}\right)$$

{**??**}

$$= \mathbf{R1_T}\left(\begin{array}{l} \mathbf{R1_T}(\neg\ ok) \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ ((\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q)) \wedge ok') \end{array}\right)$$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R1_T}\left(\begin{array}{l} \neg\ ok \\ \vee \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P) \\ \vee \\ ((\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q)) \wedge ok') \end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R1_T}\left(\begin{array}{l} \left(\begin{array}{l} ok \\ \wedge \\ \neg\ (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P)) \end{array}\right) \\ \Rightarrow \\ ((\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q)) \wedge ok') \end{array}\right)$$

{Definition of design}

$$= \mathbf{R1_T}\left(\begin{array}{l} \neg\ (\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(\neg\ P)) \\ \vdash \\ \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(Q) \end{array}\right)$$

□

Lemma L.4.6.17 *Provided tr_C and tr'_C are not free in P,*

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$$
$$=$$
$$\mathbf{R_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$$

Proof.

$$\exists\, tr_C, tr'_C, tr, tr', ref, ref', wait, wait' \bullet \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$$

{Predicate calculus}

$= \exists\, tr, tr', ref, ref', wait, wait' \bullet \exists\, tr_C, tr'_C \bullet \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$
{Lemmas L.2.1.3, L.2.2.26 and L.2.3.4}

$= \exists\, \alpha \bullet \mathbf{R012_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P))$ {Lemma L.2.2.12}

$= \exists\, \alpha \bullet \mathbf{R012_T} \circ \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P))$ {Lemma L.4.7.10}

$= \exists\, \alpha \bullet \mathbf{R012_T} \circ \mathbf{R1_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$ {Lemma L.2.2.12}

$= \exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T}(P)$
{Lemmas L.2.1.3, L.2.2.26 and L.2.3.4}

$= \mathbf{R012_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R3_T}(P))$ {Lemma L.2.2.12}

$= \mathbf{R012_T} \circ \mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R3_T}(P))$
{Lemma L.4.6.19}

$= \mathbf{R012_T} \circ \mathbf{R1_T} \circ \mathbf{R3_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$
{Lemma L.2.2.12}

$= \mathbf{R012_T} \circ \mathbf{R3_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$ {Definition of $\mathbf{R_T}$}

$= \mathbf{R_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$

□

Lemma L.4.6.18 *Provided tr, tr', ref, ref', $wait$ and $wait'$ are not free in P,*

$$\mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) = \mathbf{R1_T}(P)$$

Proof.

$\mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$ {Lemma L.4.1.13}

$$= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr \bullet (subs(P)[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(P)[true/wait'] \end{array}\right)$$
{Assumption: tr, tr', ref, ref', $wait$ and $wait'$ are not free in P}

$= \mathbf{R1_T}((P \wedge \neg\ wait'_T) \vee P)$ {Predicate calculus: absorption law}

$= \mathbf{R1_T}(P)$

□

Lemma L.4.6.19

$$\mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R3_T}(P))$$
$$=$$
$$\mathbf{R1_T} \circ \mathbf{R3_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P))$$

Proof.

$$\begin{array}{ll}
\mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R3_T}(P)) & \{\text{Definition of } \mathbf{R3_T}\} \\
= \mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(\mathit{II}_A \lhd wait_T \rhd P)) & \\
& \{\text{Distributivity of } \mathbf{CI013}\} \\
= \mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet (\mathbf{CI013}(\mathit{II}_A) \lhd wait_T \rhd \mathbf{CI013}(P))) & \\
& \{\text{Predicate calculus}\} \\
= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(\mathit{II}_A) \\ \lhd wait_T \rhd \\ \exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P) \end{array}\right) & \\
& \{\text{Property of conditional and distributivity of } \mathbf{R1_T}\} \\
= \left(\begin{array}{l} \mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(\mathit{II}_A)) \\ \lhd wait_T \rhd \\ \mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) \end{array}\right) & \\
& \{\text{Definition of } \mathit{II}_A \text{ and Lemma L.4.6.18}\} \\
= \left(\begin{array}{l} \mathbf{R1_T}(\mathit{II}_A) \\ \lhd wait_T \rhd \\ \mathbf{R1_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) \end{array}\right) & \\
& \{\text{Property of conditional and distributivity of } \mathbf{R1_T}\} \\
= \mathbf{R1_T}\left(\begin{array}{l} \mathit{II}_A \\ \lhd wait_T \rhd \\ (\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) \end{array}\right) & \{\text{Definition of } \mathbf{R3_T}\} \\
= \mathbf{R1_T} \circ \mathbf{R3_T}(\exists\, tr, tr, ref, ref', wait, wait' \bullet \mathbf{CI013}(P)) &
\end{array}$$

□

Lemma L.4.6.20

$$\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R2}(P))$$
$$=$$
$$\mathbf{R1_T}((subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \vee subsR2(P)[true/wait'])$$

Proof.

$$\begin{array}{ll}
\mathbf{R1_T}(\exists\, tr, tr', ref, ref', wait, wait' \bullet \mathbf{CI013} \circ \mathbf{R2}(P)) & \{\text{Lemma L.4.1.13}\} \\
= \mathbf{R1_T}\left(\begin{array}{l} \exists\, tr \bullet (subs(\mathbf{R2}(P))[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ \exists\, tr \bullet subs(\mathbf{R2}(P))[true/wait'] \end{array}\right) & \{\text{Predicate calculus}\} \\
= \mathbf{R1_T}\left(\begin{array}{l} ((\exists\, tr \bullet subs(\mathbf{R2}(P)))[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (\exists\, tr \bullet subs(\mathbf{R2}(P)))[true/wait'] \end{array}\right) & \{\text{Lemma L.5.1.3}\} \\
= \mathbf{R1_T}\left(\begin{array}{l} ((\exists\, tr \bullet subsR2(P))[false/wait'] \wedge \neg\ wait'_T) \\ \vee \\ (\exists\, tr \bullet subsR2(P))[true/wait'] \end{array}\right) & \{\text{Lemma L.5.1.2}\} \\
= \mathbf{R1_T}((subsR2(P)[false/wait'] \wedge \neg\ wait'_T) \vee subsR2(P)[true/wait']) &
\end{array}$$

□

Full substitution lemmas that were previously missing or just related to ok' being *false*. Here I make them more general.

Lemma L.4.6.21

$$\mathbf{R2}(P)^o_f = \mathbf{R2}(P^o_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2}(P)^o_f & \{\text{Definition of } \mathbf{R2}\} \\
= P[\langle\rangle, tr' - tr/tr, tr']^o_f & \{\text{Substitution}\} \\
= P^o_f[\langle\rangle, tr' - tr/tr, tr'] & \{\text{Definition of } \mathbf{R2}\} \\
= \mathbf{R2}(P^o_f) &
\end{array}$$

□

Lemma L.4.6.22

$$CI2.0_m \wedge \neg\ wait' = \neg\ wait' \wedge (tr' = tr \Rightarrow \#tr_T = \#tr'_T)$$

Proof.

$$\begin{array}{ll} CI2.0_m \wedge \neg\ wait' & \{\text{Definition of } CI2.0_m\} \\ = ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait' & \{\text{Predicate calculus}\} \\ = (tr' = tr \Rightarrow \#tr_T = \#tr'_T) \wedge \neg\ wait' & \end{array}$$

□

Lemma L.4.6.23

$$CI2.0_m \wedge tr' \neq tr = tr' \neq tr$$

Proof.

$$\begin{array}{ll} CI2.0_m \wedge tr' \neq tr & \{\text{Definition of } CI2.0_m\} \\ = ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr_T = \#tr'_T) \wedge tr' \neq tr & \{\text{Predicate calculus}\} \\ = tr' \neq tr & \end{array}$$

□

Lemma L.4.6.24

$$CI2.0_m \wedge wait' = wait'$$

Proof.

$$\begin{array}{ll} CI2.0_m \wedge wait' & \{\text{Definition of } CI2.0_m\} \\ = ((\neg\ wait' \wedge tr' = tr) \Rightarrow \#tr_T = \#tr'_T) \wedge wait' & \{\text{Predicate calculus}\} \\ = wait' & \end{array}$$

□

Lemma L.4.6.25

$$\mathbf{S}(P \vdash Q) = \mathbf{S}(\neg\, \mathbf{ST_m}(\neg\, P) \vdash \mathbf{ST_m}(Q))$$

Proof.

$$\begin{array}{ll}
\mathbf{S}(P \vdash Q) & \{\text{Definition of design}\} \\
= \mathbf{S}((ok \wedge P) \Rightarrow (Q \wedge ok')) & \{\text{Predicate calculus}\} \\
= \mathbf{S}(\neg\, ok \vee \neg\, P \vee (Q \wedge ok')) & \{\mathbf{S}\text{-idempotent and Lemma L.5.0.5}\} \\
= \mathbf{S}(\neg\, ok \vee \mathbf{S}(\neg\, P) \vee \mathbf{S}(Q \wedge ok')) & \{\text{Lemma L.5.0.4}\} \\
= \mathbf{S}(\neg\, ok \vee \mathbf{S} \circ \mathbf{ST_m}(\neg\, P) \vee \mathbf{S} \circ \mathbf{ST_m}(Q \wedge ok')) & \\
& \{\mathbf{S}\text{-idempotent and Lemma L.5.0.5}\} \\
= \mathbf{S}(\neg\, ok \vee \mathbf{ST_m}(\neg\, P) \vee \mathbf{ST_m}(Q \wedge ok')) & \{\text{Lemma L.5.0.8}\} \\
= \mathbf{S}(\neg\, ok \vee \mathbf{ST_m}(\neg\, P) \vee (\mathbf{ST_m}(Q) \wedge ok')) & \{\text{Predicate calculus}\} \\
= \mathbf{S}((ok \wedge \neg\, \mathbf{ST_m}(\neg\, P)) \Rightarrow (\mathbf{ST_m}(Q) \wedge ok')) & \{\text{Definition of design}\} \\
= \mathbf{S}(\neg\, \mathbf{ST_m}(\neg\, P) \vdash \mathbf{ST_m}(Q)) &
\end{array}$$

□

Lemma L.4.6.26

$$\mathbf{R2} \circ \mathbf{R012_T}(P) = \mathbf{R012_T} \circ \mathbf{R2}(P)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2} \circ \mathbf{R012_T}(P) & \{\text{Definition of } \mathbf{R012_T}\} \\
= \mathbf{R2} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{R0_T} \text{ and } tr \text{ and } tr' \text{ not free}\} \\
= \mathbf{R0_T} \circ \mathbf{R2} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P) & \\
& \{\text{Conjunctive healthiness condition } \mathbf{R1_T} \text{ and } tr \text{ and } tr' \text{ not free}\} \\
= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2} \circ \mathbf{R2_T}(P) & \{\text{Definition of } \mathbf{R2} \text{ and } \mathbf{R2_T}\} \\
= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{R2}(P) & \{\text{Definition of } \mathbf{R012_T}\} \\
= \mathbf{R012_T} \circ \mathbf{R2}(P) &
\end{array}$$

□

Lemma L.4.6.27

$$\mathbf{CI013} \circ \mathbf{R012_T}(P) = \mathbf{R012_T} \circ \mathbf{CI013}(P)$$

Proof.

$\mathbf{CI013} \circ \mathbf{R012_T}(P)$ {Definition of **CI013** and $\mathbf{R012_T}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI3** and $\mathbf{R0_T}$}

$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R0_T} \circ \mathbf{CI3} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI1** and $\mathbf{R0_T}$}

$= \mathbf{CI0} \circ \mathbf{R0_T} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI0** and $\mathbf{R0_T}$}

$= \mathbf{R0_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI3** and $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R1_T} \circ \mathbf{CI3} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI1** and $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{CI0} \circ \mathbf{R1_T} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2_T}(P)$
{Conjunctive healthiness condition **CI0** and $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2_T}(P)$ {Lemma L.4.3.6}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R2_T} \circ \mathbf{CI3}(P)$ {Lemma L.4.2.3}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI0} \circ \mathbf{R2_T} \circ \mathbf{CI1} \circ \mathbf{CI3}(P)$ {Lemma L.4.1.6}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3}(P)$ {Definition of **CI013** and $\mathbf{R012_T}$}

$= \mathbf{R012_T} \circ \mathbf{CI013}(P)$

□

4.7 Results on CIB

Lemma L.4.7.1

$$\mathbf{CIB}(P)^o_f = \mathbf{CIB}(P^o_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{CIB}(P)^o_f & \{\text{Definition of } \mathbf{CIB}\} \\
= (\mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P))^o_w & \{\text{Lemma L.4.9.2}\} \\
= \mathbf{R1_C}((\mathbf{R2_C} \circ \mathbf{CI4}(P))^o_w) & \{\text{Lemma L.4.10.2}\} \\
= \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI4}(P)^o_w) & \{\text{Lemma L.4.7.22}\} \\
= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P^o_w) & \{\text{Definition of } \mathbf{CIB}\} \\
= \mathbf{CIB}(P^o_w) &
\end{array}$$

□

Lemma L.4.7.2 *Provided tr_C and tr'_C are not free in P,*

$$\mathbf{CIB}(P) = P \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})$$

Proof.

$$\begin{array}{ll}
\mathbf{CIB}(P) & \{\text{Definition of } \mathbf{CIB}\} \\
= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P) & \{\text{Predicate calculus}\} \\
= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P \wedge \mathit{true}) & \{\text{Conjunctive healthiness condition } \mathbf{CI4}\} \\
= \mathbf{R1_C} \circ \mathbf{R2_C}(P \wedge \mathbf{CI4}(\mathit{true})) & \{\text{Distributivity of } \mathbf{R2_C}\} \\
= \mathbf{R1_C}(\mathbf{R2_C}(P) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})) & \{\text{Definition of } \mathbf{R2_C} \text{ and assumption}\} \\
= \mathbf{R1_C}(P \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})) & \{\text{Conjunctive healthiness condition } \mathbf{R1_C}\} \\
= P \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true}) &
\end{array}$$

□

Lemma L.4.7.3

$$\mathbf{CIB}(P \vee Q) = \mathbf{CIB}(P) \vee \mathbf{CIB}(Q)$$

Proof.

$$\begin{aligned}
&\mathbf{CIB}(P \vee Q) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P \vee Q) && \{\text{Conjunctive healthiness condition } \mathbf{CI4}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI4}(P) \vee \mathbf{CI4}(Q)) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI4}(P) \vee \mathbf{R2_C} \circ \mathbf{CI4}(Q)) && \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R1_C}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P) \vee \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(Q) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{CIB}(P) \vee \mathbf{CIB}(Q)
\end{aligned}$$

□

Lemma L.4.7.4

$$\mathbf{CIB}(P \wedge Q) = \mathbf{CIB}(P) \wedge \mathbf{CIB}(Q)$$

Proof.

$$\begin{aligned}
&\mathbf{CIB}(P \wedge Q) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P \wedge Q) && \{\text{Conjunctive healthiness condition } \mathbf{CI4}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI4}(P) \wedge \mathbf{CI4}(Q)) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI4}(P) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(Q)) && \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R1_C}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P) \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(Q) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{CIB}(P) \wedge \mathbf{CIB}(Q)
\end{aligned}$$

□

Lemma L.4.7.5 *Provided tr_C and tr'_C are not free in Q,*

$$\mathbf{CIB}(P \wedge Q) = \mathbf{CIB}(P) \wedge Q$$

Proof.

$$
\begin{aligned}
&\mathbf{CIB}(P \wedge Q) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P \wedge Q) && \{\text{Conjunctive healthiness condition } \mathbf{CI4}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI4}(P) \wedge Q) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI4}(P) \wedge \mathbf{R2_C}(Q)) && \{\text{Assumption: } tr_C \text{ and } tr'_C \text{ not free in } Q\}\\
&= \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI4}(P) \wedge Q) && \{\text{Conjunctive healthiness condition } \mathbf{R1_C}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P) \wedge Q && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{CIB}(P) \wedge Q
\end{aligned}
$$

□

Lemma L.4.7.6

$$\mathbf{CIB}(P) = \mathbf{CI4} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{CIB}(P) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P) && \{\text{Definition of } \mathbf{CI4}\}\\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \left(\begin{array}{l} P \wedge \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) - \mathit{fst} \circ \mathit{last}(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge \mathit{fst} \circ \mathit{last}(tr_T) \leq \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) \end{array} \right)\\
& && \{\text{Definition of } \mathbf{R2_C} \text{ and substitution}\}\\
&= \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \wedge \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) - \mathit{fst} \circ \mathit{last}(tr_T) = (tr'_C - tr_C) - \langle\rangle \\ \wedge \\ \langle\rangle \leq tr'_C - tr_C \wedge \mathit{fst} \circ \mathit{last}(tr_T) \leq \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) \end{array} \right)\\
& && \{\text{Property of sequences}\}\\
&= \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \wedge \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) - \mathit{fst} \circ \mathit{last}(tr_T) = tr'_C - tr_C \\ \wedge \\ \mathit{fst} \circ \mathit{last}(tr_T) \leq \mathit{fst} \circ \mathit{head}(tr'_T - \mathit{front}(tr_T)) \end{array} \right)\\
& && \{\text{Definition of } \mathbf{R1_C}\}
\end{aligned}
$$

$$= \left(\begin{array}{l} \mathbf{R2_C}(P) \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

{Definition of **CI4**}

$$= \mathbf{CI4} \circ \mathbf{R2_C}(P)$$

□

Lemma L.4.7.7

$$\mathbf{CI.0} \circ \mathbf{CIB}(P) = \mathbf{CIB} \circ \mathbf{CI.0}(P)$$

Proof.

$\mathbf{CI.0} \circ \mathbf{CIB}(P)$ {Lemma L.4.7.6}

$= \mathbf{CI.0} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P)$ {Definition of **CI.0**}

$= \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P)$

{Conjunctive healthiness conditions **CI4** and **CI0**}

$= \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI4} \circ \mathbf{CI0} \circ \mathbf{R2_C}(P)$

{Conjunctive healthiness conditions **CI4** and **CI1**}

$= \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI4} \circ \mathbf{CI1} \circ \mathbf{CI0} \circ \mathbf{R2_C}(P)$

{Conjunctive healthiness conditions **CI4** and **CI3**}

$= \mathbf{CI2.0} \circ \mathbf{CI4} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \circ \mathbf{R2_C}(P)$

{Conjunctive healthiness conditions **CI4** and **CI2.0**}

$= \mathbf{CI4} \circ \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0} \circ \mathbf{R2_C}(P)$

{Definition of **CI0** and Lemmas L.2.3.12 and L.2.3.18}

$= \mathbf{CI4} \circ \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{R2_C} \circ \mathbf{CI0}(P)$

{Definition of **CI1** and Lemmas L.2.3.12 and L.2.3.18}

$= \mathbf{CI4} \circ \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{R2_C} \circ \mathbf{CI1} \circ \mathbf{CI0}(P)$

{Definition of **CI3** and Lemmas L.2.3.12 and L.2.3.18}

$= \mathbf{CI4} \circ \mathbf{CI2.0} \circ \mathbf{R2_C} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0}(P)$

{Definition of **CI2.0** and Lemmas L.2.3.12 and L.2.3.18}

$= \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0}(P)$ {Lemma L.4.7.6}

$= \mathbf{CIB} \circ \mathbf{CI2.0} \circ \mathbf{CI3} \circ \mathbf{CI1} \circ \mathbf{CI0}(P)$ {Definition of **CI.0**}

$= \mathbf{CIB} \circ \mathbf{CI.0}(P)$

□

Lemma L.4.7.8

$$\mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P))$$
$$=$$
$$\mathbf{R1_T}(P[\langle\rangle, fst \circ head \circ dif_T(tr'_A, tr_A)/tr_C, tr'_C])$$

Proof.

$\mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P))$ {Lemma L.4.7.9}

$$= \mathbf{R1_T}\left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P\left[\begin{array}{l} \langle\rangle/tr_C \\ fst \circ head \circ dif_T(tr'_A, tr_A)/tr'_C \end{array}\right] \end{array}\right)$$ {Definition of $\mathbf{R1_T}$}

$= \mathbf{R1_T}(P[\langle\rangle, fst \circ head \circ dif_T(tr'_A, tr_A)/tr_C, tr'_C])$

□

Lemma L.4.7.9

$$\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)$$
$$=$$
$$\left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P\left[\begin{array}{l} \langle\rangle/tr_C \\ fst \circ head \circ dif_T(tr'_A, tr_A)/tr'_C \end{array}\right] \end{array}\right)$$

Proof.

$\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)$ {Lemma L.4.7.6}

$= \exists\, tr_C, tr'_C \bullet \mathbf{CI4} \circ \mathbf{R2_C}(P)$ {Definition of **CI4**}

$$= \exists\, tr_C, tr'_C \bullet \left(\begin{array}{l} fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ \mathbf{R2_C}(P) \end{array} \right)$$

{Property of sequences}

$$= \exists\, tr_C, tr'_C \bullet \left(\begin{array}{l} tr'_C = tr_C \frown (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T)) \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ \mathbf{R2_C}(P) \end{array} \right)$$

{Definition of $\mathbf{R2_C}$}

$$= \exists\, tr_C, tr'_C \bullet \left(\begin{array}{l} tr'_C = tr_C \frown (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T)) \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P[\langle\rangle, tr'_C - tr_C / tr_C, tr'_C] \end{array} \right)$$

{One-point rule}

$$= \exists\, tr_C \bullet \left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P \left[\begin{array}{l} \langle\rangle / tr_C \\ (tr_C \frown (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T))) - tr_C / tr'_C \end{array} \right] \end{array} \right)$$

{Property of sequences}

$$= \exists\, tr_C \bullet \left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P \left[\begin{array}{l} \langle\rangle / tr_C \\ fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) / tr'_C \end{array} \right] \end{array} \right)$$

{Predicate calculus}

$$= \left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P \left[\begin{array}{l} \langle\rangle / tr_C \\ fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) / tr'_C \end{array} \right] \end{array} \right)$$

{Lemma L.2.2.2}

$$= \left(\begin{array}{l} fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ P \left[\begin{array}{l} \langle\rangle / tr_C \\ fst \circ head \circ dif_T(tr'_A, tr_A) / tr'_C \end{array} \right] \end{array} \right)$$

□

Lemma L.4.7.10 *Provided tr_C and tr'_C are not free in P,*

$$\mathbf{R1_T}(\exists\, tr'_C, tr'_C \bullet \mathbf{CIB}(P)) = \mathbf{R1_T}(P)$$

Proof.

$\mathbf{R1_T}(\exists\, tr'_C, tr_C \bullet \mathbf{CIB}(P))$ {Lemma L.4.7.6}

$= \mathbf{R1_T}(\exists\, tr'_C, tr_C \bullet \mathbf{CI4} \circ \mathbf{R2_C}(P))$
{Assumption: tr'_C and tr_C not free in P and property of substitution}

$= \mathbf{R1_T}(\exists\, tr'_C, tr_C \bullet \mathbf{CI4}(P))$ {Definition of **CI4**}

$$= \mathbf{R1_T} \left(\exists\, tr'_C, tr_C \bullet \left(\begin{array}{l} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right) \right)$$
{Property of sequences}

$$= \mathbf{R1_T} \left(\exists\, tr'_C, tr_C \bullet \left(\begin{array}{l} P \wedge tr_C \frown (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T)) = tr'_C \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right) \right)$$
{Assumption: tr'_C not free in P and one-point rule}

$= \mathbf{R1_T}(\exists\, tr_C \bullet P \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)))$
{Assumption: tr_C not free in P and predicate calculus}

$= \mathbf{R1_T}(P \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)))$ {Definition of $\mathbf{R1_T}$}

$= \mathbf{R1_T}(P)$

□

Lemma L.4.7.11

$$\mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB} \circ \mathbf{R3_T}(P))$$

$=$

$\mathbf{R1_T} \circ \mathbf{R3_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P))$

Proof.

$\mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB} \circ \mathbf{R3_T}(P))$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(\mathit{II}_A \lhd \mathit{wait}_T \rhd P))$ {Lemma L.4.7.12}

$= \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet (\mathbf{CIB}(\mathit{II}_A) \lhd \mathit{wait}_T \rhd \mathbf{CIB}(P)))$ {Lemma L.5.1.4}

$= \mathbf{R1_T}((\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(\mathit{II}_A)) \lhd \mathit{wait}_T \rhd (\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)))$

{Distributivity of $\mathbf{R1_T}$ (Lemma L.2.2.8)}

$$= \left(\begin{array}{l} \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(\mathit{II}_A)) \\ \lhd \mathit{wait}_T \rhd \\ \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)) \end{array} \right) \quad \{\text{Lemma L.4.7.10}\}$$

$$= \left(\begin{array}{l} \mathbf{R1_T}(\mathit{II}_A) \\ \lhd \mathit{wait}_T \rhd \\ \mathbf{R1_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)) \end{array} \right) \quad \{\text{Distributivity of } \mathbf{R1_T} \text{ (Lemma L.2.2.8)}\}$$

$= \mathbf{R1_T}(\mathit{II}_A \lhd \mathit{wait}_T \rhd (\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P)))$ {Definition of $\mathbf{R3_T}$}

$= \mathbf{R1_T} \circ \mathbf{R3_T}(\exists\, tr_C, tr'_C \bullet \mathbf{CIB}(P))$

□

Lemma L.4.7.12 *Provided tr_C and tr'_C are not free in Q,*

$\mathbf{CIB}(P \lhd Q \rhd R) = \mathbf{CIB}(P) \lhd Q \rhd \mathbf{CIB}(R)$

Proof.

$\mathbf{CIB}(P \lhd Q \rhd R)$ {Lemma L.4.7.6}

$= \mathbf{CI4} \circ \mathbf{R2_C}(P \lhd Q \rhd R)$

{Assumption: tr_C and tr'_C not free in Q and distributivity of $\mathbf{R2_C}$}

$= \mathbf{CI4}(\mathbf{R2_C}(P) \lhd Q \rhd \mathbf{R2_C}(R))$

{Definition of conditional and distributivity of $\mathbf{CI4}$}

$= \mathbf{CI4} \circ \mathbf{R2_C}(P) \lhd Q \rhd \mathbf{CI4} \circ \mathbf{R2_C}(R)$ {Lemma L.4.7.6}

$= \mathbf{CIB}(P) \lhd Q \rhd \mathbf{CIB}(R)$

□

Lemma L.4.7.13

$$\mathbf{CIB} \circ \mathbf{R0_T}(P) = \mathbf{R0_T} \circ \mathbf{CIB}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{CIB} \circ \mathbf{R0_T}(P) && \{\text{Lemma L.4.7.6}\}\\
&= \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{R0_T}(P) && \{\text{Definition of } \mathbf{R0_T}\}\\
&= \mathbf{CI4} \circ \mathbf{R2_C}(P \wedge \#tr_T > 0) && \{\text{Distributivity of } \mathbf{R2_C} \text{ and definition}\}\\
&= \mathbf{CI4}(\mathbf{R2_C}(P) \wedge \#tr_T > 0) && \{\text{Definition of } \mathbf{CI4} \text{ and predicate calculus}\}\\
&= \mathbf{CI4} \circ \mathbf{R2_C}(P) \wedge \#tr_T > 0 && \{\text{Definition of } \mathbf{R0_T}\}\\
&= \mathbf{R0_T} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P) && \{\text{Lemma L.4.7.6}\}\\
&= \mathbf{R0_T} \circ \mathbf{CIB}(P)
\end{aligned}$$

□

Lemma L.4.7.14

$$\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB}(P) = \mathbf{CIB} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB}(P) && \{\text{Lemma L.4.7.6}\}\\
&= \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P) && \{\text{Definition of } \mathbf{CI4}\}\\
&= \mathbf{R1_T} \circ \mathbf{R2_T} \left(\begin{array}{l} fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \\ \wedge \\ \mathbf{R2_C}(P) \end{array} \right) && \{\text{Distributivity of } \mathbf{R2_T} \text{ (Lemma L.2.3.18)}\}
\end{aligned}$$

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ \mathbf{R2_T}(tr_C \leq tr'_C) \\ \wedge \\ \mathbf{R2_T}(fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T))) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Lemma L.2.3.12}

$$= \mathbf{R1_T} \left(\begin{array}{l} \mathbf{R2_T}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ \mathbf{R2_T}(fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T))) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Definition of $\mathbf{R2_T}$ and substitution}

$$= \mathbf{R1_T} \left(\begin{array}{l} \left(\left(\begin{array}{l} fst \circ head(dif_T(tr'_T, tr_T) - front(\langle(\langle\rangle, snd \circ last(tr_T))\rangle)) \\ - \\ fst \circ last(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) \end{array} \right) = tr'_C - tr_C \right) \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ \left(\begin{array}{l} fst \circ last(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) \\ \leq \\ fst \circ head \left(\begin{array}{l} dif_T(tr'_T, tr_T) \\ - \\ front(\langle(\langle\rangle, snd \circ last(tr_T))\rangle) \end{array} \right) \end{array} \right) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Property of sequences and definition of *last*}

$$= \mathbf{R1_T} \left(\begin{array}{l} fst \circ head(dif_T(tr'_T, tr_T) - \langle\rangle) - fst(\langle\rangle, snd \circ last(tr_T)) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ \left(\begin{array}{l} fst(\langle\rangle, snd \circ last(tr_T)) \\ \leq \\ fst \circ head(dif_T(tr'_T, tr_T) - \langle\rangle) \end{array} \right) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Property of sequences and definition of *fst*}

$$= \mathbf{R1_T} \left(\begin{array}{l} fst \circ head \circ dif_T(tr'_T, tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ \langle\rangle \leq fst \circ head \circ dif_T(tr'_T, tr_T) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Property of sequences and Lemma L.2.2.2}

$$= \mathbf{R1_T} \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Definition of $\mathbf{R1_T}$ and predicate calculus}

$$= \mathbf{R1_T} \left(\begin{array}{l} fst \circ head(tr'_A - front(tr_A)) - fst \circ last(tr_A) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ fst \circ last(tr_A) \leq fst \circ head(tr'_A - front(tr_A)) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_C}(P) \end{array} \right)$$

{Definition of **CI4**}

$= \mathbf{R1_T} \circ \mathbf{CI4} \circ \mathbf{R2_T} \circ \mathbf{R2_C}(P)$ {Lemma L.4.10.9}

$= \mathbf{R1_T} \circ \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{R2_T}(P)$

{Conjunctive healthiness conditons $\mathbf{R1_T}$ and $\mathbf{CI4}$}

$= \mathbf{CI4} \circ \mathbf{R1_T} \circ \mathbf{R2_C} \circ \mathbf{R2_T}(P)$ {Definition of $\mathbf{R1_T}$ and Lemma L.2.3.12}

$= \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Lemma L.4.7.6}

$= \mathbf{CIB} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$

□

Lemma L.4.7.15

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P) = \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CIB}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$ {Definition of $\mathbf{CI4_m}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} P \\ \wedge \\ fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \left(\begin{array}{l} P \\ \wedge \\ (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

{Definition of $\mathbf{CIB}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CIB}(P)$

□

Lemma L.4.7.16

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$$
$$=$$
$$\mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$
$\{$Conjunctive healthiness conditions **TR3** and $\mathbf{R1_C}\}$
$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{TR3} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$ $\{$Lemma L.4.10.13$\}$
$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P)$
$\{$Conjunctive healthiness conditions **TR3** and $\mathbf{CI4_m}\}$
$= \mathbf{R0_T} \circ \mathbf{R1_T}\mathbf{CI4_m} \circ \mathbf{TR3}(P)$
$\{$Conjunctive healthiness conditions **TR3** and $\mathbf{CI4_m}\}$
$= \mathbf{R0_T} \circ \mathbf{CI4_m} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P)$
$\{$Conjunctive healthiness conditions $\mathbf{R0_T}$ and $\mathbf{CI4_m}\}$
$= \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P)$

□

Lemma L.4.7.17

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$$
$$=$$
$$\mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$ $\{$Conjunctive healthiness conditions $\mathbf{R1_T}$ and $\mathbf{CI4_m}\}$
$= \mathbf{R0_T} \circ \mathbf{CI4_m} \circ \mathbf{R1_T}(P)$ $\{$Conjunctive healthiness conditions $\mathbf{R0_T}$ and $\mathbf{CI4_m}\}$
$= \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T}(P)$

□

Lemma L.4.7.18

$$\mathbf{R2} \circ \mathbf{CIB}(P) = \mathbf{CIB} \circ \mathbf{R2}(P)$$

Proof.

$\mathbf{R2} \circ \mathbf{CIB}(P)$ {Definition of **CIB**}
$= \mathbf{R2} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P)$ {Definition of **R2** and $\mathbf{R1_C}$}
$= \mathbf{R1_C} \circ \mathbf{R2} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P)$ {Definition of **R2** and $\mathbf{R2_C}$}
$= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R2} \circ \mathbf{CI4}(P)$ {Definition of **R2** and **CI4**}
$= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4} \circ \mathbf{R2}(P)$ {Definition of **CIB**}
$= \mathbf{CIB} \circ \mathbf{R2}(P)$

□

Lemma L.4.7.19

$$\mathbf{CI013} \circ \mathbf{CIB}(P) = \mathbf{CIB} \circ \mathbf{CI013}(P)$$

Proof.

$\mathbf{CI013} \circ \mathbf{CIB}(P)$ {Lemma L.4.7.6}
$= \mathbf{CI013} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P)$ {Definition of **CI013**}
$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI4} \circ \mathbf{R2_C}(P)$
{Conjunctive healthiness conditions **CI4** and **CI3**}
$= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI4} \circ \mathbf{CI3} \circ \mathbf{R2_C}(P)$
{Conjunctive healthiness conditions **CI4** and **CI1**}
$= \mathbf{CI0} \circ \mathbf{CI4} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2_C}(P)$
{Conjunctive healthiness conditions **CI4** and **CI0**}
$= \mathbf{CI4} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2_C}(P)$ {Lemma L.4.10.5}
$= \mathbf{CI4} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{R2_C} \circ \mathbf{CI3}(P)$ {Lemma L.4.10.4}
$= \mathbf{CI4} \circ \mathbf{CI0} \circ \mathbf{R2_C} \circ \mathbf{CI1} \circ \mathbf{CI3}(P)$ {Lemma L.4.10.3}
$= \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3}(P)$ {Definition of **CI013**}
$= \mathbf{CI4} \circ \mathbf{R2_C} \circ \mathbf{CI013}(P)$ {Lemma L.4.7.6}

$= \mathbf{CIB} \circ \mathbf{CI013}(P)$

□

Lemma L.4.7.20

$$\mathbf{CIB} \circ \mathbf{R012_T}(P) = \mathbf{R012_T} \circ \mathbf{CIB}(P)$$

Proof.

$\mathbf{CIB} \circ \mathbf{R012_T}(P)$ {Definition of $\mathbf{R012_T}$}
$= \mathbf{CIB} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Lemma L.4.7.13}
$= \mathbf{R0_T} \circ \mathbf{CIB} \circ \mathbf{R1_T} \circ \mathbf{R2_T}(P)$ {Lemma L.4.7.14}
$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB}(P)$ {Definition of $\mathbf{R012_T}$}
$= \mathbf{R012_T} \circ \mathbf{CIB}(P)$

□

Lemma L.4.7.21

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$$
$$=$$
$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CIB}(P)$$

Proof.

$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$ {Definition of $\mathbf{CI4_m}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \left(\begin{array}{l} P \\ \wedge \\ fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \end{array} \right)$$

{Distributivity of $\mathbf{R2_C}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{R1_C}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ tr_C \leq tr'_C \end{array} \right)$$

$$\{\text{Conjunctive healthiness condition } \mathbf{R1_T}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

$$\{\text{Property of } \mathbf{R1_C} \text{ and } \mathbf{R2_C}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ \mathbf{R1_C} \circ \mathbf{R2_C}(tr_C \leq tr'_C) \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

$$\{tr_C \text{ and } tr'_C \text{ not free}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \left(\begin{array}{l} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ \mathbf{R1_C} \circ \mathbf{R2_C}(tr_C \leq tr'_C) \\ \wedge \\ \mathbf{R2_C}(fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T))) \end{array} \right)$$

$$\{\text{Conjunctive healthiness condition } \mathbf{R1_C}\}$$

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \begin{pmatrix} \mathbf{R2_C}(P) \\ \wedge \\ \mathbf{R2_C}(fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C) \\ \wedge \\ \mathbf{R2_C}(tr_C \leq tr'_C) \\ \wedge \\ \mathbf{R2_C}(fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T))) \end{pmatrix}$$

{Distributivity of $\mathbf{R2_C}$}

$$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \begin{pmatrix} P \\ \wedge \\ fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \\ \wedge \\ fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{pmatrix}$$

{Definition of **CI4**}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(P)$ {Definition of **CIB**}

$= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{CIB}(P)$

□

Lemma L.4.7.22

$$\mathbf{CI4}(P)^o_f = \mathbf{CI4}(P^o_f)$$

Proof.

$\mathbf{CI4}(P)$ {Definition of **CI4**}

$$= \begin{pmatrix} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{pmatrix}^o_f$$

{Substitution}

$$= \left(\begin{array}{l} P_f^o \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array} \right)$$

{Definition of **CI4**}

$= \mathbf{CI4}(P_f^o)$

□

4.8 Results on $\mathbf{CI4_m}$

Lemma L.4.8.1

$$\mathbf{CI4_m}(P)_f^f = \mathbf{CI4_m}(P_f^f)$$

Proof.

$\mathbf{CI4_m}(P)_f^f$ {Definition of $\mathbf{CI4_m}$}

$= (P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C)_f^f$ {Substitution}

$= P_f^f \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C$

{Definition of $\mathbf{CI4_m}$}

$= \mathbf{CI4_m}(P_f^f)$

□

Lemma L.4.8.2

$$\mathbf{CI4_m}(P)_f^f = \mathbf{CI4_m}(P_f^f)$$

Proof.

$\mathbf{CI4_m}(P)_f^f$ {Definition of $\mathbf{CI4_m}$}

$= (P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C)_f^f$ {Substitution}

$= P_f^f \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C$

{Definition of $\mathbf{CI4_m}$}

$= \mathbf{CI4_m}(P_f^f)$

□

Lemma L.4.8.3

$$\mathbf{CIB}(P)^o_f = \mathbf{CIB}(P^o_f)$$

Proof.

$$\begin{aligned}
&\mathbf{CIB}(P) && \{\text{Definition of } \mathbf{CIB}\}\\
&= \left(\begin{array}{l} P \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array}\right)^o_f \\
& && \{\text{Substitution}\}\\
&= \left(\begin{array}{l} P^o_f \wedge fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T) = tr'_C - tr_C \\ \wedge \\ tr_C \leq tr'_C \wedge fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T)) \end{array}\right) \\
& && \{\text{Definition of } \mathbf{CIB}\}\\
&= \mathbf{CI4}(P^o_f)
\end{aligned}$$

□

Lemma L.4.8.4

$$\mathbf{CI0132} \circ \mathbf{CI4_m}(P) = \mathbf{CI4_m} \circ \mathbf{CI0132}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{CI0132} \circ \mathbf{CI4_m}(P) && \{\text{Definition of } \mathbf{CI.1}\}\\
&= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2} \circ \mathbf{CI4_m}(P) && \{\text{Lemma L.4.8.5}\}\\
&= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI4_m} \circ \mathbf{CI2}(P) \\
& && \{\text{Conjunctive healthiness conditions } \mathbf{CI3} \text{ and } \mathbf{CI4_m}\}\\
&= \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI4_m} \circ \mathbf{CI3} \circ \mathbf{CI2}(P) \\
& && \{\text{Conjunctive healthiness conditions } \mathbf{CI1} \text{ and } \mathbf{CI4_m}\}\\
&= \mathbf{CI0} \circ \mathbf{CI4_m} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P) \\
& && \{\text{Conjunctive healthiness conditions } \mathbf{CI0} \text{ and } \mathbf{CI4_m}\}
\end{aligned}$$

$= \mathbf{CI4_m} \circ \mathbf{CI0} \circ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{CI2}(P)$ {Definition of **CI.1**}

$= \mathbf{CI4_m} \circ \mathbf{CI0132}(P)$

□

Lemma L.4.8.5

$$\mathbf{CI2} \circ \mathbf{CI4_m}(P) = \mathbf{CI4_m} \circ \mathbf{CI2}(P)$$

Proof.

$\mathbf{CI4_m} \circ \mathbf{CI2}(P)$ {Definition of **CI2**}

$$= \mathbf{CI4_m} \left(\begin{array}{l} P \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{CI4_m}$}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ P_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee P_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{CI4_m}$}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee \mathbf{CI4_m}(P_f^f) \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Lemma L.4.8.2}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (wait' \vee \mathbf{CI4_m}(P)_f^f \vee \neg\ ok \vee \neg\ ok' \vee tr' \neq tr) \\ \vee \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Predicate calculus}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \mathbf{CI4_m} \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI4_m}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{CI4_m}$}

$$= \left(\begin{array}{l} \mathbf{CI4_m}(P) \\ \wedge \\ \left(\begin{array}{l} (\neg\ wait' \wedge \neg\ \mathbf{CI4_m}(P)_f^f \wedge ok \wedge ok' \wedge tr' = tr) \\ \Rightarrow \\ \#tr'_T = \#tr_T \end{array} \right) \end{array} \right)$$

{Definition of **CI2**}

$$= \mathbf{CI2} \circ \mathbf{CI4_m}(P)$$

□

Lemma L.4.8.6

$$\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P)$$
$$=$$
$$\mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{TR3} \text{ and } \mathbf{R1_C}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{TR3} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P) \qquad \{\text{Lemma L.4.10.13}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{TR3} \circ \mathbf{CI4_m}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{TR3} \text{ and } \mathbf{CI4_m}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{TR3}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{R1_T} \text{ and } \mathbf{R1_C}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R1_T} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{TR3}(P) \qquad \{\text{Lemma L.4.10.12}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R1_T} \circ \mathbf{CI4_m} \circ \mathbf{TR3}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{TR3} \text{ and } \mathbf{CI4_m}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{R0_T} \text{ and } \mathbf{R1_C}\} \\
&= \mathbf{R1_C} \circ \mathbf{R0_T} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P) \qquad \{\text{Lemma L.4.10.11}\} \\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R0_T} \circ \mathbf{CI4_m} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{R0_T} \text{ and } \mathbf{CI4_m}\} \\
&= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{TR3}(P)
\end{aligned}$$

□

Lemma L.4.8.7

$$\begin{aligned}
&\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P) \\
&= \\
&\mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T}(P)
\end{aligned}$$

Proof.

$$\begin{aligned}
&\mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P) \\
&\qquad \{\text{Conjunctive healthiness conditions } \mathbf{R1_T} \text{ and } \mathbf{R1_C}\} \\
&= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R1_T} \circ \mathbf{R2_C} \circ \mathbf{CI4_m}(P) \qquad \{\text{Lemma L.4.10.12}\}
\end{aligned}$$

$= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R1_T} \circ \mathbf{CI4_m}(P)$

{Conjunctive healthiness conditions **TR3** and $\mathbf{CI4_m}$}

$= \mathbf{R0_T} \circ \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R1_T}(P)$

{Conjunctive healthiness conditions $\mathbf{R0_T}$ and $\mathbf{R1_C}$}

$= \mathbf{R1_C} \circ \mathbf{R0_T} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R1_T}(P)$ {Lemma L.4.10.11}

$= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{R0_T} \circ \mathbf{CI4_m} \circ \mathbf{R1_T}(P)$

{Conjunctive healthiness conditions $\mathbf{R0_T}$ and $\mathbf{CI4_m}$}

$= \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4_m} \circ \mathbf{R0_T} \circ \mathbf{R1_T}(P)$

□

4.9 Results on $\mathbf{R1_C}$

Lemma L.4.9.1

$$\mathbf{R1_C}(P)^f_f = \mathbf{R1_C}(P^f_f)$$

Proof.

$\mathbf{R1_C}(P)^f_f$ {Definition of $\mathbf{R1_C}$}

$= (tr_C \leq tr'_C \wedge P)^f_f$ {Substitution}

$= tr_C \leq tr'_C \wedge P^f_f$ {Definition of $\mathbf{R1_C}$}

$= \mathbf{R1_C}(P^f_f)$

□

Lemma L.4.9.2

$$\mathbf{R1_C}(P)^o_f = \mathbf{R1_C}(P^o_f)$$

Proof.

$\mathbf{R1_C}(P)^o_f$ {Definition of $\mathbf{R1_C}$}

$= (tr_C \leq tr'_C \wedge P)^o_f$ {Substitution}

$= tr_C \leq tr'_C \wedge P^o_f$ {Definition of $\mathbf{R1_C}$}
$= \mathbf{R1_C}(P^o_f)$

□

Lemma L.4.9.3

$$\mathbf{R2_T} \circ \mathbf{R1_C}(P) = \mathbf{R1_T} \circ \mathbf{R2_T}(P)$$

Proof.

$\mathbf{R2_T} \circ \mathbf{R1_C}(P)$ {Definition of $\mathbf{R1_C}$}
$= \mathbf{R2_T}(P \wedge tr_C \leq tr'_C)$ {Distributivity of $\mathbf{R2_T}$ (Lemma L.2.3.18)}
$= \mathbf{R2_T}(P) \wedge \mathbf{R2_T}(tr_C \leq tr'_C)$ {Lemma L.2.3.12}
$= \mathbf{R2_T}(P) \wedge tr_C \leq tr'_C$ {Definition of $\mathbf{R1_C}$}
$= \mathbf{R1_T} \circ \mathbf{R2_T}(P)$

□

4.10 Results on $\mathbf{R2_C}$

Lemma L.4.10.1

$$\mathbf{R2_C}(P)^f_f = \mathbf{R2_C}(P^f_f)$$

Proof.

$\mathbf{R2_C}(P)^f_f$ {Definition of $\mathbf{R2_C}$}
$= P[\langle\rangle, tr' - tr/tr, tr']^f_f$ {Substitution}
$= P^f_f[\langle\rangle, tr' - tr/tr, tr']$ {Definition of $\mathbf{R2_C}$}
$= \mathbf{R2_C}(P^f_f)$

□

Lemma L.4.10.2

$$\mathbf{R2_C}(P)^o_f = \mathbf{R2_C}(P^o_f)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_C}(P)^o_f & \{\text{Definition of } \mathbf{R2_C}\} \\
= P[\langle\rangle, tr' - tr/tr, tr']^o_f & \{\text{Substitution}\} \\
= P^o_f[\langle\rangle, tr' - tr/tr, tr'] & \{\text{Definition of } \mathbf{R2_C}\} \\
= \mathbf{R2_C}(P^o_f) &
\end{array}$$

□

Lemma L.4.10.3

$$\mathbf{R2_C} \circ \mathbf{CI0}(P) = \mathbf{CI0} \circ \mathbf{R2_C}(P)$$

Proof.

$$\begin{array}{ll}
\mathbf{R2_C} \circ \mathbf{CI0}(P) & \{\text{Definition of } \mathbf{CI0}\} \\
= \mathbf{R2_C} \left(\begin{array}{l} P \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) & \\
& \{\text{Distributivity of } \mathbf{R2_C} \text{ and } tr_C \text{ and } tr'_C \text{ not free}\} \\
= \left(\begin{array}{l} \mathbf{R2_C}(P) \wedge (tr' - tr) = Flat(tr'_T) - Flat(tr_T) \\ \wedge \\ tr \leq tr' \wedge Flat(tr_T) \leq Flat(tr'_T) \end{array} \right) & \{\text{Definition of } \mathbf{CI0}\} \\
= \mathbf{CI0} \circ \mathbf{R2_C}(P) &
\end{array}$$

□

Lemma L.4.10.4

$$\mathbf{R2_C} \circ \mathbf{CI1}(P) = \mathbf{CI1} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{CI1}(P) && \{\text{Definition of } \mathbf{CI1}\}\\
&= \mathbf{R2_C}(P \wedge \mathit{ref} = \mathit{snd} \circ \mathit{last}(\mathit{tr}_T) \wedge \mathit{ref}' = \mathit{snd} \circ \mathit{last}(\mathit{tr}'_T)) \\
& && \{\text{Distributivity of } \mathbf{R2_C} \text{ and } \mathit{tr}_C \text{ and } \mathit{tr}'_C \text{ not free}\}\\
&= \mathbf{R2_C}(P) \wedge \mathit{ref} = \mathit{snd} \circ \mathit{last}(\mathit{tr}_T) \wedge \mathit{ref}' = \mathit{snd} \circ \mathit{last}(\mathit{tr}'_T)) && \{\text{Definition of } \mathbf{CI1}\}\\
&= \mathbf{CI1} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

□

Lemma L.4.10.5

$$\mathbf{R2_C} \circ \mathbf{CI3}(P) = \mathbf{CI3} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{CI3}(P) && \{\text{Definition of } \mathbf{CI3}\}\\
&= \mathbf{R2_C}(P \wedge \mathit{wait}_T = \mathit{wait} \wedge (\neg\ \mathit{wait}' \Rightarrow \neg\ \mathit{wait}'_T)) \\
& && \{\text{Distributivity of } \mathbf{R2_C} \text{ and } \mathit{tr}_C \text{ and } \mathit{tr}'_C \text{ not free}\}\\
&= \mathbf{R2_C}(P) \wedge \mathit{wait}_T = \mathit{wait} \wedge (\neg\ \mathit{wait}' \Rightarrow \neg\ \mathit{wait}'_T)) && \{\text{Definition of } \mathbf{CI3}\}\\
&= \mathbf{CI3} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

□

Lemma L.4.10.6

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3}(P)\\
&=\\
&\mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3}(P)\\
& && \{\text{Commutatitivy of healthiness conditions and Lemma L.3.5.3}\}
\end{aligned}
$$

$$= \mathbf{R2_C}(P \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T \wedge ((ok \wedge wait_T) \Rightarrow \#tr'_T = \#tr_T))$$

{Definition of $\mathbf{R2_C}$}

$$= \mathbf{R2_C}(P) \wedge \#tr_T > 0 \wedge \#tr_T \leq \#tr'_T \wedge front(tr_T) < tr'_T \wedge ((ok \wedge wait_T) \Rightarrow \#tr'_T = \#tr_T)$$

{Commutatitivy of healthiness conditions and Lemma L.3.5.3}

$$= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{TR2} \circ \mathbf{TR3} \circ \mathbf{R2_C}(P)$$

□

Lemma L.4.10.7

$$\mathbf{R2_C} \circ \mathbf{TR4}(P) = \mathbf{TR4} \circ \mathbf{R2_C}(P)$$

Proof.

$\mathbf{R2_C} \circ \mathbf{TR4}(P)$ {Definition of $\mathbf{TR4}$}

$$= \mathbf{R2_C}\left(P\left[\begin{array}{l} \langle Flat(tr_T), snd \circ last(tr_T)\rangle / tr_T \\ \left(\begin{array}{l} \langle \left(\begin{array}{l} Flat(front(tr_T) \frown head(tr'_T - front(tr_T))), \\ snd \circ head(tr'_T - front(tr_T)) \end{array}\right) \rangle \\ \frown \\ tail(tr'_T - front(tr_T)) \end{array}\right) / tr'_T \end{array}\right]\right)$$

{Definition of $\mathbf{R2_C}$ and substitution}

$$= \mathbf{R2_C}(P)\left[\begin{array}{l} \langle Flat(tr_T), snd \circ last(tr_T)\rangle / tr_T \\ \left(\begin{array}{l} \langle \left(\begin{array}{l} Flat(front(tr_T) \frown head(tr'_T - front(tr_T))), \\ snd \circ head(tr'_T - front(tr_T)) \end{array}\right) \rangle \\ \frown \\ tail(tr'_T - front(tr_T)) \end{array}\right) / tr'_T \end{array}\right]$$

{Definition of $\mathbf{TR4}$}

$$= \mathbf{TR4} \circ \mathbf{R2_C}(P)$$

□

Lemma L.4.10.8

$$\mathbf{R2_C} \circ \mathbf{R2_T}(P) = \mathbf{R2_T} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{R2_T}(P) && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \mathbf{R2_C}\left(P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right]\right) && \{\text{Definition of } \mathbf{R2_C}\}\\
&= P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\\ \langle\rangle, tr'_C - tr_C/tr_C, tr'_C\end{array}\right] && \{\text{Property of substitution}\}\\
&= P\left[\begin{array}{l}\langle\rangle, tr'_C - tr_C/tr_C, tr'_C\\ \langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right] && \{\text{Definition of } \mathbf{R2_C}\}\\
&= \mathbf{R2_C}(P)\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right] && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \mathbf{R2_T} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

□

Lemma L.4.10.9

$$\mathbf{R2_T} \circ \mathbf{R2_C}(P) = \mathbf{R2_C} \circ \mathbf{R2_T}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_T} \circ \mathbf{R2_C}(P) && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \mathbf{R2_C}\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right] && \{\text{Definition of } \mathbf{R2_C}\}\\
&= P[\langle\rangle, tr'_C - tr_C/tr_C, tr'_C]\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right] && \{\text{Property of substitution}\}\\
&= P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T\\ dif_T(tr'_T, tr_T)/tr'_T\end{array}\right][\langle\rangle, tr'_C - tr_C/tr_C, tr'_C] && \{\text{Definition of } \mathbf{R2_T}\}\\
&= \mathbf{R2_T}(P)[\langle\rangle, tr'_C - tr_C/tr_C, tr'_C] && \{\text{Definition of } \mathbf{R2_C}\}\\
&= \mathbf{R2_C} \circ \mathbf{R2_T}(P)
\end{aligned}
$$

□

Lemma L.4.10.10

$$\mathbf{R3_T} \circ \mathbf{R2_C}(P) = \mathbf{R2_C} \circ \mathbf{R3_T}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{R3_T}(P) && \{\text{Definition of } \mathbf{R3_T}\}\\
&= \mathbf{R2_C}(\mathit{II}_A \lhd \mathit{wait}_T \rhd P) && \{\text{Property of conditional}\}\\
&= \mathbf{R2_C}(\mathit{II}_A) \lhd \mathit{wait}_T \rhd \mathbf{R2_C}(P) && \{tr_C \text{ and } tr'_C \text{ not free in } \mathit{II}_A\}\\
&= \mathit{II}_A \lhd \mathit{wait}_T \rhd \mathbf{R2_C}(P) && \{\text{Definition of } \mathbf{R3_T}\}\\
&= \mathbf{R3_T} \circ \mathbf{R2_C}(P)
\end{aligned}$$

□

Lemma L.4.10.11

$$\mathbf{R2_C} \circ \mathbf{R0_T}(P) = \mathbf{R0_T} \circ \mathbf{R2_C}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{R0_T}(P) && \{\text{Definition of } \mathbf{R0_T}\}\\
&= \mathbf{R2_C} \circ \mathbf{TR0} \circ \mathbf{TR1}(P) && \{\text{Definition of } \mathbf{TR0} \text{ and } \mathbf{TR1}\}\\
&= \mathbf{R2_C}(P \wedge \#tr > 0 \wedge \#tr_T \leq \#tr'_T) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R2_C}(P) \wedge \mathbf{R2_C}(\#tr > 0) \wedge \mathbf{R2_C}(\#tr_T \leq \#tr'_T) && \{tr_C \text{ and } tr'_C \text{ not free}\}\\
&= \mathbf{R2_C}(P) \wedge \#tr > 0 \wedge \#tr_T \leq \#tr'_T && \{\text{Definition of } \mathbf{TR0} \text{ and } \mathbf{TR1}\}\\
&= \mathbf{TR0} \circ \mathbf{TR1} \circ \mathbf{R2_C}(P) && \{\text{Definition of } \mathbf{R0_T}\}\\
&= \mathbf{R0_T} \circ \mathbf{R2_C}(P)
\end{aligned}$$

□

Lemma L.4.10.12

$$\mathbf{R2_C} \circ \mathbf{R1_T}(P) = \mathbf{R1_T} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{R1_T}(P) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R2_C}(P \wedge \mathit{Expands}_A(tr_T, tr'_T)) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R2_C}(P) \wedge \mathbf{R2_C}(\mathit{Expands}_A(tr_T, tr'_T)) && \{tr_C \text{ and } tr'_C \text{ not free}\}\\
&= \mathbf{R2_C}(P) \wedge \mathit{Expands}_A(tr_T, tr'_T) && \{\text{Definition of } \mathbf{R1_T}\}\\
&= \mathbf{R1_T} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

□

Lemma L.4.10.13

$$\mathbf{R2_C} \circ \mathbf{TR3}(P) = \mathbf{TR3} \circ \mathbf{R2_C}(P)$$

Proof.

$$
\begin{aligned}
&\mathbf{R2_C} \circ \mathbf{TR3}(P) && \{\text{Definition of } \mathbf{TR3}\}\\
&= \mathbf{R2_C}(P \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T))) && \{\text{Distributivity of } \mathbf{R2_C}\}\\
&= \mathbf{R2_C}(P) \wedge \mathbf{R2_C}((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) && \{tr_C \text{ and } tr'_C \text{ not free}\}\\
&= \mathbf{R2_C}(P) \wedge ((ok \wedge wait_T) \Rightarrow (\#tr'_T = \#tr_T \wedge wait'_T)) && \{\text{Definition of } \mathbf{TR3}\}\\
&= \mathbf{TR3} \circ \mathbf{R2_C}(P)
\end{aligned}
$$

□

Chapter 5

Results on S

Lemma L.5.0.1

$$\mathbf{S} \circ \mathbf{R012_T}(P) = \mathbf{S}(P)$$

Proof.

$\mathbf{S} \circ \mathbf{R012_T}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{R012_T}(P)$ {Lemma L.4.6.26}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P)$ {$\mathbf{R012_T}$-idempotent}

$= \mathbf{R012_T} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P)$ {Lemma L.4.7.20}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P)$ {Lemma L.4.6.27}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R012_T} \circ \mathbf{R3_T} \circ \mathbf{R012_T} \circ \mathbf{R2}(P)$
{Definition of $\mathbf{R3_T}$, distributivity of $\mathbf{R012_T}$ and idempotent}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R012_T} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Lemma L.4.6.27}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Lemma L.4.7.20}

$= \mathbf{R012_T} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {$\mathbf{R012_T}$-idempotent}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{S}(P)$

□

Lemma L.5.0.2

$$\mathbf{S} \circ \mathbf{CI013}(P) = \mathbf{S}(P)$$

Proof.

$\mathbf{S} \circ \mathbf{CI013}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{CI013}(P)$

{Conjunctive healthiness conditions **CI0**, **CI1** and **CI3** and tr and tr' not free in them}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$

{Definition of $\mathbf{R3_T}$, distributivity of **CI013** and idempotent}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{S}(P)$

□

Lemma L.5.0.3

$$\mathbf{S} \circ \mathbf{CIB}(P) = \mathbf{S}(P)$$

Proof.

$\mathbf{S} \circ \mathbf{CIB}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2} \circ \mathbf{CIB}(P)$ {Lemma L.4.7.18}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{CIB} \circ \mathbf{R2}(P)$ {Lemma L.4.7.19}

$= \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{CIB} \circ \mathbf{R3_T} \circ \mathbf{CIB} \circ \mathbf{R2}(P)$

{Definition of $\mathbf{R3_T}$ and Lemma L.4.7.3 and **CIB**-idempotent}

$= \mathbf{R012_T} \circ \mathbf{CI013} \circ \mathbf{CIB} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Lemma L.4.7.19}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P)$ {Definition of $\mathbf{S}$}

$= \mathbf{S}(P)$

□

Lemma L.5.0.4

$$\mathbf{S} \circ \mathbf{ST_m}(P) = \mathbf{S}(P)$$

Proof.

$$\begin{array}{ll}
\mathbf{S} \circ \mathbf{ST_m}(P) & \{\text{Definition of } \mathbf{ST_m}\} \\
= \mathbf{S} \circ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) & \{\text{Lemma L.5.0.1}\} \\
= \mathbf{S} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) & \{\text{Lemma L.5.0.3}\} \\
= \mathbf{S} \circ \mathbf{CI013} \circ \mathbf{R2}(P) & \{\text{Lemma L.5.0.2}\} \\
= \mathbf{S} \circ \mathbf{R2}(P) & \{\text{Definition of } \mathbf{S} \text{ and } \mathbf{R2} \text{ idempotent}\} \\
= \mathbf{S}(P) &
\end{array}$$

□

Lemma L.5.0.5

$$\mathbf{S}(P \vee Q) = \mathbf{S}(P) \vee \mathbf{S}(Q)$$

Proof.

$$\begin{array}{ll}
\mathbf{S}(P \vee Q) & \{\text{Definition of } \mathbf{S}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P \vee Q) & \{\text{Distributivity of } \mathbf{R2}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T}(\mathbf{R2}(P) \vee \mathbf{R2}(Q)) & \{\text{Distributivity of } \mathbf{R3_T}\} \\
= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \left(\begin{array}{l} \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \\
\qquad \{\text{Distributivity of conjunctive healthiness conditions } \mathbf{CI0}, \mathbf{CI1} \text{ and } \mathbf{CI3}\} & \\
= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) & \{\text{Lemma L.4.7.3}\}
\end{array}$$

$$= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Lemmas L.2.2.6 and L.2.3.19 and conjunctive healthiness condition $\mathbf{R0_T}$}

$$= \left(\begin{array}{l} \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(P) \\ \vee \\ \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R3_T} \circ \mathbf{R2}(Q) \end{array} \right) \qquad \{\text{Definition of } \mathbf{S}\}$$

$$= \mathbf{S}(P) \vee \mathbf{S}(Q)$$

□

Lemma L.5.0.6 *Provided tr_C and tr'_C are not free in Q,*

$$\mathbf{ST_m}(P \lhd Q \rhd R) = \mathbf{ST_m}(P) \lhd \mathbf{R2_T} \circ \mathbf{R2}(Q) \rhd \mathbf{ST_m}(R)$$

Proof.

$\mathbf{ST_m}(P) \lhd \mathbf{R2_T} \circ \mathbf{R2}(Q) \rhd \mathbf{ST_m}(R)$ {Definition of conditional}

$$= \left(\begin{array}{l} (\mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(P)) \\ \vee \\ (\neg\, \mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(R)) \end{array} \right) \qquad \{\text{Definition of } \mathbf{R2} \text{ and } \mathbf{R2_T}\}$$

$$= \left(\begin{array}{l} (\mathbf{R2_T} \circ \mathbf{R2}(Q) \wedge \mathbf{ST_m}(P)) \\ \vee \\ (\mathbf{R2_T} \circ \mathbf{R2}(\neg\, Q) \wedge \mathbf{ST_m}(R)) \end{array} \right) \qquad \{\text{Assumption and Lemma L.5.0.9}\}$$

$= \mathbf{ST_m}(Q \wedge P) \vee \mathbf{ST_m}(\neg\, Q \wedge R)$ {Lemma L.5.0.12}

$= \mathbf{ST_m}((Q \wedge P) \vee (\neg\, Q \wedge R))$ {Definition of conditional}

$= \mathbf{ST_m}(P \lhd Q \rhd R)$

□

Lemma L.5.0.7

$$ok \wedge \neg\, \mathbf{ST_m} \circ \mathbf{H1}(P) = ok \wedge \neg\, \mathbf{ST_m}(P)$$

Proof.

$$
\begin{aligned}
& ok \wedge \neg\, \mathbf{ST_m} \circ \mathbf{H1}(P) && \{\text{Definition of } \mathbf{H1}\} \\
&= ok \wedge \neg\, \mathbf{ST_m}(ok \Rightarrow P) && \{\text{Predicate calculus}\} \\
&= ok \wedge \neg\, \mathbf{ST_m}(\neg\, ok \vee P) && \{\text{Lemma L.5.0.12}\} \\
&= ok \wedge \neg\, (\mathbf{ST_m}(\neg\, ok) \vee \mathbf{ST_m}(P)) && \{\text{Predicate calculus}\} \\
&= ok \wedge \neg\, \mathbf{ST_m}(\neg\, ok) \wedge \neg\, \mathbf{ST_m}(P) && \{\text{Lemma L.5.0.10}\} \\
&= ok \wedge \neg\, (\neg\, ok \wedge \mathbf{ST_m}(\mathit{true})) \wedge \neg\, \mathbf{ST_m}(P) && \{\text{Predicate calculus}\} \\
&= ok \wedge (ok \vee \neg\, \mathbf{ST_m}(\mathit{true})) \wedge \neg\, \mathbf{ST_m}(P) && \{\text{Predicate calculus: absorption law}\} \\
&= ok \wedge \neg\, \mathbf{ST_m}(P)
\end{aligned}
$$

□

Lemma L.5.0.8 *Provided tr, tr', tr_C, tr'_C, tr_T and tr'_T are not free in P,*

$$\mathbf{ST_m}(P \wedge Q) = P \wedge \mathbf{ST_m}(Q)$$

Proof.

$$
\begin{aligned}
& \mathbf{ST_m}(P \wedge Q) && \{\text{Lemma L.5.0.11}\} \\
&= \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q) && \{\text{Assumption and Lemma L.5.0.10}\} \\
&= P \wedge \mathbf{ST_m}(\mathit{true}) \wedge \mathbf{ST_m}(Q) && \{\text{Lemma L.5.0.11}\} \\
&= P \wedge \mathbf{ST_m}(\mathit{true} \wedge Q) && \{\text{Predicate calculus}\} \\
&= P \wedge \mathbf{ST_m}(Q)
\end{aligned}
$$

□

Lemma L.5.0.9 *Provided tr_C, tr'_C are not free in Q,*

$$\mathbf{ST_m}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q) = \mathbf{ST_m}(P \wedge Q)$$

Proof.

$$\mathbf{ST_m}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q) \qquad \{\text{Definition of } \mathbf{ST_m}\}$$

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q)$
{Conjunctive healthiness condition $\mathbf{R0_T}$}

$= \mathbf{R0_T}(\mathbf{R12_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q))$
{Conjunctive healthiness condition $\mathbf{R1_T}$}

$= \mathbf{R0_T} \circ \mathbf{R1_T}(\mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2_T} \circ \mathbf{R2}(Q))$ {Lemma L.2.3.18}

$= \mathbf{R012_T}(\mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$ {Assumption and Lemma L.4.7.5}

$= \mathbf{R012_T} \circ \mathbf{CIB}(\mathbf{CI013} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI0**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0}(\mathbf{CI12} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI1**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \circ \mathbf{CI1}(\mathbf{CI3} \circ \mathbf{R2}(P) \wedge \mathbf{R2}(Q))$
{Conjunctive healthiness condition **CI1**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \wedge \mathbf{R2}(Q))$ {Distributivity of **R2**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \wedge Q)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{ST_m}(P \wedge Q)$

□

Lemma L.5.0.10 *Provided tr, tr', tr_C, tr'_C, tr_T and tr'_T are not free in P,*

$$\mathbf{ST_m}(P) = P \wedge \mathbf{ST_m}(true)$$

Proof.

$\mathbf{ST_m}(P)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P)$ {Assumption and definition of **R2**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(P)$ {Predicate calculus}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(P \wedge true)$ {Conjunctive healthiness condition **CI3**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01}(P \wedge \mathbf{CI3}(true))$ {Conjunctive healthiness condition **CI1**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0}(P \wedge \mathbf{CI1} \circ \mathbf{CI3}(true))$
{Conjunctive healthiness condition **CI0**}

$= \mathbf{R012_T} \circ \mathbf{CIB}(P \wedge \mathbf{CI013}(true))$ {Assumption and Lemma L.4.7.2}

$= \mathbf{R012_T}(P \wedge \mathbf{CI013}(\mathit{true}) \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true}))$

{Conjunctive healthiness condition $\mathbf{R1_C}$}

$= \mathbf{R012_T}(P \wedge \mathbf{R1_C}(\mathbf{CI013}(\mathit{true}) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})))$

{Definition of **CI0**, **CI1**, **CI3** and $\mathbf{R2_C}$}

$= \mathbf{R012_T}(P \wedge \mathbf{R1_C}(\mathbf{R2_C} \circ \mathbf{CI013}(\mathit{true}) \wedge \mathbf{R2_C} \circ \mathbf{CI4}(\mathit{true})))$

{Distributivity of $\mathbf{R2_C}$}

$= \mathbf{R012_T}(P \wedge \mathbf{R1_C} \circ \mathbf{R2_C}(\mathbf{CI013}(\mathit{true}) \wedge \mathbf{CI4}(\mathit{true})))$

{Conjunctive healthiness conditions and predicate calculus}

$= \mathbf{R012_T}(P \wedge \mathbf{R1_C} \circ \mathbf{R2_C} \circ \mathbf{CI4} \circ \mathbf{CI013}(\mathit{true}))$ {Definition of **CIB**}

$= \mathbf{R012_T}(P \wedge \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true}))$ {Distributivity of $\mathbf{R2_T}$}

$= \mathbf{R01_T}(\mathbf{R2_T}(P) \wedge \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true}))$

{Definition of $\mathbf{R2_T}$ and assumption}

$= \mathbf{R01_T}(P \wedge \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true}))$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$= \mathbf{R0_T}(P \wedge \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true}))$

{Conjunctive healthiness condition $\mathbf{R1_T}$}

$= P \wedge \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathit{true})$ {Definition of $\mathbf{ST_m}$}

$= P \wedge \mathbf{ST_m}(\mathit{true})$

□

Lemma L.5.0.11

$$\mathbf{ST_m}(P \wedge Q) = \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q)$$

Proof.

$\mathbf{ST_m}(P \wedge Q)$ {Definition of $\mathbf{ST_m}$}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \wedge Q)$ {Distributivity of **R2**}

$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \wedge \mathbf{R2}(Q))$

{Conjunctive healthiness condition **CI3**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\begin{array}{l} \mathbf{CI3} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition **CI1**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \left(\begin{array}{l} \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition **CI0**}

$$= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Lemma L.4.7.4}

$$= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Lemma L.2.3.18}

$$= \mathbf{R01_T} \left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Lemma L.2.2.4}

$$= \mathbf{R0_T} \left(\begin{array}{l} \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R2_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Conjunctive healthiness condition $\mathbf{R0_T}$}

$$= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \wedge \\ \mathbf{R0_T} \circ \mathbf{R2_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right)$$

{Definition of $\mathbf{ST_m}$}

$$= \mathbf{ST_m}(P) \wedge \mathbf{ST_m}(Q)$$

□

Lemma L.5.0.12

$$\mathbf{ST_m}(P \vee Q) = \mathbf{ST_m}(P) \vee \mathbf{ST_m}(Q)$$

Proof.

$$
\begin{aligned}
&\mathbf{ST_m}(P \lor Q) && \{\text{Definition of } \mathbf{ST_m}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P \lor Q) && \{\text{Distributivitu of } \mathbf{R2}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI013}(\mathbf{R2}(P) \lor \mathbf{R2}(Q)) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{CI3}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI01} \left(\begin{array}{l} \mathbf{CI3} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{CI1}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \circ \mathbf{CI0} \left(\begin{array}{l} \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{CI1} \circ \mathbf{CI3} \circ \mathbf{R2}(Q) \end{array} \right) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{CI0}\}\\
&= \mathbf{R012_T} \circ \mathbf{CIB} \left(\begin{array}{l} \mathbf{CI013} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) && \{\text{Lemma L.4.7.3}\}\\
&= \mathbf{R012_T} \left(\begin{array}{l} \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) && \{\text{Lemma L.2.3.19}\}\\
&= \mathbf{R01_T} \left(\begin{array}{l} \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) && \{\text{Lemma L.2.2.6}\}\\
&= \mathbf{R0_T} \left(\begin{array}{l} \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) \\
& && \{\text{Conjunctive healthiness condition } \mathbf{R0_T}\}\\
&= \left(\begin{array}{l} \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(P) \\ \lor \\ \mathbf{R0_T} \circ \mathbf{R1_T} \circ \mathbf{R2_T} \circ \mathbf{CIB} \circ \mathbf{CI013} \circ \mathbf{R2}(Q) \end{array} \right) && \{\text{Definition of } \mathbf{ST_m}\}\\
&= \mathbf{ST_m}(P) \lor \mathbf{ST_m}(Q)
\end{aligned}
$$

□

Lemma L.5.0.13

$$ok \wedge \mathbf{ST_m}(\neg\ ok) = false$$

Proof.

$$\begin{array}{ll} ok \wedge \mathbf{ST_m}(\neg\ ok) & \{\text{Lemma L.5.0.10}\} \\ = ok \wedge \neg\ ok \wedge \mathbf{ST_m}(true) & \{\text{Predicate calculus}\} \\ = false & \end{array}$$

□

5.1 Miscellaneous Results

Lemma L.5.1.1

$$\exists\, tr \bullet subs(\mathbf{R2}(P)) = subsR2(P)$$

Proof.

$$\begin{array}{ll} \exists\, tr \bullet subs(\mathbf{R2}(P)) & \{\text{Definition of } subs\} \\ = \exists\, tr \bullet \mathbf{R2}(P) \left[\begin{array}{l} snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown (Flat(tr'_T) - Flat(tr_T))/tr' \end{array}\right] & \{\text{Definition of } \mathbf{R2}\} \\ = \exists\, tr \bullet P \left[\begin{array}{l} \langle\rangle/tr \\ tr' - tr/tr' \\ snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown (Flat(tr'_T) - Flat(tr_T))/tr' \end{array}\right] & \{\text{Substitution}\} \end{array}$$

$$
\begin{array}{ll}
= \exists\, tr \bullet P \left[\begin{array}{l} \langle\rangle / tr \\ (tr \frown (Flat(tr'_T) - Flat(tr_T))) - tr / tr' \\ snd \circ last(tr_T) / ref \\ snd \circ last(tr'_T) / ref' \\ wait_T / wait \end{array}\right] & \{\text{Property of sequences}\} \\
= \exists\, tr \bullet P \left[\begin{array}{l} \langle\rangle / tr \\ Flat(tr'_T) - Flat(tr_T))) / tr' \\ snd \circ last(tr_T) / ref \\ snd \circ last(tr'_T) / ref' \\ wait_T / wait \end{array}\right] & \{\text{Predicate calculus}\} \\
= P \left[\begin{array}{l} \langle\rangle / tr \\ Flat(tr'_T) - Flat(tr_T))) / tr' \\ snd \circ last(tr_T) / ref \\ snd \circ last(tr'_T) / ref' \\ wait_T / wait \end{array}\right] & \{\text{Definition of } subsR2\} \\
= subsR2(P) &
\end{array}
$$

□

Lemma L.5.1.2

$$\exists\, tr \bullet subsR2(P) = subsR2(P)$$

Proof.

$$
\begin{array}{ll}
\exists\, tr \bullet subsR2(P) & \{\text{Defiition of } subsR2\} \\
= \exists\, tr \bullet P \left[\begin{array}{l} \langle\rangle / tr \\ Flat(tr'_T) - Flat(tr_T) / tr' \\ snd \circ last(tr_T) / ref \\ snd \circ last(tr'_T) / ref' \\ wait_T / wait \end{array}\right] & \{\text{Predicate calculus}\}
\end{array}
$$

$$= P \left[\begin{array}{l} \langle\rangle/tr \\ Flat(tr'_T) - Flat(tr_T)/tr' \\ snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \end{array} \right] \quad \{\text{Defiition of } subsR2\}$$

$$= subsR2(P)$$

□

Lemma L.5.1.3

$$subs(\mathbf{R2}(P)) = subsR2(P)$$

Proof.

$subs(\mathbf{R2}(P))$ {Definition of **R2**}

$= subs(P[\langle\rangle, tr' - tr/tr, tr'])$ {Definition of $subs$}

$$= P \left[\begin{array}{l} \langle\rangle/tr \\ tr' - tr/tr' \\ snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \\ tr \frown Flat(tr'_T) - Flat(tr_T)/tr' \end{array} \right] \quad \{\text{Substitution}\}$$

$$= P \left[\begin{array}{l} \langle\rangle/tr \\ (tr \frown Flat(tr'_T) - Flat(tr_T)) - tr/tr' \\ snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \end{array} \right] \quad \{\text{Property of sequences}\}$$

$$= P \left[\begin{array}{l} \langle\rangle/tr \\ Flat(tr'_T) - Flat(tr_T)/tr' \\ snd \circ last(tr_T)/ref \\ snd \circ last(tr'_T)/ref' \\ wait_T/wait \end{array} \right] \quad \{\text{Definition of } subsR2\}$$

$= subsR2(P)$

□

Lemma L.5.1.4

$$\exists\, v \bullet (P \lhd Q \rhd R) = (\exists\, v \bullet P) \lhd Q \rhd (\exists\, v \bullet R)$$

Proof.

$\exists\, v \bullet (P \lhd Q \rhd R)$ {Definition of conditional}

$= \exists\, v \bullet ((Q \wedge P) \vee (\neg\, Q \wedge R))$ {Predicate calculus}

$= (\exists\, v \bullet (Q \wedge P)) \vee (\exists\, v \bullet (\neg\, Q \wedge R))$

{Assumption: v is not free in Q and predicate calculus}

$= (Q \wedge \exists\, v \bullet P) \vee (\neg\, Q \wedge \exists\, v \bullet R)$ {Definition of conditional}

$= (\exists\, v \bullet P) \lhd Q \rhd (\exists\, v \bullet R)$

□

Lemma L.5.1.5

$$\mathit{II}_{rea} = \left(\begin{array}{l} (\neg\, ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge ok \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait) \end{array} \right)$$

Proof.

II_{rea} {Definition of II_{rea}}

$$= \left(\begin{array}{l} (\neg\, ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait) \end{array} \right)$$ {Predicate calculus}

$$= \left(\begin{array}{l} (\neg\, ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait \wedge (ok \vee \neg\, ok)) \end{array} \right)$$

{Predicate calculus}

$$
= \left(\begin{array}{l} (\neg\ ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait \wedge ok) \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait \wedge \neg\ ok) \end{array} \right) \qquad \{\text{Predicate calculus}\}
$$

$$
= \left(\begin{array}{l} (\neg\ ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait \wedge ok) \\ \vee \\ (ok' \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait \wedge \neg\ ok \wedge tr \leq tr') \end{array} \right)
$$

$\{\text{Predicate calculus: absorption law}\}$

$$
= \left(\begin{array}{l} (\neg\ ok \wedge tr \leq tr') \\ \vee \\ (ok' \wedge ok \wedge tr' = tr \wedge ref' = ref \wedge wait' = wait) \end{array} \right)
$$

□

Lemma L.5.1.6 *Provided* $\#s > 0$ *and* $\#t > 0$,

$$Flat(t) = Flat(s) \wedge \#t = \#s \wedge front(s) < t \wedge snd \circ last(s) = snd \circ last(t) \Leftrightarrow t = s$$

Proof. Isabelle theorem: `t__eq__s__iff__FlatA_eq_and_length_eq_and_fronts_¬lt_t_and_snd_last_e` □

Lemma L.5.1.7 *Provided* $\#s > 0$ *and* $\#s \leq \#t$,

$$front(s) < t \Leftrightarrow t = front(s) \frown (t - front(s))$$

Proof. Isabelle theorem: `front_s_lt_t__iff__t_eq_front_s_cat_t_minus_front¬_s` □

Chapter 6

Results on dif_T

Lemma L.6.0.1 *Provided* $fst \circ last(tr_T) \leq fst \circ head(tr'_T - front(tr_T))$,

$$\mathbf{TR4} \circ \mathbf{R2loc_T}(P) = \mathbf{R2_T}(P)$$

Proof.

$$\begin{aligned}
&\mathbf{TR4} \circ \mathbf{R2loc_T}(P) && \{\text{Definition of } \mathbf{TR4}\}\\
&= \mathbf{R2loc_T}(P) \begin{bmatrix} \langle (Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T)) \rangle / tr_T \\ difloc(Flat \circ front(tr_T) \frown fst \circ head(tr'_T - front(tr_T)), tr'_T, tr_T) / tr'_T \end{bmatrix} && \{\text{Definition of } \mathbf{R2loc_T}\}\\
&= \begin{pmatrix} P \begin{bmatrix} front(tr_T) \frown \langle (\langle\rangle, snd \circ last(tr_T)) \rangle / tr_T \\ front(tr_T) \frown dif_T(tr'_T, tr_T) / tr'_T \end{bmatrix} \\ \begin{bmatrix} \langle (Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T)) \rangle / tr_T \\ difloc(Flat \circ front(tr_T) \frown fst \circ head(tr'_T - front(tr_T)), tr'_T, tr_T) / tr'_T \end{bmatrix} \end{pmatrix} && \{\text{Substitution}\}
\end{aligned}$$

$$= P\left[\begin{array}{l}\left(\begin{array}{l} front(\langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle) \\ \frown \\ \langle\left(\langle\rangle, snd \circ last\left(\langle\left(\begin{array}{l} Flat \circ front(tr_T) \frown fst \circ last(tr_T), \\ snd \circ last(tr_T) \end{array}\right)\rangle\right)\right)\rangle \end{array}\right)/tr_T \\ \left(\begin{array}{l} front(\langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle) \\ \frown \\ dif_T\left(\begin{array}{l} difloc\left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array}\right), tr'_T, tr_T\right), \\ \langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle \end{array}\right) \end{array}\right)/tr'_T \end{array}\right]$$

{Property of sequences: $front(\langle e\rangle) = \langle\rangle$}

$$= P\left[\begin{array}{l}\langle\left(\langle\rangle, snd \circ last\left(\langle\left(\begin{array}{l} Flat \circ front(tr_T) \frown fst \circ last(tr_T), \\ snd \circ last(tr_T) \end{array}\right)\rangle\right)\right)\rangle/tr_T \\ dif_T\left(\begin{array}{l} difloc\left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array}\right), tr'_T, tr_T\right), \\ \langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle \end{array}\right)/tr'_T \end{array}\right]$$

{Definition of *last* and *snd*}

$$= P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T \\ dif_T\left(\begin{array}{l} difloc\left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array}\right), tr'_T, tr_T\right), \\ \langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle \end{array}\right)/tr'_T \end{array}\right]$$

{Lemma L.6.0.5}

$$= P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T \\ dif_T\left(\begin{array}{l} difloc\left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array}\right), tr'_T, tr_T\right), \\ \langle(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T))\rangle \end{array}\right)/tr'_T \end{array}\right]$$

{Lemma L.6.0.2}

$$= P\left[\begin{array}{l}\langle(\langle\rangle, snd \circ last(tr_T))\rangle/tr_T \\ dif_T(tr'_T, tr_T)/tr'_T \end{array}\right]$$

{Definition of $\mathbf{R2_T}$}

$$= \mathbf{R2_T}(P)$$

□

Lemma L.6.0.2

$$dif_T \left(\begin{array}{l} difloc \left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array} \right), tr'_T, tr_T \right), \\ \langle (Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T)) \rangle \end{array} \right)$$

$$= dif_T(tr'_T, tr_T)$$

Proof.

$$dif_T \left(\begin{array}{l} difloc \left(\left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array} \right), tr'_T, tr_T \right), \\ \langle (Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T)) \rangle \end{array} \right) \quad \{\text{Lemma L.6.0.4}\}$$

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array} \right) \\ - \\ fst(Flat \circ front(tr_T) \frown fst \circ last(tr_T), snd \circ last(tr_T)) \end{array} \right), \\ snd \circ head(tr'_T - front(s)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_T - front(tr_T)) \end{array} \right)$$

{Definition of *fst*}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} \left(\begin{array}{l} Flat \circ front(tr_T) \\ \frown \\ fst \circ head(tr'_T - front(tr_T)) \end{array} \right) \\ - \\ (Flat \circ front(tr_T) \frown fst \circ last(tr_T)) \end{array} \right), \\ snd \circ head(tr'_T - front(s)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_T - front(tr_T)) \end{array} \right)$$

{Property of sequences}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} (fst \circ head(tr'_T - front(tr_T)) - fst \circ last(tr_T)), \\ snd \circ head(tr'_T - front(s)) \end{array} \right) \rangle \\ \frown \\ tail(tr'_T - front(tr_T)) \end{array} \right) \quad \{\text{Definition of } dif_T\}$$

$$= dif_T(tr'_T, tr_T)$$

□

Lemma L.6.0.3

$$dif_T(t, \langle s \rangle) = \langle (fst(head(t)) - fst(s), snd(head(t))) \rangle \frown tail(t)$$

Proof. Isabelle theorem: `difA_t_lseq_s_rseq`. □

Lemma L.6.0.4

$$dif_T(difloc(c, t, s), \langle z \rangle) = \langle (c - fst(z), snd \circ head(t - front(s))) \rangle \frown tail(t - front(s))$$

Proof.

$dif_T(difloc(c, t, s), \langle z \rangle)$ {Lemma L.6.0.3}

$= \langle (fst(head(difloc(c, t, s))) - fst(z), snd(head(difloc(c, t, s)))) \rangle \frown tail(difloc(c, t, s))$

{Lemma L.6.0.7}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} fst(c, snd \circ head(t - front(s))) - fst(z), \\ snd(c, snd \circ head(t - front(s)) \end{array} \right) \rangle \\ \frown \\ tail(difloc(c, t, s) \end{array} \right)$$

{Definition of *fst* and *snd*}

$$= \left(\begin{array}{l} \langle (c - fst(z), snd \circ head(t - front(s))) \rangle \\ \frown \\ tail(difloc(c, t, s) \end{array} \right) \quad \{\text{Lemma L.6.0.8}\}$$

$= \langle (c - fst(z), snd \circ head(t - front(s))) \rangle \frown tail(t - front(s))$

□

Lemma L.6.0.5

$$dif_T(difloc(c, t, \langle s \rangle), \langle s \rangle) = \langle (c - fst(s), snd \circ head(t)) \rangle \frown tail(t)$$

Proof.

$dif_T(difloc(c, t, \langle s \rangle), \langle s \rangle)$ {Lemma L.6.0.3}

$= \langle (fst(head(difloc(c, t, \langle s \rangle))) - fst(s), snd(head(difloc(c, t, \langle s \rangle)))) \rangle \frown tail(difloc(c, t, \langle s \rangle))$ {Lemma L.6.0.7}

$$= \left(\begin{array}{l} \langle \left(\begin{array}{l} fst(c, snd \circ head(t - front(\langle s \rangle))) - fst(s), \\ snd(c, snd \circ head(t - front(\langle s \rangle)) \end{array} \right) \rangle \\ \frown \\ tail(difloc(c, t, \langle s \rangle)) \end{array} \right)$$

{Definition of *fst* and *snd*}

$$= \left(\begin{array}{l} \langle (c - fst(s), snd \circ head(t - front(\langle s \rangle))) \rangle \\ \frown \\ tail(difloc(c, t, \langle s \rangle)) \end{array} \right)$$ {Lemma L.6.0.8}

$= \langle (c - fst(s), snd \circ head(t - front(\langle s \rangle))) \rangle \frown tail(t - front(\langle s \rangle))$ {Property of sequences}

$= \langle (c - fst(s), snd \circ head(t)) \rangle \frown tail(t)$

□

Lemma L.6.0.6

$$difloc(c, t, s) - front(s)$$

Proof.

$difloc(c, t, s) - front(s)$ {Definition of *difloc*}

$= \langle (c, snd \circ head(t - front(s)) \rangle \frown tail(t - front(s)) - front(s)$

□

Lemma L.6.0.7

$$head(difloc(c, t, s)) = (c, snd \circ head(t - front(s))$$

Proof.

$$\begin{aligned}
&head(difloc(c, t, s)) && \{\text{Definition of } difloc\}\\
&= head(\langle (c, snd \circ head(t - front(s))\rangle \frown tail(t - front(s))) && \{\text{Definition of } head\}\\
&= (c, snd \circ head(t - front(s))
\end{aligned}$$

□

Lemma L.6.0.8

$$tail(difloc(c, t, s) = tail(t - front(s))$$

Proof.

$$\begin{aligned}
&tail(difloc(c, t, s) && \{\text{Definition of } difloc\}\\
&= tail(\langle (c, snd \circ head(t - front(s))\rangle \frown tail(t - front(s))) && \{\text{Definition of } tail\}\\
&= tail(t - front(s))
\end{aligned}$$

□

Chapter 7

Operators

7.1 Skip

The *CSP* process *Skip* can be mapped into the timed model as established by the following Lemma L.7.1.1. The precondition is also *true*, while the postcondition requires immediate termination without any event happening. This is in exact correspondence with the original definition of $Skip_A$.

Lemma L.7.1.1

$$csp2t(Skip_{CSP}) = \mathbf{R_T}(true \vdash \#tr'_T = \#tr_T \wedge Flat(tr'_T) = Flat(tr_T) \wedge \neg\, wait'_T)$$

Proof.

$csp2t(Skip_{CSP})$ {Definition of $Skip_{CSP}$ and $csp2t$}

$= \exists\, st\alpha, untimed\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(true \vdash tr' = tr \wedge \neg\, wait')$ {Lemma L.3.6.9}

$$= \mathbf{R_T}\left(\begin{array}{l}(subsR2(true)[false/wait'] \vee wait'_T) \wedge subsR2(true)[true/wait'] \\ \vdash \\ \left(\begin{array}{l}\left(\begin{array}{l}(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ subsR2(tr' = tr \wedge \neg\ wait')[false/wait'] \\ \wedge \\ \neg\ wait'_T\end{array}\right) \\ \vee \\ subsR2(tr' = tr \wedge \neg\ wait')[true/wait']\end{array}\right)\end{array}\right)$$

{Substitution and definition of $subsR2$}

$$= \mathbf{R_T}\left(\begin{array}{l}(true \vee wait'_T) \wedge true \\ \vdash \\ \left(\begin{array}{l}\left(\begin{array}{l}(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ (Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge \neg\ false) \\ \wedge \\ \neg\ wait'_T\end{array}\right) \\ \vee \\ (Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge \neg\ true)\end{array}\right)\end{array}\right)$$

{Predicate calculus and property of sequences}

$$= \mathbf{R_T}\left(\begin{array}{l}true \\ \vdash \\ \left(\begin{array}{l}(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ Flat(tr'_T) = Flat(tr_T) \\ \wedge \\ \neg\ wait'_T\end{array}\right)\end{array}\right)$$

{Predicate calculus}

$$= \mathbf{R_T}(true \vdash \#tr'_T = \#tr_T \wedge Flat(tr'_T) = Flat(tr_T) \wedge \neg\ wait'_T)$$

□

7.2 Stop

The result of mapping $Stop_{CSP}$ through $csp2t$, however, is different from $Stop_A$. The resulting precondition is also *true*, however the postcondition only requires that no event takes place, irrespective of termination. This is a consequence of the coupling invariant **CI0132**, which only requires that termination in the untimed model leads to termination in the timed model, but does not impose any requirement when the process is waiting.

Lemma L.7.2.1

$$csp2t(Stop_{CSP}) = \mathbf{R_T}(true \vdash Flat(tr'_T) = Flat(tr_T))$$

Proof.

$$
\begin{array}{ll}
csp2t(Stop_{CSP}) & \{\text{Definition of } Stop_{CSP} \text{ and } csp2t\} \\
= \exists\, st\alpha, untimed\alpha \bullet \mathbf{CITR} \circ \mathbf{R}(true \vdash tr' = tr \wedge wait') & \{\text{Lemma L.3.6.9}\} \\
= \mathbf{R_T}\left(\begin{array}{l}
(subsR2(true)[false/wait'] \vee wait'_T) \wedge subsR2(true)[true/wait'] \\
\vdash \\
\left(\begin{array}{l}
\left(\begin{array}{l}
(Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\
\wedge \\
subsR2(tr' = tr \wedge wait')[false/wait'] \\
\wedge \\
\neg\, wait'_T
\end{array}\right) \\
\vee \\
subsR2(tr' = tr \wedge wait')[true/wait']
\end{array}\right)
\end{array}\right) & \\
\multicolumn{2}{r}{\{\text{Substitution and definition of } subsR2\}}
\end{array}
$$

$$= \mathbf{R_T}\left(\begin{array}{l} (true \vee wait'_T) \wedge true \\ \vdash \\ \left(\begin{array}{l} \left(\begin{array}{l} (Flat(tr'_T) = Flat(tr_T) \Rightarrow \#tr'_T = \#tr_T) \\ \wedge \\ (Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge false) \\ \wedge \\ \neg\, wait'_T \end{array}\right) \\ \vee \\ (Flat(tr'_T) - Flat(tr_T) = \langle\rangle \wedge true) \end{array}\right) \end{array}\right)$$

{Predicate calculus and property of sequences}

$$= \mathbf{R_T}(true \vdash Flat(tr'_T) = Flat(tr_T))$$

□

Chapter 8

UTP

This chapter contains results about relations and basic UTP theories. They were mostly sourced from [**?**].

Lemma L.8.0.1

$$P \ ; (Q \lhd c \rhd (R \vee S))$$
$$=$$
$$(P \ ; (Q \lhd c \rhd S)) \vee (P \ ; (\neg\ c \wedge R))$$

Proof.

$P \ ; (Q \lhd c \rhd (R \vee S))$ {Definition of conditional}
$= P \ ; ((c \wedge Q) \vee (\neg\ c \wedge (R \vee S)))$ {Predicate calculus}
$= P \ ; ((c \wedge Q) \vee (\neg\ c \wedge R) \vee (\neg\ c \wedge S))$
{Distributivity of sequential composition}
$= (P \ ; ((c \wedge Q) \vee (\neg\ c \wedge S))) \vee (P \ ; (\neg\ c \wedge R))$ {Definition of conditional}
$= (P \ ; (Q \lhd c \rhd S)) \vee (P \ ; (\neg\ c \wedge R))$

□

Lemma L.8.0.2 *Provided no dashed variables are free in P,*

$$P \wedge ((\neg\ P \wedge Q) \ ; R) = \mathit{false}$$

Proof.

$$
\begin{aligned}
& P \wedge ((\neg\ P \wedge Q)\ ;\ R) && \{\text{Definition of sequential composition}\} \\
&= P \wedge (\exists\, v_0 \bullet (\neg\ P \wedge Q)[v_0/v'] \wedge R[v_0/v]) && \{\text{Assumption and substitution}\} \\
&= P \wedge (\exists\, v_0 \bullet \neg\ P \wedge Q[v_0/v'] \wedge R[v_0/v]) && \{\text{Predicate calculus}\} \\
&= P \wedge (\neg\ P \wedge \exists\, v_0 \bullet Q[v_0/v'] \wedge R[v_0/v]) && \{\text{Predicate calculus}\} \\
&= \mathit{false}
\end{aligned}
$$

□

Lemma L.8.0.3 *Provided no dashed variables are free in P, $(P \wedge Q)\ ;\ R = P \wedge (Q\ ;\ R)$*

Proof.

$$
\begin{aligned}
& (P \wedge Q)\ ;\ R && \{\text{Definition of sequential composition}\} \\
&= \exists\, v_0 \bullet (P \wedge Q)[v_0/v'] \wedge R[v_0/v] && \{\text{Assumption and substitution}\} \\
&= \exists\, v_0 \bullet P \wedge Q[v_0/v'] \wedge R[v_0/v] && \{\text{Predicate calculus}\} \\
&= P \wedge \exists\, v_0 \bullet Q[v_0/v'] \wedge R[v_0/v] && \{\text{Definition of sequential composition}\} \\
&= P \wedge (Q\ ;\ R)
\end{aligned}
$$

□

Lemma L.8.0.4 *Provided no undashed variables are free in R, $(P\ ;\ (Q \wedge R)) \wedge \neg\ R = \mathit{false}$*

Proof.

$$
\begin{aligned}
& (P\ ;\ (Q \wedge R)) \wedge \neg\ R && \{\text{Assumption and Lemma L.8.0.5}\} \\
&= (P\ ;\ Q) \wedge R \wedge \neg\ R && \{\text{Predicate calculus}\} \\
&= \mathit{false}
\end{aligned}
$$

□

Lemma L.8.0.5 *Provided no undashed variables are free in R, $(P\ ;\ Q) \wedge R = P\ ;\ (Q \wedge R)$*

Proof.

$$
(P\ ;\ Q) \wedge R \qquad \{\text{Definition of sequential composition}\}
$$

$= (\exists\, v_0 \bullet P[v_0/v'] \wedge Q[v_0/v]) \wedge R$ {Assumption and predicate calculus}

$= \exists\, v_0 \bullet P[v_0/v'] \wedge Q[v_0/v] \wedge R$ {Assumption and substitution}

$= \exists\, v_0 \bullet P[v_0/v'] \wedge (Q \wedge R)[v_0/v]$ {Definition of sequential composition}

$= P \;;\; (Q \wedge R)$

□

Lemma L.8.0.6✓$_P$ ✓$_A$ $\quad (P \wedge Q) \Leftrightarrow P = P \Rightarrow Q$

Proof.

$(P \wedge Q) \Leftrightarrow P$ {Predicate calculus}

$= ((P \wedge Q) \Rightarrow P) \wedge (P \Rightarrow (P \wedge Q))$ {Predicate calculus}

$= (P \Rightarrow (P \wedge Q))$ {Predicate calculus}

$= P \Rightarrow Q$

□

Lemma L.8.0.7✓$_P$ ✓$_A$ $\quad (P \vee Q) \Leftrightarrow (P \vee R) = P \vee (Q \Leftrightarrow R)$

Proof.

$(P \vee Q) \Leftrightarrow (P \vee R)$ {Predicate calculus}

$= ((P \vee Q) \Rightarrow (P \vee R)) \wedge ((P \vee R) \Rightarrow (P \vee Q))$ {Predicate calculus}

$= (P \Rightarrow (P \vee R)) \wedge (Q \Rightarrow (P \vee R)) \wedge (P \Rightarrow (P \vee Q)) \wedge (R \Rightarrow (P \vee Q))$ {Predicate calculus}

$= (Q \Rightarrow (P \vee R)) \wedge (R \Rightarrow (P \vee Q))$ {Predicate calculus}

$= (\neg\, Q \vee P \vee R) \wedge (\neg\, R \vee P \vee Q)$ {Predicate calculus}

$= P \vee ((\neg\, Q \vee R) \wedge (\neg\, R \vee Q))$ {Predicate calculus}

$= P \vee (Q \Leftrightarrow R)$

□

Lemma L.8.0.8

$$(P \Rightarrow Q) \Leftrightarrow (P \Rightarrow R) = P \Rightarrow (Q \Leftrightarrow R)$$

Proof.

$$
\begin{aligned}
&(P \Rightarrow Q) \Leftrightarrow (P \Rightarrow R) && \{\text{Predicate calculus}\}\\
&= (\neg\, P \vee Q) \Leftrightarrow (\neg\, P \vee R) && \{\text{Lemma L.8.0.7}\}\\
&= \neg\, P \vee (Q \Leftrightarrow R) && \{\text{Predicate calculus}\}\\
&= P \Rightarrow (Q \Leftrightarrow R)
\end{aligned}
$$

□

Lemma L.8.0.9

$$(P \wedge Q) \Leftrightarrow (P \wedge R) = P \Rightarrow (Q \Leftrightarrow R)$$

Proof.

$$
\begin{aligned}
&(P \wedge Q) \Leftrightarrow (P \wedge R) && \{\text{Predicate calculus}\}\\
&= ((P \wedge Q) \Rightarrow (P \wedge R)) \wedge ((P \wedge R) \Rightarrow (P \wedge Q)) && \{\text{Predicate calculus}\}\\
&= ((P \wedge Q) \Rightarrow P) \wedge ((P \wedge Q) \Rightarrow R) \wedge ((P \wedge R) \Rightarrow P) \wedge ((P \wedge R) \Rightarrow Q)\\
& && \{\text{Predicate calculus}\}\\
&= ((P \wedge Q) \Rightarrow R) \wedge ((P \wedge R) \Rightarrow Q) && \{\text{Predicate calculus}\}\\
&= (\neg\, P \vee \neg\, Q \vee R) \wedge (\neg\, P \vee \neg\, R \vee Q) && \{\text{Predicate calculus}\}\\
&= \neg\, P \vee ((\neg\, Q \vee R) \wedge (\neg\, R \vee Q)) && \{\text{Predicate calculus}\}\\
&= \neg\, P \vee ((Q \Rightarrow R) \wedge (R \Rightarrow Q)) && \{\text{Predicate calculus}\}\\
&= \neg\, P \vee (Q \Leftrightarrow R) && \{\text{Predicate calculus}\}\\
&= P \Rightarrow (Q \Leftrightarrow R)
\end{aligned}
$$

□

Lemma L.8.0.10 *Provided b is a Boolean variable,*

$$(P \wedge b')\ ;\ (Q \wedge b) = (P \wedge b')\ ;\ Q$$

Proof.

$$
\begin{aligned}
&(P \wedge b')\ ;\ (Q \wedge b) && \{\text{Definition of sequential composition}\}
\end{aligned}
$$

$= \exists\, b_0, v_0 \bullet (P \wedge b')[b_0, v_0/b', v'] \wedge (Q \wedge b)[b_0, v_0/b, v]$ {Substitution}

$= \exists\, b_0, v_0 \bullet P[b_0, v_0/b', v'] \wedge b_0 \wedge Q[b_0, v_0/b, v] \wedge b_0$
{Assumption and predicate calculus}

$= \exists\, b_0, v_0 \bullet P[b_0, v_0/b', v'] \wedge b_0 \wedge Q[b_0, v_0/b, v]$ {Substitution}

$= \exists\, b_0, v_0 \bullet (P \wedge b')[b_0, v_0/b', v'] \wedge Q[b_0, v_0/b, v]$
{Definition of sequential composition}

$= (P \wedge b')\ ;\ Q$

□

Lemma L.8.0.11

$$P\ ;\ (Q \wedge R) \Rightarrow P\ ;\ Q$$

Proof.

$P\ ;\ (Q \wedge R)$ {Definition of sequential composition}

$= \exists\, v_0 \bullet P[v_0/v'] \wedge (Q \wedge R)[v_0/v]$ {Substitution}

$= \exists\, v_0 \bullet P[v_0/v'] \wedge Q[v_0/v] \wedge R[v_0/v]$ {Predicate calculus}

$\Rightarrow (\exists\, v_0 \bullet P[v_0/v'] \wedge Q[v_0/v]) \wedge (\exists\, v_0 \bullet R[v_0/v])$ {Predicate calculus}

$\Rightarrow (\exists\, v_0 \bullet P[v_0/v'] \wedge Q[v_0/v])$ {Definition of sequential composition}

$= P\ ;\ Q$

□

Chapter 9

Isabelle/UTP Mechanisation

theory *super-theory-alpha*

imports
utp-reactive
utp-deduct
begin

9.1 Alphabet

A timed trace

type-synonym $'\alpha$ *timedtrace* = ($'\alpha$ *trace* × $'\alpha$ *refusal*) *list*

record $'\vartheta$ *alpha-st* = $'\vartheta$ *alpha-rp* +
st-wait :: *bool*
st-trT :: $'\vartheta$ *timedtrace*
st-trC :: $'\vartheta$ *trace*

definition $wait_T$ = *VAR st-wait*
definition tr_T = *VAR st-trT*
definition tr_C = *VAR st-trC*

declare $wait_T$*-def* [*upred-defs*]
declare tr_T*-def* [*upred-defs*]
declare tr_C*-def* [*upred-defs*]

type-synonym $('\vartheta,'\alpha)$ *alphabet-st* $= ('\vartheta,'\alpha)$ *alpha-st-scheme alphabet*
type-synonym $('\vartheta,'\alpha,'\beta)$ *relation-st* $= (('\vartheta,'\alpha)$ *alphabet-st*, $('\vartheta,'\beta)$ *alphabet-st*$)$ *relation*
type-synonym $('\vartheta,'\alpha)$ *hrelation-st* $= (('\vartheta,'\alpha)$ *alphabet-st*, $('\vartheta,'\alpha)$ *alphabet-st*$)$ *relation*
type-synonym $('\vartheta,'\sigma)$ *predicate-st* $= ('\vartheta,'\sigma)$ *alphabet-st upred*
type-synonym $('a,'\vartheta,'\alpha)$ *huexpr-st* $=$ $('a,$ $('\vartheta,'\alpha)$ *alpha-st-scheme* $\times$ $('\vartheta,'\alpha)$ *alpha-st-scheme*$)$ *uexpr*
type-synonym $('a,'\vartheta,'\alpha,'\beta)$ *uexpr-st* $=$ $('a,$ $('\vartheta,'\alpha)$ *alpha-st-scheme* $\times$ $('\vartheta,'\beta)$ *alpha-st-scheme*$)$ *uexpr*

lemma *uvar-wait* [*simp*]: *uvar* $wait_T$
by (*unfold-locales*, *simp-all add*: $wait_T$*-def*)
(*metis* (*no-types*, *lifting*) *alpha-d.ext-inject alpha-rp.ext-inject alpha-st.ext-inject alpha-st.surjective alpha-st.update-convs*(*1*))

lemma *uvar-tr* [*simp*]: *uvar* tr_T
by (*unfold-locales*, *simp-all add*: tr_T*-def*)
(*metis* (*no-types*, *lifting*) *alpha-d.ext-inject alpha-rp.ext-inject alpha-st.ext-inject alpha-st.surjective alpha-st.update-convs*(*2*))

lemma *uvar-trc* [*simp*]: *uvar* tr_C
by (*unfold-locales*, *simp-all add*: tr_C*-def*)
(*metis* (*no-types*, *lifting*) *alpha-d.ext-inject alpha-rp.ext-inject alpha-st.ext-inject alpha-st.surjective alpha-st.update-convs*(*3*))

lemma tr_T*-*$wait_T$*-indep* [*simp*]: $tr_T \bowtie wait_T$ $wait_T \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma tr_T*-wait_indep* [*simp*]: $tr_T \bowtie wait$ $wait \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma tr_T*-tr-indep* [*simp*]: $tr_T \bowtie tr$ $tr \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma tr_T*-*tr_C*-indep* [*simp*]: $tr_T \bowtie tr_C$ $tr_C \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma tr_T-*ok-indep* [*simp*]: $tr_T \bowtie ok\ ok \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma tr_T-*ref-indep* [*simp*]: $tr_T \bowtie ref\ ref \bowtie tr_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

lemma $wait_T$-*ok-indep* [*simp*]: $wait_T \bowtie ok\ ok \bowtie wait_T$
by (*simp add*: *uvar-indep-def*, *pred-tac*)+

end
theory *super-theory-lists-pairs*

imports
utp-theory
begin

type-synonym ('*a*, '*b*) *listPairs* = ('*a* × '*b*) *list*
type-synonym ('*a*, '*b*) *listPairs-List* = ('*a list*, '*b*) *listPairs*

9.2 Timed traces

Because timed traces are a list of pairs, whose first component is a list itself, in this section we define the HOL type and prove useful results about it.

9.2.1 Definition of dif_T and *Flat*

In this encoding, we define dif_T and *Flat* at the level of HOL, rather than at the UTP level. The advantage is that results to do with timed traces can be proved using the HOL list type directly, rather than having to transfer all the time. UTP lemmas can still use these results by transferring appropriately.

definition *difLP* :: ('*a*, '*b*) *listPairs-List* ⇒ ('*a*, '*b*) *listPairs-List* ⇒ ('*a*, '*b*) *listPairs-List* (*difLP'*(-,-') *67*)
where *difLP t s* ≡ [(*fst*(*hd*(*t* − *butlast*(*s*))) − *fst*(*last*(*s*)),*snd*(*hd*(*t* − *butlast*(*s*))))] • *tl*(*t* − *butlast*(*s*))

definition *FlatLP* :: ('*a*, '*b*) *listPairs-List* ⇒ '*a list* (*FlatLP'*(-') *66*)

where *FlatLP tx* ≡ *concat* (*map fst tx*)

declare *difLP-def* [*upred-defs*]
declare *FlatLP-def* [*upred-defs*]

declare [[*show-sorts*]]

lemma *prefix-ord* [*simp*]: $s < t \longleftrightarrow$ *prefix s t*
by (*metis dual-order.strict-iff-order less-eq-list-def prefix-def*)

lemma *prefixeq-ord* [*simp*]: $s \leq t \longleftrightarrow$ *prefixeq s t*
by (*simp add*: *less-eq-list-def*)

lemma *last-difLP-t-s--eq--last-t-one*:
assumes *butlast*(*s*) < *t* **and** *length s* > *0* **and** *length t* > *length s*
shows *last*(*difLP*(*t*,*s*)) = *last*(*t*)
using *assms*
by (*simp add*:*prefix-ord difLP-def butlast-prefix-length-lt-imp-last-tl-minus-butlast-eq-last tl-list-minus-butlast-not-empty*)

lemma *last-difLP-t-s--eq--last-t-two*:
assumes *butlast*(*s*) < *t* **and** *length s* > *0* **and** *length t* = *length s*
shows *last*(*difLP*(*t*,*s*)) = (*fst*(*last*(*t*)) − *fst*(*last*(*s*)),*snd*(*last*(*t*)))
proof −
have *last*(*difLP*(*t*,*s*)) = *last*([(*fst*(*hd*(*t* − *butlast*(*s*))) − *fst*(*last*(*s*)),*snd*(*hd*(*t* − *butlast*(*s*))))]
• *tl*(*t* − *butlast*(*s*)))
by (*simp add*:*difLP-def*)
also have ... = *last*([(*fst*(*hd*(*t* − *butlast*(*s*))) − *fst*(*last*(*s*)),*snd*(*hd*(*t* − *butlast*(*s*))))])
using *assms*
by (*simp add*: *prefixeq-def tl-list-minus-butlast-empty*)
also have ... = (*fst*(*hd*(*t* − *butlast*(*s*))) − *fst*(*last*(*s*)),*snd*(*hd*(*t* − *butlast*(*s*))))
by *simp*
also have ... = (*fst*(*hd*([*last*(*t*)])) − *fst*(*last*(*s*)),*snd*(*hd*([*last*(*t*)])))
using *assms*
by (*simp add*: *assms*(*1*) *assms*(*3*) *list-minus-butlast-eq-butlast-list*)
also have ... = (*fst*(*last*(*t*)) − *fst*(*last*(*s*)),*snd*(*last*(*t*)))

by *simp*

finally show *?thesis* **.**
qed

lemma *snd-last-difLP-t-s--eq--snd-last-t-two*:
assumes *butlast*(*s*) < *t* **and** *length s* > *0* **and** *length t* ≥ *length s*
shows *snd*(*last*(*difLP*(*t*,*s*))) = *snd*(*last*(*t*))
using *assms*
using *last-difLP-t-s--eq--last-t-one last-difLP-t-s--eq--last-t-two* **by** *fastforce*

lemma *difLP-length-t-eq-lengths--eq--fst*:
assumes *butlast*(*s*) < *t* **and** *length t* = *length s*
shows *difLP*(*t*,*s*) = [(*fst*(*last*(*t*)) − *fst*(*last*(*s*)),*snd*(*last*(*t*)))]
using *assms*
by (*simp add*:*difLP-def list-minus-butlast-eq-butlast-list*)

lemma *snd-last-difLP--eq--snd-last*:
assumes *butlast*(*s*) < *t*
shows *snd*(*last*(*difLP*(*t*,*s*))) = *snd*(*last*(*t*))
using *assms*
by (*simp add*:*difLP-def*, *metis* (*no-types*, *lifting*) *hd-Cons-tl length-gt-zero-last-concat last-snoc last-tl append-minus prefix-eq-exists prefix-minus-not-empty*)

lemma *last-difLP-t-s--eq--fst-last-t-minus-fst-last-s--snd-last-t*:
assumes *length t* = *length s* **and** *butlast*(*s*) < *t*
shows *last*(*difLP*(*t*,*s*)) = (*fst*(*last*(*t*)) − *fst*(*last*(*s*)),*snd*(*last*(*t*)))
using *assms*
by (*simp add*: *difLP-length-t-eq-lengths--eq--fst*)

lemma *difLP-t-cat-s-t-cat-u--eq--difLP-s-u*:
assumes *length u* > *0* **and** *butlast*(*u*) ≤ *s*
shows *difLP*(*t* • *s*, *t* • *u*) = *difLP*(*s*, *u*)
proof −
have *difLP*(*t* • *s*, *t* • *u*) =
[(*fst* (*hd*((*t* • *s*) − *butlast*(*t* • *u*))) − *fst* (*last*(*t* • *u*)),
snd (*hd*((*t* • *s*) − *butlast*(*t* • *u*))))] • *tl*((*t* • *s*) − *butlast*(*t* • *u*))

```
    by (simp only:difLP-def)
  also have ... = [(fst (hd((t • s) − (t • butlast(u)))) − fst (last(t • u)),
    snd (hd((t • s) − (t • butlast(u))))] •
    tl((t • s) − (t • butlast(u)))
    using assms(1)
    by (metis (erased, hide-lams) length-gt-zero-butlast-concat)
  also have ... = [(fst (hd((t • s) − (t • butlast(u)))) − fst (last(u)),
    snd (hd((t • s) − (t • butlast(u))))] •
    tl((t • s) − (t • butlast(u)))
    using assms(1)
    by simp
  also have ... = [(fst (hd(s − butlast(u))) − fst (last(u)),
    snd (hd(s − butlast(u)))] •
    tl(s − butlast(u))
    using assms(1) assms(2)
    by (metis prefixeq-ord concat-minus-list-concat-butlast-eq-list-minus-butlast)
  also have ... = difLP(s, u)
    by (simp add:difLP-def)
  finally show ?thesis .
qed

lemma difLP-s-butlast-u-cat--fst-last-u--eq--difLP-s-u:
  shows difLP(s, butlast(u) • [(fst(last(u)),r)]) = difLP(s, u)
  by (metis butlast-snoc difLP-def fst-conv last-snoc)

lemma difLP-s-t--eq--difLP-s-u:
  assumes butlast(u) = butlast(t) and fst(last(t)) = fst(last(u))
  shows difLP(s,t) = difLP(s,u)
  by (metis assms(1) assms(2) difLP-def)

lemma difLP-difLP-t-s-empty-snd-last-s--eq--difLP-t-s:
  assumes butlast(s) < t and length s > 0
  shows difLP(difLP(t,s),[([],snd(last(s)))]) = difLP(t,s)
proof −
  have difLP(difLP(t,s),[([],snd(last(s)))])
       =
       [(fst(hd(difLP(t,s) − butlast([([],snd(last(s)))]))))
```

$-$
$fst(last([([],snd(last(s)))])),$
$snd(hd(difLP(t,s) - butlast([([],snd(last(s)))])))]$
$\bullet$
$tl(difLP(t,s) - butlast([([],snd(last(s)))]))$
by (*simp add:difLP-def*)
also have ... = $[(fst(hd(difLP(t,s) - []))$
$-$
$fst(last([([],snd(last(s)))])),$
$snd(hd(difLP(t,s) - [])))]$
$\bullet$
$tl(difLP(t,s) - [])$
by (*metis butlast-single-element*)
also have ... = $[(fst(hd(difLP(t,s) - []))$
$-$
$fst(([],snd(last(s)))),$
$snd(hd(difLP(t,s) - [])))]$
$\bullet$
$tl(difLP(t,s) - [])$
by (*metis last-single-element*)
also have ... =
$[(fst(hd(difLP(t,s))) - fst(([],snd(last(s)))),$
$snd(hd(difLP(t,s))))]$
$\bullet$
$tl(difLP(t,s))$
by (*metis minus-right-nil*)
also have ... =
$[(fst(hd(difLP(t,s))),snd(hd(difLP(t,s))))]$
$\bullet$
$tl(difLP(t,s))$
by (*metis fst-conv minus-right-nil*)
also have ... = $[hd(difLP(t,s))] \bullet tl(difLP(t,s))$
by (*metis prod.collapse*)
also have ... = $difLP(t,s)$
by (*metis append-Cons append-Nil difLP-def list.collapse list.distinct(1)*)

finally show *?thesis* **.**

qed

lemma
assumes *butlast*(*s*) < *t* **and** *length s* > *0*
shows *fst*(*hd*(*difLP*(*t*,*s*))) = *fst*(*hd*(*t* − *butlast*(*s*))) − *fst*(*last*(*s*))
by (*metis append-Cons difLP-def fst-conv list.sel*(*1*))

9.2.2 Results about *Flat*

lemma *FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*:
shows *FlatLP*(*s* • *t*) = (*FlatLP*(*s*) • *FlatLP*(*t*))
by (*simp add*:*FlatLP-def*)

lemma *FlatLP-seq-s--eq--seq-s*:
shows *FlatLP*([([*s*],*t*)]) = [*s*]
by (*simp add*:*FlatLP-def*)

lemma *FlatLP-s-non-empty--eq--fst-hd-s-cat-FlatLP-tl-s*:
assumes *length s* > *0*
shows *FlatLP*(*s*) = *fst*(*hd*(*s*)) • *FlatLP*(*tl*(*s*))
by (*metis FlatLP-def assms concat.simps*(*2*) *length-greater-0-conv list.collapse list.simps*(*9*))

lemma *FlatLP-s-non-empty--eq--FlatLP-butlast-s-cat-fst-last-s*:
assumes *length s* > *0*
shows *FlatLP*(*s*) = *FlatLP*(*butlast*(*s*)) • *fst*(*last*(*s*))
by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t FlatLP-s-non-empty--eq--fst-hd-s-cat-FlatLP-tl-s append-butlast-last-id assms butlast-snoc last-snoc less-numeral-extra*(*3*) *list.distinct*(*1*) *list.size*(*3*) *prefix-simps*(*2*) *prefixeq-length-less rotate1.simps*(*2*) *rotate1-hd-tl self-append-conv*)

lemma
assumes *length s* > *0* **and** *length t* > *0* **and** *length z* > *0*
shows *FlatLP*(*s*) = [] ∧ *FlatLP*(*z*) = [] ∧ *length s* = *length z* ∧ *butlast*(*s*) < *t* ∧ *butlast*(*z*) < *t*
⟶
fst(*last*(*s*)) = *fst*(*last*(*z*)) ∧ *butlast*(*s*) = *butlast*(*z*)
by (*metis prefix-ord FlatLP-s-non-empty--eq--FlatLP-butlast-s-cat-fst-last-s append-is-Nil-conv assms*(*1*) *butlast-eq-if-eq-length-and-prefix*)

lemma *FlatLP-hd-s--eq--fst-hd-s*:
assumes *length s > 0*
shows *FlatLP*([*hd*(*s*)]) = *fst*(*hd*(*s*))
by (*simp add*:*FlatLP-def*)

lemma *FlatLP-last-s--eq--fst-last-s*:
assumes *length s > 0*
shows *FlatLP*([*last*(*s*)]) = *fst*(*last*(*s*))
by (*simp add*:*FlatLP-def*)

lemma *FlatLP-t-minus-s--eq--FlatLP-t-minus-FlatLP-s*:
assumes $s \leq t$
shows $FlatLP(t - s) = FlatLP(t) - FlatLP(s)$
using *assms*
by (*metis prefixeq-ord FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t prefixeq-concat-minus append-minus*)

lemma *FlatLP-t-minus-butlast-s--eq--FlatLP-t-minus-FlatLP-butlast-s*:
assumes *butlast*(*s*) < *t* **and** *length s > 0*
shows $FlatLP(t - butlast(s)) = FlatLP(t) - FlatLP(butlast(s))$
using *assms*(*1*)
by (*simp add*: *FlatLP-t-minus-s--eq--FlatLP-t-minus-FlatLP-s assms*(*1*) *dual-order.strict-implies-order*)

lemma *FlatLP-t-minus-s-minus-FlatLP-z--eq--FlatLP-t-minus-FlatLP-s-cat-z*:
assumes $s \leq t$ **and** $(s \bullet z) \leq t$
shows $FlatLP(t-s) - FlatLP(z) = FlatLP(t) - FlatLP(s \bullet z)$
using *assms*
by (*metis prefixeq-ord FlatLP-t-minus-s--eq--FlatLP-t-minus-FlatLP-s prefixeq-concat-minus append-minus strict-prefixI prefixeq-minus-concat*)

lemma *FlatLP-t-minus-s-minus-FlatLP-z--eq--FlatLP-t-minus-FlatLP-s-cat-z-lt*:
assumes $s < t$ **and** $(s \bullet z) < t$
shows $FlatLP(t-s) - FlatLP(z) = FlatLP(t) - FlatLP(s \bullet z)$
using *assms*
by (*simp add*: *FlatLP-t-minus-s-minus-FlatLP-z--eq--FlatLP-t-minus-FlatLP-s-cat-z*)

lemma *FlatLP-t-minus-s-minus-FlatLP-z--eq--FlatLP-t-minus-FlatLP-s-cat-z-le*:

assumes $s < t$ **and** $(s \bullet z) \leq t$
shows $FlatLP(t-s) - FlatLP(z) = FlatLP(t) - FlatLP(s \bullet z)$
using *assms*
by (*simp add*: *FlatLP-t-minus-s-minus-FlatLP-z--eq--FlatLP-t-minus-FlatLP-s-cat-z dual-order.strict-im*

lemma
assumes $butlast(s) < t$ **and** $length\ s > 0$ **and** $fst(last(s)) \leq fst(hd(t - butlast(s)))$
shows $FlatLP(t - butlast(s)) \bullet FlatLP([last(s)]) = FlatLP(t) - FlatLP(s)$
oops

lemma *fst-last-s--le--fst-hd-t-minus-butlast-s*:
assumes $butlast(s) < t$ **and** $length\ s > 0$ **and** $fst(last(s)) \leq fst(hd(t - butlast(s)))$
shows $FlatLP([last(s)]) \leq FlatLP(t - butlast(s))$
using *assms*
by (*simp add*:*FlatLP-def*, (*metis prefixeq-ord concat.simps*(*2*) *hd-Cons-tl list.simps*(*9*) *prefix-prefix prefix-minus-not-empty*))

lemma *FlatLP-t-minus-FlatLP-s--eq--FlatLP-t-minus-butlast-s--minus--FlatLP-last-s*:
assumes $butlast(s) < t$ **and** $length\ s > 0$ **and** $fst(last(s)) \leq fst(hd(t - butlast(s)))$
shows $FlatLP(t) - FlatLP(s) = FlatLP(t - butlast(s)) - FlatLP([last(s)])$
proof −
have $FlatLP(t) - FlatLP(s) = FlatLP(t) - FlatLP(butlast(s) \bullet [last(s)])$
by (*metis append-butlast-last-id assms*(*2*) *length-greater-0-conv*)
also have $... = FlatLP(butlast(s) \bullet (t - butlast(s))) - FlatLP(butlast(s) \bullet [last(s)])$
using *assms*(*1*)
by (*metis dual-order.strict-implies-order append-minus strict-prefixE*)
also have $... = (FlatLP(butlast(s)) \bullet FlatLP(t - butlast(s))) - FlatLP(butlast(s) \bullet [last(s)])$
by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have $... = (FlatLP(butlast(s)) \bullet FlatLP(t - butlast(s))) - (FlatLP(butlast(s)) \bullet FlatLP([last(s)]))$
by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have $... = FlatLP(t - butlast(s)) - FlatLP([last(s)])$
using *assms*(*1*)
by (*simp add*:*list-concat-minus-list-concat fst-last-s--le--fst-hd-t-minus-butlast-s*)

finally show *?thesis* **.**
qed

The following is Lemma 24 from the initial draft on CircusTime.

lemma *FlatLP-fidLP-eq-FlatLP-minus-FlatLP*:
assumes $butlast(s) < t$ **and** $length\ s > 0$ **and** $fst(last(s)) \leq fst(hd(t - butlast(s)))$
shows $FlatLP(difLP(t,s)) = FlatLP(t) - FlatLP(s)$
proof −

next

have *thesis*:$FlatLP(difLP(t,s)) = FlatLP(t) - FlatLP(s)$
proof (*cases length s < length t*)

have *FlatLPminus*: $FlatLP([last(s)]) \leq FlatLP([hd(t - butlast(s))])$
using *assms*
by (*simp add*: *FlatLP-hd-s--eq--fst-hd-s FlatLP-last-s--eq--fst-last-s prefix-minus-not-empty*)

show $length\ s < length\ t \Longrightarrow FlatLP(difLP(t,s)) = FlatLP(t) - FlatLP(s)$
proof −
have $FlatLP(difLP(t,s)) = FlatLP([(fst\ (hd(t - butlast(s))) - fst\ (last(s)), snd\ (hd(t - butlast(s))))] \bullet$
$tl(t - butlast(s)))$
by (*simp add*:*difLP-def*)
also have ... $= FlatLP([(fst\ (hd(t - butlast(s))) - fst\ (last(s)), snd\ (hd(t - butlast(s))))]) \bullet$
$FlatLP(tl(t - butlast(s)))$
by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have ... $= (fst\ (hd(t - butlast(s))) - fst\ (last(s))) \bullet$
$FlatLP(tl(t - butlast(s)))$
by (*simp add*:*FlatLP-def*)
also have ... $= (fst\ (hd(t - butlast(s))) - FlatLP([last(s)])) \bullet$
$FlatLP(tl(t - butlast(s)))$
by (*simp add*:*FlatLP-def*)
also have ... $= (FlatLP([hd(t - butlast(s))]) - FlatLP([last(s)])) \bullet$
$FlatLP(tl(t - butlast(s)))$
by (*simp add*:*FlatLP-def*)
also have ... $= (FlatLP([hd(t - butlast(s))]) \bullet$

FlatLP(*tl*(*t* − *butlast*(*s*)))) − *FlatLP*([*last*(*s*)])
by (*smt FlatLPminus append-assoc append-minus strict-prefixE*)

also have ... = (*FlatLP*([*hd*(*t* − *butlast*(*s*))] •
tl(*t* − *butlast*(*s*))) − *FlatLP*([*last*(*s*)]))
by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have ... = (*FlatLP*(*t* − *butlast*(*s*)) − *FlatLP*([*last*(*s*)]))
by (*metis* (*no-types*, *lifting*) *prefixeq-ord append-Cons append-Nil append-Nil2 assms*(*1*) *dual-order.strict-iff-order hd-Cons-tl append-minus prefixeq-eq-exists*)
also have ... = *FlatLP*(*t*) − *FlatLP*(*butlast*(*s*) • [*last*(*s*)])
using *assms*
by (*simp add*:*FlatLP-t-minus-FlatLP-s--eq--FlatLP-t-minus-butlast-s--minus--FlatLP-last-s*)
also have ... = *FlatLP*(*t*) − *FlatLP*(*s*)
using *assms*(*2*)
by (*metis append-butlast-last-id less-numeral-extra*(*3*) *list.size*(*3*))
finally show *?thesis* **.**
qed

next
show *length s* = *length t* ⟹ *FlatLP*(*difLP*(*t*,*s*)) = *FlatLP*(*t*) − *FlatLP*(*s*)

proof −
assume *0*: *length s* = *length t*
have *FlatLP*(*difLP*(*t*,*s*)) = *FlatLP*([(*fst* (*hd*(*t* − *butlast*(*s*))) − *fst* (*last*(*s*)), *snd* (*hd*(*t* − *butlast*(*s*))))] •
tl(*t* − *butlast*(*s*)))
by (*simp add*:*difLP-def*)
also have ... = *FlatLP*([(*fst* (*hd*(*t* − *butlast*(*s*))) − *fst* (*last*(*s*)), *snd* (*hd*(*t* − *butlast*(*s*))))])
using *assms* **and** *0*
by (*simp add*:*tl-list-minus-butlast-eq-empty*)
also have ... = *fst* (*hd*(*t* − *butlast*(*s*))) − *fst* (*last*(*s*))
using *assms* **and** *0*
by (*metis FlatLP-hd-s--eq--fst-hd-s fst-conv length-greater-0-conv list.distinct*(*1*) *list.sel*(*1*))
also have ... = *FlatLP*(*t* − *butlast*(*s*)) − *fst*(*last*(*s*))
using *assms* **and** *0*

by (*metis prefix-ord FlatLP-hd-s--eq--fst-hd-s length-greater-0-conv list.distinct*(*1*) *list.sel*(*1*) *list-minus-butlast-eq-butlast-list*)

also have ... = (*FlatLP*(*t*) − *FlatLP*(*butlast*(*s*))) − *fst*(*last*(*s*))

using *assms*

by (*metis FlatLP-t-minus-butlast-s--eq--FlatLP-t-minus-FlatLP-butlast-s*)

also have ... = (*FlatLP*(*t*) − *FlatLP*(*butlast*(*s*))) − *FlatLP*([*last*(*s*)])

by (*metis FlatLP-last-s--eq--fst-last-s assms*(*2*))

also have ... = *FlatLP*(*t*) − (*FlatLP*(*butlast*(*s*)) • *FlatLP*([*last*(*s*)]))

by (*metis FlatLP-last-s--eq--fst-last-s FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t FlatLP-t-minus-FlatLP-s-*
‹*FlatLP*(*t* − *butlast*(*s*)) − *fst* (*last*(*s*)) = (*FlatLP*(*t*) − *FlatLP*(*butlast*(*s*))) − *fst* (*last*(*s*))›
append-butlast-last-id assms(*1*) *assms*(*2*) *assms*(*3*) *less-nat-zero-code list.size*(*3*))

also have ... = *FlatLP*(*t*) − *FlatLP*(*butlast*(*s*) • [*last*(*s*)])

by (*metis FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t*)

also have ... = *FlatLP*(*t*) − *FlatLP*(*s*)

using *assms*

by (*metis append-butlast-last-id length-greater-0-conv*)

finally show *?thesis* **.**

qed

next

show *length s* < *length t* ⟹ *length s* < *length t*

by *auto*

show ¬ *length s* < *length t* ⟹ *length s* = *length t*

using *assms*(*1*) *assms*(*2*) *length-not-gt-iff-eq-length* **by** *auto*

qed

then

show *FlatLP*(*difLP*(*t*,*s*)) = *FlatLP*(*t*) − *FlatLP*(*s*) **using** *assms* **by** (*simp*)

qed

Lemma 26

lemma *length-difLP-eq-length-dif-plus-one*:

assumes *butlast*(*s*) < *t* **and** *length s* > *0*

shows *length*(*difLP*(*t*,*s*)) = (*length t* − *length s*) + *1*

proof −

have *length*(*difLP*(*t*,*s*)) = *length*(*tl*(*t*− *butlast*(*s*))) + *1*

by (*simp add*:*difLP-def*)

also have ... = *length*(*t*− *butlast*(*s*)) − *1* + *1*

by *auto*
also have ... = *length*(*t*− *butlast*(*s*))
using *assms*
by (*simp add*:*prefix-minus-not-empty*)
also have ... = *length*(*t*) − *length*(*butlast*(*s*))
using *assms*
by (*simp add*:*length-list-minus*)
also have ... = *length*(*t*) − *length*(*s*) + *1*
using *assms*
by (*metis* (*erased*, *hide-lams*) *Suc-eq-plus1 Suc-pred* ‹*length* (*t* − *butlast*(*s*)) = *length t* − *length* (*butlast*(*s*))› ‹*length* (*t* − *butlast*(*s*)) − *1* + *1* = *length* (*t* − *butlast*(*s*))› *diff-diff-left length-butlast monoid-add-class.add.left-neutral*)
finally show *?thesis* **.**
qed

lemma *fst-last-lt-fst-hd-diff-implies-FlatLP-prefix*:
assumes *butlast*(*s*) < *t fst*(*last*(*s*)) ≤ *fst*(*hd*(*t*−*butlast*(*s*)))
shows *FlatLP*(*s*) ≤ *FlatLP*(*t*)
using *assms*
by (*smt FlatLP-last-s--eq--fst-last-s FlatLP-s-cat-t--eq--Flat-s-cat-Flat-t FlatLP-s-non-empty--eq--FlatLP-Prefix-Order.prefixE append-minus butlast.simps*(*1*) *fst-last-s--le--fst-hd-t-minus-butlast-s length-greater-0 order.strict-implies-order same-prefix-prefix strict-prefixI*)

lemma *FlatLP-eq-length-eq-front-prefix-and-snd-last-eq*:
assumes *length s* > *0*
shows *FlatLP*(*s*) = *FlatLP*(*t*) ∧ (*length s*) = (*length t*) ∧ (*butlast s*) < *t* ∧ *snd*(*last*(*s*)) = *snd*(*last*(*t*)) ⟷ *t* = *s*
using *assms* **unfolding** *FlatLP-def*
by (*smt FlatLP-def FlatLP-last-s--eq--fst-last-s FlatLP-t-minus-s--eq--FlatLP-t-minus-FlatLP-s Prefix-Order.prefixE append-butlast-last-id append-eq-append-conv append-minus dual-order.strict-iff-orde length-butlast length-greater-0-conv list.distinct*(*1*) *prod.collapse self-append-conv strict-prefixI*)

end
theory *super-theory-healths*

imports
super-theory-alpha

super-theory-lists-pairs
begin

9.3 Healthiness Conditions

The following is a rough tactic to consume lemmas proved using bindings.

method *bind-tac* =
(*simp only*:*utp-pred.inf.assoc*)*?* ;
((*rule ueqe3-imp* | *simp only*: *utp-pred.inf.commute*, *rule ueqe3-imp*)*?*, ((*erule uconjE*)+)*?*)*?*
;
(*simp only*:*uexpr-eq-iff conj-bind-dist disj-bind-dist imp-bind-dist*)*?*)

9.3.1 Lifting *Flat* and dif_T

Here we lift the UTP $Flat_u$ and dif_T to their HOL counterparts, *FlatLP* and *difLP*, respectively.

syntax
-utrflat :: ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a trace*, 'α) *uexpr* ($Flat_u$'(-'))
-utrflat :: ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a trace*, 'α) *uexpr* ($Flat_u$)
-utrdifu :: ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* (dif_T'(-,-') *90*)

translations
$Flat_u$(*cs*) == *CONST uop CONST FlatLP cs*
$Flat_u$ *cs* == *CONST uop CONST FlatLP cs*
dif_T(*t*,*s*) == *CONST bop CONST difLP t s*

9.3.2 $difloc_T$ and $Expands_T$

First we define the auxiliary predicates $Expands_T$ and *difloc*.

definition *ExpandsT* :: ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* ⇒ (*bool*, 'α) *uexpr* ($Expands_T$'(-,-') *67*)
where *ExpandsT s t* = ($front_u(s) <_u t \wedge \pi_1(last_u(s)) \leq_u \pi_1(head_u(t - front_u(s)))$)

definition *difloc* :: ('*a trace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* ⇒ ('*a timedtrace*, 'α) *uexpr* ($difloc_T$'(-,-,-') *92*)

where *difloc c t s* = $\langle(c,\pi_2(head_u(t-front_u(s))))_u\rangle$
$\hat{}_u$
$tail_u(t-front_u(s))$

9.3.3 Results about *UFlat* and $Flat_u$

lemma *front-cat-last*:
assumes '$\#_u(s) >_u 0$'
shows $front_u(s)\ \hat{}_u\ \langle last_u(s)\rangle = s$
using *assms* **unfolding** *upred-defs*
by *pred-tac*

lemma *front-single-empty* [*simp*]:
$front_u(\langle s\rangle) = \langle\rangle$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *simp add*:*butlast-def*)

lemma *ulast-single* [*simp*]:
$last_u(\langle s\rangle) = s$
by (*transfer*, *simp*)

lemma *pair-snd* [*simp*]:
$\pi_2((s,t)_u) = t$
by *pred-tac*

lemma *pair-fst* [*simp*]:
$\pi_1((s,t)_u) = s$
by *pred-tac*

lemma *Flatu-s-cat-t--eq--Flat-s-cat-Flat-t*:
shows $Flat_u(s\ \hat{}_u\ t) = Flat_u(s)\ \hat{}_u\ Flat_u(t)$
by (*transfer*, *simp add*:*FlatLP-def*)

lemma *Flatu-seq-s--eq--seq-s*:
shows $Flat_u(\langle(\langle s\rangle,t)_u\rangle) = \langle s\rangle$
by (*transfer*, *simp add*:*FlatLP-def*)

lemma *Flatu-last-s--eq--fst-last-s*:
shows $Flat_u(\langle last_u(s)\rangle) = \pi_1(last_u(s))$
by (*transfer*, *simp add*:*FlatLP-def*)

lemma *Flatu-s-non-empty--eq--fst-head-s-cat-FlatA-tail-s*:
assumes $\#_u(s) >_u 0 = true$
shows $Flat_u(s) = \pi_1(head_u(s)) \ ^\frown_u \ Flat_u(tail_u(s))$
using *assms* **unfolding** *upred-defs*
apply *transfer*
apply (*simp add*:*FlatLP-def*)
by (*metis* (*no-types*, *hide-lams*) *concat.simps*(*2*) *hd-Cons-tl map-eq-Cons-conv*)

lemma *FlatU-difT-FlatU-minus-FlatU*:
assumes ‘$\#_u(t_T) >_u 0$‘ **and** ‘$front_u(t_T) <_u t_T{}'$‘
and ‘$Expands_T(t_T,t_T{}')$‘
shows $Flat_u(dif_T(t_T{}',t_T)) = Flat_u(t_T{}') - Flat_u(t_T)$
using *assms* **unfolding** *upred-defs*
apply (*simp add*:*ExpandsT-def*)
apply *pred-tac*
apply (*subst FlatLP-fidLP-eq-FlatLP-minus-FlatLP*)
by (*simp-all*)

lemma *binding-FlatU-difT-FlatU-minus-FlatU*:
assumes $[\![\#_u(\$tr_T) >_u 0]\!]_e\ b$ **and** $[\![front_u(\$tr_T) <_u \$tr_T{}']\!]_e\ b$
and $[\![Expands_T(\$tr_T,\$tr_T{}')]\!]_e\ b$
shows $[\![Flat_u(dif_T(\$tr_T{}',\$tr_T))]\!]_e\ b = [\![Flat_u(\$tr_T{}') - Flat_u(\$tr_T)]\!]_e\ b$
using *assms* **unfolding** *upred-defs*
apply (*simp add*:*ExpandsT-def*)
apply *pred-tac*
by (*simp add*:*FlatLP-fidLP-eq-FlatLP-minus-FlatLP*)

lemma *Flatu-s-non-empty--eq--FlatA-front-s-cat-fst-last-s*:
shows $(\#_u(s) >_u 0 \Rightarrow (Flat_u(s) =_u Flat_u(front_u(s)) \ ^\frown_u \ \pi_1(last_u(s)))) = true$
proof −
have $(\#_u(s) >_u 0 \Rightarrow (Flat_u(s) =_u Flat_u(front_u(s)) \ ^\frown_u \ \pi_1(last_u(s))))$
$=$
$(\#_u(s) >_u 0 \Rightarrow (Flat_u(s) =_u Flat_u(front_u(s)) \ ^\frown_u \ Flat_u(\langle last_u(s)\rangle)))$

by (*simp add*:*Flatu-last-s--eq--fst-last-s*)
also have ... =
(#$_u$(*s*) >$_u$ *0* ⇒ (*Flat*$_u$(*s*) =$_u$ *Flat*$_u$(*front*$_u$(*s*) ^$_u$ ⟨*last*$_u$(*s*)⟩)))
by (*simp add*:*Flatu-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have ... =
(#$_u$(*s*) >$_u$ *0* ⇒ (*Flat*$_u$(*s*) =$_u$ *Flat*$_u$(*s*)))
apply (*simp add*:*front-cat-last*)
by *pred-tac*
also have ... = (#$_u$(*s*) >$_u$ *0* ⇒ *true*)
by *pred-tac*
also have ... = *true*
by *simp*

finally show *?thesis* **by** *pred-tac*
qed

lemma *binding-Flatu-s-non-empty--eq--FlatA-front-s-cat-fst-last-s*:
fixes *b* :: ('ϑ, 'α) *alpha-st-scheme* × ('ϑ, 'α) *alpha-st-scheme*
assumes ⟦#$_u$(*s*) >$_u$ *0*⟧$_e$ *b*
shows ⟦*Flat*$_u$(*s*) =$_u$ *Flat*$_u$(*front*$_u$(*s*)) ^$_u$ π$_1$(*last*$_u$(*s*))⟧$_e$ *b*
proof −
have ⟦*Flat*$_u$(*front*$_u$(*s*)) ^$_u$ π$_1$(*last*$_u$(*s*))⟧$_e$ *b* = ⟦*Flat*$_u$(*front*$_u$(*s*)) ^$_u$ *Flat*$_u$(⟨*last*$_u$(*s*)⟩)⟧$_e$ *b*
by (*simp add*:*Flatu-last-s--eq--fst-last-s*)
also have ... = ⟦*Flat*$_u$(*front*$_u$(*s*) ^$_u$ ⟨*last*$_u$(*s*)⟩)⟧$_e$ *b*
by (*simp add*:*Flatu-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have ... = ⟦*Flat*$_u$(*s*)⟧$_e$ *b*
using *assms* **unfolding** *upred-defs*
by *pred-tac*

finally show *?thesis* **by** *pred-tac*
qed

lemma *impl-Flatu-s-non-empty--eq--FlatA-front-s-cat-fst-last-s*:
assumes '#$_u$(*s*) >$_u$ *0* ∧ *s* =$_u$ *b*'
shows (*Flat*$_u$(*s*) = *Flat*$_u$(*front*$_u$(*s*)) ^$_u$ π$_1$(*last*$_u$(*s*))) ∧ *s* = *b*
proof −

have $Flat_u(front_u(s))\ \hat{}_u\ \pi_1(last_u(s)) = Flat_u(front_u(s))\ \hat{}_u\ Flat_u(\langle last_u(s)\rangle)$
by (*simp add*:*Flatu-last-s--eq--fst-last-s*)
also have ... = $Flat_u(front_u(s)\ \hat{}_u\ \langle last_u(s)\rangle)$
by (*simp add*:*Flatu-s-cat-t--eq--Flat-s-cat-Flat-t*)
also have ... = $Flat_u(s)$
using *assms* **unfolding** *upred-defs*
apply *pred-tac*
by (*metis append-butlast-last-id*)

finally show *?thesis* **using** *assms* **unfolding** *upred-defs* **by** *pred-tac*
qed

lemma *Flatu-head-s--eq--fst-head-s*:
shows $Flat_u(\langle head_u(s)\rangle) = \pi_1(head_u(s))$
by (*transfer*, *simp add*:*FlatLP-def*)

lemma *FlatA-t-minus-s--eq--FlatA-t-minus-FlatA-s*:
assumes ‘$s \leq_u t$‘
shows $Flat_u(t - s) = Flat_u(t) - Flat_u(s)$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *simp add*:*FlatLP-t-minus-s--eq--FlatLP-t-minus-FlatLP-s*)

9.3.3.1 Results on $difloc_T$

We show that $difloc_T$ and dif_T are related.

lemma *difT-eq-diflocT*: $dif_T(t,s) = difloc_T(\pi_1(head_u(t{-}front_u(s)))-\pi_1(last_u(s)),t,s)$
apply (*simp add*:*difloc-def*)
apply *pred-tac*
by (*simp add*:*difLP-def*)

lemma dif_t*-single-tr*: $dif_T(t, \langle s\rangle) = \langle(\pi_1(head_u(t)) - \pi_1(s), \pi_2(head_u(t)))_u\rangle\ \hat{}_u\ tail_u(t)$
by (*pred-tac*, *simp add*:*difLP-def*)

lemma dif_t*-difloc-single-tr*: $dif_T(difloc_T(c, t, s), \langle z\rangle) = \langle(c - \pi_1(z), \pi_2(head_u(t{-}front_u(s))))_u\rangle$
$\hat{}_u\ tail_u(t{-}front_u(s))$
by (*simp add*:*difloc-def*, *pred-tac*, *simp add*: *difLP-def*)

lemma *dif$_t$-difloc$_t$-same-single-tr*:
dif$_T$(*difloc$_T$*(*c*, *t*, ⟨*s*⟩), ⟨*s*⟩) = ⟨(*c* − π_1(*s*), π_2(*head$_u$*(*t*)))$_u$⟩ ^$_u$ *tail$_u$*(*t*)
apply (*simp add*:*dif$_t$-single-tr difloc-def*)
by *pred-tac*

L.8.9.2

lemma *dif$_T$-difloc$_T$-eq-dif$_T$*:
shows *dif$_T$*(*difloc$_T$*(*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))), *t$_T$*′,*t$_T$*)
,
⟨(*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*last$_u$*(*t$_T$*)),π_2(*last$_u$*(*t$_T$*)))$_u$⟩)
= *dif$_T$*(*t$_T$*′,*t$_T$*)
proof −
have *dif$_T$*(*difloc$_T$*(*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))), *t$_T$*′,*t$_T$*)
,
⟨(*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*last$_u$*(*t$_T$*)),π_2(*last$_u$*(*t$_T$*)))$_u$⟩)
=
⟨((*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*)))
−
π_1((*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*last$_u$*(*t$_T$*)),π_2(*last$_u$*(*t$_T$*)))$_u$)),
π_2(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))))$_u$⟩ ^$_u$ *tail$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))
by (*simp add*:*dif$_t$-difloc-single-tr*)
also have ... =
⟨((*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))))
−
(*Flat$_u$*(*front$_u$*(*t$_T$*)) ^$_u$ π_1(*last$_u$*(*t$_T$*))),
π_2(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))))$_u$⟩ ^$_u$ *tail$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))
by *simp*
also have ... =
⟨(π_1(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))) − π_1(*last$_u$*(*t$_T$*)), π_2(*head$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))))$_u$⟩
^$_u$ *tail$_u$*(*t$_T$*′−*front$_u$*(*t$_T$*))
apply *pred-tac*
using *list-concat-minus-list-concat* **by** *blast*
also have ... = *dif$_T$*(*t$_T$*′,*t$_T$*)
by (*pred-tac*, *simp add*:*difLP-def*)

finally show *?thesis* **.**
qed

9.3.4 Some Results on dif_T and UTP lists

lemma *length-one*: $\#_u(\langle s \rangle) = 1$
by *pred-tac*

lemma $head_u$*-single*: $head_u(\langle s \rangle) = s$
by (*pred-tac*)

lemma $head_u$*-single-concat*: $head_u(\langle s \rangle \ ^\frown_u \ t) = s$
by *pred-tac*

lemma *seq-minus-empty*: $s - \langle\rangle = s$
by *pred-tac*

lemma *minus-append*: $(s \ \# \ t) - (s \ \# \ z) = t - z$
by (*simp add*:*minus-list-def*)

lemma *minus-concat*: $(s \bullet t) - (s \bullet z) = t - z$
by (*simp add*:*minus-list-def*)

lemma *seq-dif*: $(s - t = z \wedge t \leq s) \longleftrightarrow (t \bullet z = s)$
apply (*simp add*:*minus-list-def*)
apply *auto*
using *prefixeq-drop* **by** *fastforce*

lemma $head_u$*-dif*$_T$: $head_u(dif_T(t, s)) = (\pi_1(head_u(t{-}front_u(s))){-}\pi_1(last_u(s)),\pi_2(head_u(t{-}front_u(s))))_u$
apply (*pred-tac*)
by (*simp add*:*difLP-def*)

lemma *sequence-strict-prefix*:
$s < t \longleftrightarrow (\exists\, xs \ .\ s \bullet xs = t \wedge (length\ xs) > 0)$
by (*metis length-greater-0-conv less-list-def same-prefix-nil strict-prefixE strict-prefixI*)

lemma *length-t-minus-front-s*:
assumes ‘$front_u(s) <_u t$‘ **and** ‘$\#_u(s) >_u 0$‘
shows ‘$\#_u(t - \ front_u(s)) >_u 0$‘
using *assms* **unfolding** *upred-defs*

by (*transfer*, *metis append-Nil2 append-minus length-greater-0-conv less-le strict-prefixE*)

lemma *length-t-minus-s--eq--length-t-minus-length-s*:
fixes $t :: ('b\ list, 'a)\ uexpr$
assumes ‘$s <_u t$‘
shows $\#_u(t - s) = (\#_u(t) - \#_u(s))$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *metis length-drop minus-list-def prefixeq-def sequence-strict-prefix*)

lemma *binding-length-t-minus-s--eq--length-t-minus-length-s*:
fixes $t :: ('b\ list, 'a)\ uexpr$
assumes $[\![s <_u t]\!]_e\ b$
shows $[\![\#_u(t - s)]\!]_e\ b = [\![\#_u(t) - \#_u(s)]\!]_e\ b$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *metis length-drop minus-list-def prefixeq-def sequence-strict-prefix*)

lemma *binding-length-front-s*:
assumes $[\![\#_u(s) >_u 0]\!]_e\ b$
shows $[\![\#_u(front_u(s))]\!]_e\ b = [\![\#_u(s) - 1]\!]_e\ b$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *simp*)

lemma *length-front-s*:
assumes ‘$\#_u(s) >_u 0$‘
shows $\#_u(front_u(s)) = \#_u(s) - 1$
using *assms* **unfolding** *upred-defs*
by (*transfer*, *simp*)

lemma *front-s-implies-length-s-lt-length-t*:
assumes *length s* $>$ *0*
shows (*butlast s*) $<$ *t* $\longrightarrow$ *length s* $\leq$ *length t*
using *assms*
apply *auto*
using *butlast-prefix-imp-length-not-gt le-less-linear* **by** *auto*

lemma *utp-front-s-implies-length-s-lt-length-t*:
assumes ‘$\#_u(s) >_u 0$‘

shows *‘front$_u$(s) <$_u$ t ⇒ #$_u$(s) ≤$_u$ #$_u$(t)‘*
using *assms* **unfolding** *upred-defs*
apply *transfer*
by (*simp add*:*front-s-implies-length-s-lt-length-t*)

lemma *binding-length-dif$_T$-eq-length-t-minus-length-s-plus-one*:
assumes ⟦*front$_u$(s) <$_u$ t*⟧$_e$ *b* **and** ⟦#$_u$(s) >$_u$ *0*⟧$_e$ *b*
shows ⟦#$_u$(*dif$_T$*(t,s))⟧$_e$ *b* = ⟦((#$_u$(t) − #$_u$(s)) + *1*)⟧$_e$ *b*
proof −
have ⟦#$_u$(*dif$_T$*(t,s))⟧$_e$ *b* = ⟦#$_u$(*tail$_u$*(t− *front$_u$*(s))) + *1*⟧$_e$ *b*
by (*pred-tac*, *simp add*:*difLP-def*)
also have ... = ⟦#$_u$(t− *front$_u$*(s)) − *1* + *1*⟧$_e$ *b*
by (*pred-tac*)
also have ... = ⟦#$_u$(t− *front$_u$*(s))⟧$_e$ *b*
using *assms* **unfolding** *upred-defs*
apply (*simp add*:*length-t-minus-front-s*)
apply *pred-tac*
by (*simp add*: *prefix-minus-not-empty*)
also have ... = ⟦#$_u$(t) − #$_u$(*front$_u$*(s))⟧$_e$ *b*
using *assms*(*1*)
by (*subst binding-length-t-minus-s--eq--length-t-minus-length-s*, *auto*)
also have ... = ⟦#$_u$(t) − (#$_u$(s) − *1*)⟧$_e$ *b*
using *assms*(*2*)
by (*simp add*: *binding-length-front-s bop.rep-eq minus-uexpr-def*)
also have ... = ⟦(#$_u$(t) − #$_u$(s)) + *1*⟧$_e$ *b*

using *assms* **unfolding** *upred-defs*
apply (*simp add*:*front-s-implies-length-s-lt-length-t*)
apply *pred-tac*
by (*metis Nitpick.size-list-simp*(*2*) *One-nat-def Suc-diff-Suc length-butlast length-tl prefixeq-length-less*)

finally show *?thesis* **.**
qed

lemma *s-cat-t-minus-s-cat-z-eq-t-minus-z*:
shows (*s* ^$_u$ *t*) − (*s* ^$_u$ *z*) = *t* − *z*

using *assms* **unfolding** *upred-defs*
apply *transfer*
by (*simp add*:*minus-concat*)

lemma *t-gte-front-eq-gt-front*:
$(t_T{}' \geq_u front_u(t_T) \wedge \#_u(t_T) >_u 0 \wedge \#_u(t_T{}') \geq_u \#_u(t_T))$
$=$
$(t_T{}' >_u front_u(t_T) \wedge \#_u(t_T) >_u 0 \wedge \#_u(t_T{}') \geq_u \#_u(t_T))$
apply *pred-tac*
by (*metis append-butlast-last-id butlast-conv-take length-butlast list.distinct*(*2*)
prefix-order.order.not-eq-order-implies-strict self-append-conv take-all)

lemma *lengtht-eq-lengths*:
fixes $s :: ('a\ list, 'b)\ uexpr$ **and** $t :: ('a\ list, 'b)\ uexpr$
assumes $[\![\#_u(s) \leq_u \#_u(t)]\!]_e\ b$
shows $[\![\#_u(t) =_u \#_u(s)]\!]_e\ b = [\![x =_u \#_u(t) - \#_u(s) + x]\!]_e\ b$
using *assms*
by *pred-tac*

lemma *length-dif*$_T$*-eq-length-t-minus-length-s-plus-one*:
assumes ‘$front_u(s) <_u t$‘ **and** ‘$\#_u(s) >_u 0$‘
shows $\#_u(dif_T(t,s)) = ((\#_u(t) - \#_u(s)) + 1)$
proof −
have $\#_u(dif_T(t,s)) = \#_u(tail_u(t- front_u(s))) + 1$
by (*pred-tac*, *simp add*:*difLP-def*)
also have ... $= (\#_u(t- front_u(s)) - 1 + 1)$
by (*pred-tac*)
also have ... $= \#_u(t- front_u(s))$
using *assms* **unfolding** *upred-defs*
apply (*simp add*:*length-t-minus-front-s*)
apply *pred-tac*
by (*simp add*: *prefix-minus-not-empty*)
also have ... $= \#_u(t) - \#_u(front_u(s))$
using *assms*
by (*simp add*:*length-t-minus-s--eq--length-t-minus-length-s*)
also have ... $= \#_u(t) - (\#_u(s) - 1)$
using *assms*

by (*simp add*:*length-front-s*)
also have ... = ($\#_u(t) - \#_u(s)$) + *1*

using *assms* **unfolding** *upred-defs*
apply (*simp add*:*front-s-implies-length-s-lt-length-t*)
apply *pred-tac*
by (*metis Nitpick.size-list-simp*(*2*) *One-nat-def Suc-diff-Suc length-butlast*
length-tl prefixeq-length-less)

finally show *?thesis* **.**
qed

9.3.5 Healthiness Conditions of Circus Time

Then we introduce the definitions of **R0T**, **R1T**, **R2T**, **R3T**.

definition *R0T-def* [*upred-defs*]: *R0T* (*P*) = ($P \wedge \#_u(\$tr_T) >_u 0 \wedge \#_u(\$tr_T) \leq_u \#_u(\$tr_T')$)

definition *R1T-def* [*upred-defs*]: *R1T* (*P*) = ($P \wedge Expands_T(\$tr_T, \$tr_T')$)

definition *R2T-def* [*upred-defs*]: *R2T* (*P*) = $P[\![\langle(\langle\rangle:_u{'}a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T', \$tr_T)/\$tr_T, \$$

definition II_T*-def* [*urel-defs*]: $II_T = (R1T(\neg\ \$ok) \vee (\$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T'))$

definition *R3T-def* [*upred-defs*]: *R3T* (*P*) = ($II_T \triangleleft \$wait_T \triangleright P$)

9.3.6 Healthiness Conditions of the Super-Theory

definition *TR0-def* [*upred-defs*]: *TR0* (*P*) = ($P \wedge \#_u(\$tr_T) >_u 0$)

definition *TR1-def* [*upred-defs*]: *TR1* (*P*) ≡ ($P \wedge \#_u(\$tr_T) \leq_u \#_u(\$tr_T')$)

definition *TR2-def* [*upred-defs*]: *TR2* (*P*) ≡ ($P \wedge front_u(\$tr_T) \leq_u \$tr_T'$)

definition *TR3-def* [*upred-defs*]: *TR3* (*P*) = ($P \wedge ((\$ok \wedge \$wait_T) \Rightarrow (\#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T')))$

definition *TR4-def* [*upred-defs*]:
$TR4\ (P) = P[\![\langle(Flat_u(\$tr_T),\pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T)\ \hat{}_u\ \langle head_u(\$tr_T{}'-front_u(\$tr_T))\rangle), \pi_2(head_u(\$tr_T{}'-front_u(\$tr_T))))_u\rangle$
$\hat{}_u\ tail_u(\$tr_T{}'-front_u(\$tr_T))$
$/\$tr_T,\$tr_T{}'\!]\!]$

definition *TR4-alt* [*upred-defs*]:
$TR4alt\ (P) = P[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))), \pi_2(head_u(\$tr_T{}'-front_u(\$tr_T))))_u\rangle$
$\hat{}_u\ tail_u(\$tr_T{}'-front_u(\$tr_T))$
$/\$tr_T,\$tr_T{}'\!]\!]$

definition *R2loc$_T$-def* [*upred-defs*]:
$R2loc_T\ (P) = P[\![(front_u(\$tr_T)\ \hat{}_u\ \langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle),(front_u(\$tr_T)\ \hat{}_u\ dif_T(\$tr_T{}',\$tr_T))/\$tr_T,\$tr_T$

definition *R1C-def* [*upred-defs*]:
$R1_C(P) = (P \wedge \$tr_C \leq_u \$tr_C{}')$

The composition of the healthiness conditions **TR0** to **TR4** and including **R2loc$_T$** is **TR**.

definition *TR-def* [*upred-defs*]: $TR\ (P) \equiv TR0(TR1(TR2(TR3(TR4(R2loc_T(R1_C(P)))))))$

definition *TRP* [*upred-defs*]: $TRP\ (P) \equiv TR(R2loc_T(P))$

abbreviation *TR0123* :: (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr*
⇒ (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr* **where**
$TR0123(P) \equiv TR0(TR1(TR2(TR3(P))))$

9.3.6.1 Results on TR0

lemma *TR0-implies-length-gt-zero*:
fixes *P* :: ('*ϑ*,'*α*) *hrelation-st*
shows ‘$TR0(P)$‘ ⟶ ‘$\#_u(\$tr_T) >_u 0$‘
apply (*simp add*:*TR0-def*)
apply (*simp add*:*ueval unrest*)
apply *pred-tac*
by (*metis alpha-st.select-convs*(*2*) *tr$_T$-def uvar.select-convs*(*1*))

lemma *TR0-inside*: $TR0\ (TR1\ (TR2(TR3(P)))) = TR0\ (TR1\ (TR2(TR3(TR0(P)))))$
by *pred-tac*

lemma *TR0-TR1-eq-R0T*: $TR0\ (TR1\ (P)) = R0T(P)$
by *pred-tac*

9.3.6.2 Results on TR4

lemma *binding-TR4-eq-TR4alt*:
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
fixes $b :: ('\vartheta,\ '\alpha)$ *alpha-st-scheme* $\times$ $('\vartheta,\ '\alpha)$ *alpha-st-scheme*
assumes $[\![\#_u(\$tr_T) >_u 0]\!]_e\ b$
shows $[\![TR4(P)]\!]_e\ b = [\![TR4alt(P)]\!]_e\ b$
proof −
have $[\![TR4(P)]\!]_e\ b = [\![P[\![\langle(Flat_u(\$tr_T),\pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T)\ \hat{}_u\ \langle head_u(\$tr_T{}'-front_u(\$tr_T))\rangle), \pi_2(head_u(\$tr_T{}'-front_u(\$tr_T))))_u\rangle$
$\hat{}_u\ tail_u(\$tr_T{}'-front_u(\$tr_T))$
$/\$tr_T,\$tr_T{}']\!]]_e\ b$
by (*simp add*:*TR4-def*)
also have ... =
$[\![P[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\ \pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T)\ \hat{}_u\ \langle head_u(\$tr_T{}'\ -\ front_u(\$tr_T))\rangle),$
$\pi_2(head_u(\$tr_T{}'\ -\ front_u(\$tr_T))))_u\rangle$
$\hat{}_u\ tail_u(\$tr_T{}'-front_u(\$tr_T))/\$tr_T,\$tr_T{}']\!]]_e\ b$
apply (*insert binding-Flatu-s-non-empty--eq--FlatA-front-s-cat-fst-last-s*
[**where** *s*=*iuvar tr*$_T$ **and** *b*=*b*])
using *assms*
by *pred-tac*
also have ... =
$[\![P[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\ \pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ Flat_u(\langle head_u(\$tr_T{}'\ -\ front_u(\$tr_T))\rangle),$
$\pi_2(head_u(\$tr_T{}'\ -\ front_u(\$tr_T))))_u\rangle\ \hat{}_u\ tail_u(\$tr_T{}'-front_u(\$tr_T))/\$tr_T,\$tr_T{}']\!]]_e$
b
apply (*subst Flatu-s-cat-t--eq--Flat-s-cat-Flat-t*)
by *auto*
also have ... =
$[\![P[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\ \pi_2(last_u(\$tr_T)))_u\rangle,$

$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))),$
$\pi_2(head_u(\$tr_T{}' - front_u(\$tr_T))))_u\rangle\ \hat{}_u\ tail_u(\$tr_T{}' - front_u(\$tr_T))/\$tr_T,\$tr_T{}'\rrbracket\!\rrbracket_e$
b

by (*subst Flatu-head-s--eq--fst-head-s*, *auto*)
also have ... = $[\![TR4alt(P)]\!]_e\ b$
by (*simp add*:*TR4-alt*)

finally show *?thesis* **.**
qed

lemma *TR4-R2loc$_T$-eq-R2T*:
fixes P :: $('\vartheta,'\alpha)$ *hrelation-st*
fixes b :: $('\vartheta,\ '\alpha)$ *alpha-st-scheme* × $('\vartheta,\ '\alpha)$ *alpha-st-scheme*
assumes $[\![\#_u(\$tr_T) >_u 0]\!]_e\ b$
shows $[\![TR4(R2loc_T(P))]\!]_e\ b = [\![R2T(P)]\!]_e\ b$
proof −
have $[\![TR4(R2loc_T(P))]\!]_e\ b = [\![TR4alt(R2loc_T(P))]\!]_e\ b$
using *assms*
apply (*subst binding-TR4-eq-TR4alt*)
by (*pred-tac*, *auto*)
also have ... =
$[\![P[\![front_u(\$tr_T)\ \hat{}_u\ \langle(\langle\rangle,\ \pi_2(last_u(\$tr_T)))_u\rangle,$
$front_u(\$tr_T)\ \hat{}_u\ dif_T(\$tr_T{}',\$tr_T)$
$/\$tr_T,\$tr_T{}'\rrbracket$
$[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle,$
$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))),\ \pi_2(head_u(\$tr_T{}' - front_u(\$tr_T))))_u\rangle$
$\hat{}_u\ tail_u(\$tr_T{}' - front_u(\$tr_T))$
$/\$tr_T,\$tr_T{}'\rrbracket$
$]\!]_e\ b$
by (*simp add*:*TR4-alt R2loc$_T$-def*)
also have ... =
$[\![P[\![front_u(\$tr_T)\ \hat{}_u\ \langle(\langle\rangle,\ \pi_2(last_u(\$tr_T)))_u\rangle,$
$front_u(\$tr_T)\ \hat{}_u\ dif_T(\$tr_T{}',\$tr_T)$
$/\$tr_T,\$tr_T{}'\rrbracket$
$[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle,$
$difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$
$/\$tr_T,\$tr_T{}'\rrbracket$

$]\!]_e\ b$

by (*simp add*:*difloc-def*)

also have ... =

$[\![P[\![(front_u(\$tr_T)\ \hat{}_u\ \langle(\langle\rangle,\ \pi_2(last_u(\$tr_T)))_u\rangle)$

$[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle,$

$difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$

$/\$tr_T,\$tr_T{}']\!],$

$(front_u(\$tr_T)\ \hat{}_u\ dif_T(\$tr_T{}',\$tr_T))$

$[\![\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle,$

$difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$

$/\$tr_T,\$tr_T{}']\!]$

$/\$tr_T,\$tr_T{}']\!]$

$]\!]_e\ b$

by *subst-tac*

also have ... =

$[\![P[\![(front_u(\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle)$

$\hat{}_u$

$\langle(\langle\rangle,\ \pi_2(last_u(\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle))_u\rangle)$

,

$(front_u(\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle)$

$\hat{}_u$

$dif_T(difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$

,

$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle)$

)

$/\$tr_T,\$tr_T{}']\!]$

$]\!]_e\ b$

by (*subst-tac*)

also have ... =

$[\![P[\![\langle(\langle\rangle,\ \pi_2(last_u(\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle))_u\rangle$

,

$(dif_T(difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$

,

$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle)$

)

$/\$tr_T,\$tr_T{}']\!]$

$]\!]_e\ b$

by (*simp*)
also have ... =
$[\![P[\![\langle(\langle\rangle,\ \pi_2(last_u(\$tr_T)))_u\rangle,$
$dif_T(difloc_T(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(head_u(\$tr_T{}'-front_u(\$tr_T))),\ \$tr_T{}',\$tr_T)$
,
$\langle(Flat_u(front_u(\$tr_T))\ \hat{}_u\ \pi_1(last_u(\$tr_T)),\pi_2(last_u(\$tr_T)))_u\rangle)$
$/\$tr_T,\$tr_T{}']\!]$
$]\!]_e\ b$
by *simp*
also have ... = $[\![P[\![\langle(\langle\rangle,\ \pi_2(last_u(\$tr_T)))_u\rangle,dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!]]\!]_e\ b$
by (*simp add*:dif_T-$difloc_T$-*eq*-dif_T)
also have ... = $[\![R2T(P)]\!]_e\ b$
by (*simp add*:*R2T-def*)

finally show *?thesis* **using** *assms* **by** *pred-tac*
qed

lemma
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $[\![(TR4\ (R2loc_T\ P) \wedge \#_u(\$tr_T) >_u 0)]\!]_e\ b = [\![R2T(P) \wedge \#_u(\$tr_T) >_u 0]\!]_e\ b$
proof −
have $[\![(TR4\ (R2loc_T\ P) \wedge \#_u(\$tr_T) >_u 0)]\!]_e\ b$
=
$([\![(TR4\ (R2loc_T\ P) \wedge \#_u(\$tr_T) >_u 0)]\!]_e\ b$
$\wedge$
$([\![\#_u(\$tr_T) >_u 0]\!]_e\ b \longrightarrow [\![TR4(R2loc_T(P))]\!]_e\ b = [\![R2T(P)]\!]_e\ b))$
by (*simp add*:*TR4-R2loc*$_T$-*eq-R2T*)
have $[\![(TR4\ (R2loc_T\ P) \wedge \#_u(\$tr_T) >_u 0)]\!]_e\ b$
=
$[\![R2T(P) \wedge \#_u(\$tr_T) >_u 0]\!]_e\ b$

by (*meson TR4-R2loc*$_T$-*eq-R2T conj-implies uconjI*)
oops

lemma
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $[\![TR0(TR4\ (R2loc_T\ P))]\!]_e\ b = [\![TR0(R2T(P))]\!]_e\ b$

apply (*simp add*:*TR0-def*)
by (*meson TR4-R2loc$_T$-eq-R2T conj-implies uconjI*)

lemma *TR0-TR4-R2loc-TR0-R2T*:
fixes $P :: ('\vartheta, '\alpha)$ *hrelation-st*
shows $TR0(TR4\ (R2loc_T\ P)) = TR0(R2T(P))$
apply (*simp add*:*TR0-def*)
apply *bind-tac*
apply (*meson TR4-R2loc$_T$-eq-R2T conj-implies uconjI*)
done

9.3.7 Properties of the Healthiness Conditions

9.3.7.1 Results on $\mathbf{R0_T}$

lemma *R0T-conj*: $R0T(P \wedge Q) = (R0T(P) \wedge Q)$
by (*simp add*:*R0T-def*, *pred-tac*)

lemma *R0T-distr-conj*: $R0T(P \wedge Q) = (R0T(P) \wedge R0T(Q))$
by (*simp add*:*R0T-def*, *pred-tac*)

lemma *R0T-distr-disj*: $R0T(P \vee Q) = (R0T(P) \vee R0T(Q))$
by (*simp add*:*R0T-def*, *pred-tac*)

lemma *R0T-idempotent*: $R0T(R0T(P)) = R0T(P)$
by (*simp add*:*R0T-def*)

lemma *R0T-imp2*: $R0T(P \Rightarrow Q) = R0T(P \Rightarrow R0T(Q))$
by (*simp add*: *R0T-distr-disj R0T-idempotent impl-alt-def*)

9.3.7.2 Results on $\mathbf{R1_T}$

lemma *R1T-distr-conj*: $R1T(P \wedge Q) = (R1T(P) \wedge R1T(Q))$
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *R1T-distr-disj*: $R1T(P \vee Q) = (R1T(P) \vee R1T(Q))$
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *R1T-conj*: $R1T(P \wedge Q) = (R1T(P) \wedge Q)$
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *R1T-idempotent*: $R1T(R1T(P)) = R1T(P)$
by (*simp add*:*R1T-def*)

lemma *R1T-imp2*: $R1T(P \Rightarrow Q) = R1T(P \Rightarrow R1T(Q))$
by (*simp add*: *R1T-distr-disj R1T-idempotent impl-alt-def*)

lemma *R0T-R1T-commute*: $R0T(R1T(P)) = R1T(R0T(P))$
by (*simp add*:*R0T-def R1T-def ExpandsT-def*, *pred-tac*)

9.3.7.3 Results on $\mathbf{R2_T}$

lemma *R2T-distr-conj*: $R2T(P \wedge Q) = (R2T(P) \wedge R2T(Q))$
by (*simp add*:*R2T-def*, *subst-tac*)

lemma *R2T-distr-disj*: $R2T(P \vee Q) = (R2T(P) \vee R2T(Q))$
by (*simp add*:*R2T-def*, *subst-tac*)

lemma *R2T-ok-true*: $R2T(P^t) = (R2T(P))^t$
proof −
have $(R2T(P))^t = (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle,dif_T(\$tr_T\acute{},\$tr_T)/\$tr_T,\$tr_T\acute{}]\!][\![true/\$ok\acute{}]\!]$
by (*simp add*:*R2T-def*)
also have ... $= (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle,dif_T(\$tr_T\acute{},\$tr_T)/\$tr_T,\$tr_T\acute{}]\!][\![true/\$ok\acute{}]\!]$
by *simp*
also have ... $= P[\![true/\$ok\acute{}]\!][\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle,dif_T(\$tr_T\acute{},\$tr_T)/\$tr_T,\$tr_T\acute{}]\!]$
apply (*subst-tac*)
apply (*subst usubst-upd-comm*, *simp*)
apply (*subst usubst-upd-comm2*, *simp*, *simp*)
by *subst-tac*

ultimately show *?thesis* **using** *R2T-def* **by** *pred-tac*
qed

lemma *R2T-ok-false*: $R2T(P^f) = (R2T(P))^f$
proof −
have $(R2T(P))^f = (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!][\![false/\$ok']\!]$
by (*simp add*:*R2T-def*)
also have ... $= (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!][\![false/\$ok']\!]$
by *simp*
also have ... $= P[\![false/\$ok']\!][\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!]$
apply (*subst-tac*)
apply (*subst usubst-upd-comm*, *simp*)
apply (*subst usubst-upd-comm2*, *simp*, *simp*)
by *subst-tac*

ultimately show *?thesis* **using** *R2T-def* **by** *pred-tac*
qed

lemma *R2T-wait-f*: $R2T((P)_f) = (R2T(P))_f$
proof −
have $(R2T(P))_f = (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!][\![false/\$wait]\!]$
by (*simp add*:*R2T-def*)
also have ... $= (P[\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!][\![false/\$wait]\!]$
by *simp*
also have ... $= P[\![false/\$wait]\!][\![\langle(\langle\rangle,\pi_2(last_u(\$tr_T)))_u\rangle, dif_T(\$tr_T{}',\$tr_T)/\$tr_T,\$tr_T{}']\!]$
apply (*subst-tac*)
apply (*subst usubst-upd-comm*, *simp*)
apply (*subst usubst-upd-comm2*, *simp*, *simp*)
by *subst-tac*

ultimately show *?thesis* **using** *R2T-def* **by** *pred-tac*
qed

lemma *R2T-wait-f-ok-t*: $R2T((P)^t{}_f) = (R2T(P))^t{}_f$
by (*simp add*: *R2T-ok-true R2T-wait-f*)

lemma *R2T-wait-f-ok-f*: $R2T((P)^f{}_f) = (R2T(P))^f{}_f$
by (*simp add*: *R2T-ok-false R2T-wait-f*)

lemma $R2T(\$wait_T) = \$wait_T$
apply (*simp add*:*R2T-def*)
by (*subst-tac*)

lemma *TR3-is-Conjunctive*:
Conjunctive(*TR3*)
unfolding *Conjunctive-def TR3-def* **by** *auto*

9.4 Coupling invariants

definition *CI0-def* [*upred-defs*]:
$CI0\ (P) \equiv P \wedge \$tr \leq_u \$tr' \wedge (\$tr'-\$tr =_u Flat_u(\$tr_T')-Flat_u(\$tr_T))$
$\wedge\ Flat_u(\$tr_T) \leq_u Flat_u(\$tr_T')$

definition *CI1-def* [*upred-defs*]:
$CI1\ (P) \equiv P \wedge (\$ref =_u \pi_2(last_u(\$tr_T))) \wedge (\$ref' =_u \pi_2(last_u(\$tr_T')))$

definition *CI2-def* [*upred-defs*]:
$CI2\ (P) \equiv P \wedge ((\neg\ \$wait' \wedge \neg P^f{}_f \wedge \$ok \wedge \$ok' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T) =_u \#_u(\$tr_T'))$

definition $CI2_m$*-def* [*upred-defs*]:
$CI2_m\ (P) \equiv ((\neg\ \$wait' \wedge \neg P^f{}_f \wedge \$ok \wedge \$ok' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T) =_u \#_u(\$tr_T'))$

definition *CI3-def* [*upred-defs*]:
$CI3\ (P) \equiv P \wedge \$wait_T =_u \$wait \wedge (\neg\ \$wait' \Rightarrow \neg\ \$wait_T')$

definition *CI4-def* [*upred-defs*]:
$CI4\ (P) \equiv P \wedge \$tr_C \leq_u \$tr_C' \wedge (\pi_1(head_u(\$tr_T'-front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C'-\$tr_C)$
$\wedge$
$\pi_1(last_u(\$tr_T)) \leq_u \pi_1(head_u(\$tr_T'-front_u(\$tr_T)))$

definition $CI4_m$*-def* [*upred-defs*]:
$CI4_m\ (P) \equiv P \wedge \$tr_C \leq_u \$tr_C' \wedge (\pi_1(head_u(\$tr_T'-front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C'-\$tr_C)$

definition *CI-def* [*upred-defs*]:
$CI\ (P) \equiv CI0(CI1(CI3(CI2(CI4(P)))))$

abbreviation *CI012* :: (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr*
⇒ (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr* **where**
$CI012(P) \equiv CI0(CI1(CI2(P)))$

abbreviation *CI0132* :: (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr*
⇒ (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr* **where**
$CI0132(P) \equiv CI0(CI1(CI3(CI2(P))))$

abbreviation *CI013* :: (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr*
⇒ (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr* **where**
$CI013(P) \equiv CI0(CI1(CI3(P)))$

abbreviation *CI0134* :: (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr*
⇒ (*bool*, ('*a*, '*b*) *alpha-st-scheme* × ('*a*, '*c*) *alpha-st-scheme*) *uexpr* **where**
$CI0134(P) \equiv CI0(CI1(CI3(CI4(P))))$

9.4.1 Properties of coupling invariants

9.4.1.1 Results on CI0

lemma *CI0-idempotent*: $CI0(CI0(P)) = CI0(P)$
by *pred-tac*

lemma *CI0-is-Conjunctive*: *Conjunctive*(*CI0*)
unfolding *CI0-def Conjunctive-def* **by** (*auto*)

lemma *CI0-distr-conj*: $CI0(P \wedge Q) = (CI0(P) \wedge CI0(Q))$
by *pred-tac*

lemma *CI0-distr-disj*: $CI0(P \vee Q) = (CI0(P) \vee CI0(Q))$
by *pred-tac*

lemma *CI0-conj*: $CI0(P \wedge Q) = (CI0(P) \wedge Q)$
by *pred-tac*

lemma *CI0-distr-cond*: $CI0(P \lhd b \rhd Q) = (CI0(P) \lhd b \rhd CI0(Q))$
by (*simp add*: *CI0-is-Conjunctive Conjunctive-distr-cond*)

lemma *CI0-CI1-commute*: $CI0(CI1(P)) = CI1(CI0(P))$
by *pred-tac*

lemma *CI0-CI3-commute*: $CI0(CI3(P)) = CI3(CI0(P))$
by *pred-tac*

9.4.1.2 Results on CI1

lemma *CI1-idempotent*: $CI1(CI1(P)) = CI1(P)$
by *pred-tac*

lemma *CI1-is-Conjunctive*: *Conjunctive*(*CI1*)
unfolding *CI1-def Conjunctive-def* **by** (*auto*)

lemma *CI1-distr-conj*: $CI1(P \wedge Q) = (CI1(P) \wedge CI1(Q))$
by *pred-tac*

lemma *CI1-distr-disj*: $CI1(P \vee Q) = (CI1(P) \vee CI1(Q))$
by *pred-tac*

lemma *CI1-conj*: $CI1(P \wedge Q) = (CI1(P) \wedge Q)$
by *pred-tac*

lemma *CI1-distr-cond*: $CI1(P \lhd b \rhd Q) = (CI1(P) \lhd b \rhd CI1(Q))$
by (*simp add*: *CI1-is-Conjunctive Conjunctive-distr-cond*)

9.4.1.3 Results on CI2

lemma *CI2-is-WeakConjunctive*: *WeakConjunctive*(*CI2*)
unfolding *CI2-def WeakConjunctive-def* **by** (*auto*)

lemma *CI2-distr-conj*: $CI2(P \wedge Q) = (CI2(P) \wedge CI2(Q))$
by *pred-tac*

lemma *CI2-ok′-and-wait-unrest*:

assumes $\$ok' \sharp P$ **and** $\$wait \sharp P$
shows $CI2(P) = (P)$
using *assms* **unfolding** *CI2-def*
by (*subst-tac*, *pred-tac*)

lemma *CI2-distr-cond*:
assumes $\$ok' \sharp b$ **and** $\$wait \sharp b$
shows $CI2(P \lhd b \rhd Q) = (CI2(P) \lhd b \rhd CI2(Q))$
using *assms*
by (*simp add*:*cond-def*, *pred-tac*)

lemma *CI2-eq-P-and-CI2*$_m$: $CI2(P) = (P \wedge CI2_m(P))$
by (*simp add*:*CI2*$_m$*-def CI2-def*)

definition *CI23-def*:
$CI23\ (P) \equiv P \wedge (\neg P^f{}_f \Rightarrow \$tr' =_u \$tr)$

lemma *CI2-conj*: $CI2(P \wedge Q) = (CI2(P) \wedge CI2(Q))$
apply (*simp add*:*CI2-def*)
by (*pred-tac*)

9.4.1.4 Results on CI3

lemma *CI3-idempotent*: $CI3(CI3(P)) = CI3(P)$
by *pred-tac*

lemma *CI3-distr-conj*: $CI3(P \wedge Q) = (CI3(P) \wedge CI3(Q))$
by *pred-tac*

lemma *CI3-distr-disj*: $CI3(P \vee Q) = (CI3(P) \vee CI3(Q))$
by *pred-tac*

lemma *CI3-is-Conjunctive*: *Conjunctive*(*CI3*)
unfolding *CI3-def Conjunctive-def* **by** (*auto*)

lemma *CI3-conj*: $CI3(P \wedge Q) = (CI3(P) \wedge Q)$
by *pred-tac*

lemma *CI3-distr-cond*: $CI3(P \lhd b \rhd Q) = (CI3(P) \lhd b \rhd CI3(Q))$
unfolding *cond-def* **by** *pred-tac*

lemma *Monotonic*(*CI3*)
apply (*simp add*:*CI3-def Monotonic-def*)
by *pred-tac*

lemma *CI1-CI3-commute*: $CI1(CI3(P)) = CI3(CI1(P))$
by *pred-tac*

9.4.1.5 Results on $\mathbf{R1_C}$

lemma $R1_C$-*TR4-commute*: $R1_C(TR4(P)) = TR4(R1_C(P))$
by (*simp add*:*TR4-def R1C-def*, *pred-tac*)

lemma $R1_C$-*TR3-commute*: $R1_C(TR3(P)) = TR3(R1_C(P))$
by (*simp add*:*TR3-def R1C-def*, *pred-tac*)

lemma $R1_C$-*TR2-commute*: $R1_C(TR2(P)) = TR2(R1_C(P))$
by (*simp add*:*TR2-def R1C-def*, *pred-tac*)

lemma $R1_C$-*TR1-commute*: $R1_C(TR1(P)) = TR1(R1_C(P))$
by (*simp add*:*TR2-def R1C-def*, *pred-tac*)

lemma $R1_C$-*TR0-commute*: $R1_C(TR0(P)) = TR0(R1_C(P))$
by (*simp add*:*TR0-def R1C-def*, *pred-tac*)

lemma $R1_C$-*TR0123-commute*: $R1_C(TR0123(P)) = TR0123(R1_C(P))$
by (*simp add*:*TR0-def TR1-def TR2-def TR3-def*, *pred-tac*)

lemma *CI-*$R1_C$*-eq-CI0132*: $CI(R1_C(P)) = CI(P)$
by (*simp add*:*CI-def R1C-def CI4-def utp-pred.inf.assoc*)

9.4.1.6 Results on $\mathbf{CI4_m}$

lemma $CI4_m$-*TR3-commute*: $CI4_m\ (TR3(P)) = TR3(\ CI4_m(P))$
by (*simp add*:$CI4_m$*-def TR3-def*, *pred-tac*)

lemma *CI4*$_m$*-R2T-commute*: *CI4*$_m$ (*R2T*(*P*)) = *R2T*(*CI4*$_m$(*P*))
apply (*simp add*:*CI4*$_m$*-def R2T-def*)
apply *subst-tac*
apply (*simp add*:*seq-minus-empty*)
by (*simp add*:*head*$_u$*-dif*$_T$)

9.4.1.7 Results on R1

lemma *TR3-CI4*$_m$*-R1-commute*: *TR3* (*CI4*$_m$ (*R1*(*P*))) = *R1*(*TR3* (*CI4*$_m$ (*P*)))
by (*simp add*:*CI4*$_m$*-def TR3-def R1-def*, *pred-tac*)

lemma *R2T-R1-commute*: *R2T*(*R1*(*P*)) = *R1*(*R2T*(*P*))
by (*simp add*:*R1-def R2T-def*, *subst-tac*)

lemma *R0T-R1T-R1-commute*: *R0T*(*R1T*(*R1*(*P*))) = *R1*(*R0T*(*R1T*(*P*)))
by (*simp add*:*R1-def R1T-def R0T-def ExpandsT-def*, *pred-tac*)

lemma *R0T-R1T-R2T-TR3-CI4*$_m$*-R1-commute*: *R0T* (*R1T* (*R2T* (*TR3* (*CI4*$_m$ (*R1*(*P*))))))
=
R1 (*R0T* (*R1T* (*R2T* (*TR3* (*CI4*$_m$ (*P*))))))
apply (*simp add*:*TR3-CI4*$_m$*-R1-commute*)
apply (*simp add*:*R2T-R1-commute*)
by (*simp add*:*R0T-R1T-R1-commute*)

lemma *CI0132-R1*: *CI0132* (*R1* (*P*)) = *CI0132* (*P*)
apply (*simp add*:*R1-def CI0-def CI1-def CI3-def CI2-def*)
by *pred-tac*

lemma *CI0-R1*: *CI0*(*R1*(*P*)) = *CI0*(*P*)
apply (*simp add*:*R1-def CI0-def*)
by (*simp add*:*utp-pred.inf.assoc utp-pred.inf.commute*)

9.4.1.8 Results on $\mathbf{R0_T}$ and CI

lemma *CI0-R0T-commute*: *CI0* (*R0T* (*P*)) = *R0T*(*CI0* (*P*))
by *pred-tac*

lemma *CI1-R0T-commute*: *CI1 (R0T (P)) = R0T(CI1 (P))*
by (*pred-tac*)

lemma *CI2-R0T-commute*: *CI2 (R0T (P)) = R0T(CI2 (P))*
by *pred-tac*

lemma *CI3-R0T-commute*: *CI3 (R0T (P)) = R0T(CI3 (P))*
by *pred-tac*

lemma *CI0132-R0T-commute*: *CI0132(R0T(P)) = R0T(CI0132(P))*
by (*simp add*: *CI2-R0T-commute CI3-R0T-commute CI1-R0T-commute CI0-R0T-commute*)

9.4.1.9 Results on $\mathbf{R1_T}$ and CI

lemma *CI0-R1T-commute*: *CI0 (R1T (P)) = R1T(CI0 (P))*
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *CI1-R1T-commute*: *CI1 (R1T (P)) = R1T(CI1 (P))*
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *CI2-R1T-commute*: *CI2 (R1T (P)) = R1T(CI2 (P))*
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *CI3-R1T-commute*: *CI3 (R1T (P)) = R1T(CI3 (P))*
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *CI0132-R1T-commute*: *CI0132(R1T(P)) = R1T(CI0132(P))*
by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

lemma *CI0132-R0T-R1T-commute*: *CI0132 (R0T (R1T(P))) = R0T (R1T(CI0132(P)))*
by (*simp add*:*CI0132-R0T-commute CI0132-R1T-commute*)

9.4.1.10 Results on R2T and CI

lemma $last_u$-dif_T:
assumes $[\![front_u(s) <_u t]\!]_e\ b$ **and** $[\![\#_u(s) >_u 0]\!]_e\ b$ **and** $[\![\#_u(t) >_u \#_u(s)]\!]_e\ b$
shows $[\![last_u(dif_T(t,s))]\!]_e\ b = [\![last_u(t)]\!]_e\ b$
using *assms* **unfolding** *upred-defs*

apply *transfer*
by (*simp add*:*last-difLP-t-s--eq--last-t-one*)

lemma *snd-last*$_u$*-dif*$_T$:
assumes $[\![front_u(s) <_u t]\!]_e\ b$ **and** $[\![\#_u(s) >_u 0]\!]_e\ b$ **and** $[\![\#_u(t) \geq_u \#_u(s)]\!]_e\ b$
shows $[\![\pi_2(last_u(dif_T(t,s)))]\!]_e\ b = [\![\pi_2(last_u(t))]\!]_e\ b$
using *assms* **unfolding** *upred-defs*
apply *transfer*
by (*simp add*:*snd-last-difLP-t-s--eq--snd-last-t-two*)

lemma *R0T-R1T-snd-last*$_u$*-dif*$_T$:
shows $R0T(R1T(x =_u \pi_2(last_u(dif_T(\$tr_T\acute{},\$tr_T)))))) = R0T(R1T(x =_u \pi_2(last_u(\$tr_T\acute{}))))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
apply (*simp only*:*utp-pred.inf.assoc*)
apply *bind-tac*
apply *transfer*
by (*metis snd-last*$_u$*-dif*$_T$ *uexpr-eq*)

lemma *R0T-R1T-CI1-R2T-commute*: $R0T(R1T(CI1(R2T(P)))) = R0T(R1T(R2T(CI1(P))))$
proof −
have $R0T(R1T(R2T(CI1(P)))) = R0T(R1T(R2T(P \wedge \$ref =_u \pi_2(last_u(\$tr_T)) \wedge \$ref\acute{}$
$=_u \pi_2(last_u(\$tr_T\acute{}))))))$
by (*simp add*:*CI1-def*)
also have ... $= R0T(R1T(R2T(P) \wedge \$ref =_u \pi_2(last_u(\langle(\langle\rangle :_u{'}a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle)))$
$\wedge\ \$ref\acute{} =_u \pi_2(last_u(dif_T(\$tr_T\acute{},\$tr_T)))))$
by (*simp add*:*R2T-distr-conj R2T-def*, *subst-tac*)
also have ... $= R0T(R1T(R2T(P) \wedge \$ref =_u \pi_2(last_u(\$tr_T))$
$\wedge\ \$ref\acute{} =_u \pi_2(last_u(dif_T(\$tr_T\acute{},\$tr_T)))))$
by *simp*
also have ... $= R0T(R1T(R2T(P) \wedge \$ref =_u \pi_2(last_u(\$tr_T))$
$\wedge\ \$ref\acute{} =_u \pi_2(last_u(\$tr_T\acute{})))))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
apply *bind-tac*
by (*metis snd-last*$_u$*-dif*$_T$ *uexpr-eq*)
also have ... $= R0T(R1T(CI1(R2T(P))))$
by (*simp add*:*CI1-def*)

finally show *?thesis* **..**
qed

lemma *CI3-R2T-commute*: $CI3(R2T(P)) = R2T(CI3(P))$
apply (*simp add*:*CI3-def R2T-def*)
by (*subst-tac*)

lemma *R0T-R1T-R2T-length-tr-eq*:
$R0T(R1T(R2T(\#_u(\$tr_T) =_u \#_u(\$tr_T\acute{}))))$
$=$
$R0T(R1T(\#_u(\$tr_T) =_u \#_u(\$tr_T\acute{})))$
proof −
have $R0T(R1T(R2T(\#_u(\$tr_T) =_u \#_u(\$tr_T\acute{}))))$
$=$
$R0T(R1T(1 =_u \#_u(dif_T(\$tr_T\acute{},\$tr_T))))$
apply (*simp add*:*R2T-def*)
apply *subst-tac*
by (*simp add*: *length-one*)
also have ... $= R0T(R1T(1 =_u \#_u(\$tr_T\acute{}) - \#_u(\$tr_T) + 1))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
apply *bind-tac*
by (*metis binding-length-dif*$_T$*-eq-length-t-minus-length-s-plus-one uexpr-eq*)
also have ... $= R0T(R1T(\#_u(\$tr_T) =_u \#_u(\$tr_T\acute{})))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
by *pred-tac*

finally show *?thesis* **.**
qed

lemma *CI2-R2T-commute*: $R0T(R1T(CI2(R2T(P)))) = R0T(R1T(R2T(CI2\ (P))))$
proof −
have $R0T(R1T(R2T(CI2\ (P))))$
$=$
$R0T(R1T(R2T(P \wedge ((\neg\ \$wait\acute{} \wedge \neg P^f{}_f \wedge \$ok \wedge \$ok\acute{} \wedge \$tr\acute{} =_u \$tr) \Rightarrow \#_u(\$tr_T) =_u \#_u(\$tr_T\acute{})))))$
by (*simp add*:*CI2-def*)

also have ... = $R0T(R1T(R2T(P \wedge ((\$wait' \vee P^f{}_f \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr) \vee \#_u(\$tr_T) =_u \#_u(\$tr_T'))))$

by (*simp add*:*impl-alt-def*)

also have ... = $R0T(R1T(R2T(P)$
$\wedge$
$(R2T(\$wait' \vee P^f{}_f \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr)$
$\vee$
$R2T(\#_u(\$tr_T) =_u \#_u(\$tr_T')))))$

by (*simp only*:*R2T-distr-conj R2T-distr-disj*)

also have ... = $R0T(R1T(R2T(P)$
$\wedge$
$((\$wait' \vee R2T(P^f{}_f) \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr)$
$\vee$
$R2T(\#_u(\$tr_T) =_u \#_u(\$tr_T')))))$

by (*simp add*:*CI2-def R2T-def*, *subst-tac*)

also have ... = $(R0T(R1T(R2T(P)))$
$\wedge$
$(R0T(R1T(\$wait' \vee R2T(P^f{}_f) \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr))$
$\vee$
$R0T(R1T(R2T(\#_u(\$tr_T) =_u \#_u(\$tr_T'))))$
$))$

by (*simp add*:*R0T-distr-conj R1T-distr-conj R0T-distr-disj R1T-distr-disj*)

also have ... = $(R0T(R1T(R2T(P)))$
$\wedge$
$(R0T(R1T(\$wait' \vee R2T(P^f{}_f) \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr))$
$\vee$
$R0T(R1T(\#_u(\$tr_T) =_u \#_u(\$tr_T')))$
$))$

by (*simp add*: *R0T-R1T-R2T-length-tr-eq*)

also have ... = $R0T(R1T(R2T(P)$
$\wedge$
$((\$wait' \vee R2T(P^f{}_f) \vee \neg \$ok \vee \neg \$ok' \vee \neg \$tr' =_u \$tr)$
$\vee$
$(\#_u(\$tr_T) =_u \#_u(\$tr_T'))$
$)))$

by (*simp add*:*R0T-distr-conj R1T-distr-conj R0T-distr-disj R1T-distr-disj*)

also have ... = $R0T(R1T(R2T(P)$

$\wedge$
$((\$wait' \vee (R2T(P))^f{}_f \vee \neg\ \$ok \vee \neg\ \$ok' \vee \neg\ \$tr' =_u \$tr)$
$\vee$
$(\#_u(\$tr_T) =_u \#_u(\$tr_T'))$
$)))$

by (*simp add*:*R2T-wait-f-ok-f*)
also have ... = *R0T*(*R1T*(*R2T*(*P*)
$\wedge$
$((\neg\ \$wait' \wedge \neg\ (R2T(P))^f{}_f \wedge \$ok \wedge \$ok' \wedge \$tr' =_u \$tr)$
$\Rightarrow$
$\#_u(\$tr_T) =_u \#_u(\$tr_T'))$
$))$
by (*simp add*:*impl-alt-def*)
also have ... = *R0T*(*R1T*(*CI2*(*R2T*(*P*))))
by (*simp add*:*CI2-def*)

finally show *?thesis* **..**
qed

lemma *FlatU-empty-trace*:
$Flat_u(\langle(\langle\rangle:_u$ *'a trace*,*s*:$_u$ *'a refusal*$)_u\rangle) = \langle\rangle$
by (*pred-tac*, *simp add*:*FlatLP-def*)

lemma *FlatU-empty-prefix*:
$Flat_u\ (dif_T(\$tr_T', \$tr_T)) \geq_u \langle\rangle = true$
by *pred-tac*

lemma *empty-prefix*: ‘$s \geq_u \langle\rangle$‘
by *pred-tac*

lemma *FlatU-minus-empty*:
$(\$tr' - \$tr =_u Flat_u\ (dif_T(\$tr_T', \$tr_T)) - \langle\rangle) = (\$tr' - \$tr =_u Flat_u\ (dif_T(\$tr_T', \$tr_T)))$
by *pred-tac*

lemma *R1T-implies-FlatU-prefix*:
$R1T(P) = R1T(P \wedge Flat_u\ (\$tr_T') \geq_u Flat_u\ (\$tr_T))$
apply (*simp only*:*R1T-def ExpandsT-def prefix-def prefixeq-def*)

apply *pred-tac*
apply (*insert fst-last-lt-fst-hd-diff-implies-FlatLP-prefix*)
by *auto*

lemma *CI0-R2T-commute*: $R0T(R1T(CI0\ (R2T\ (P)))) = R0T(R1T(R2T(CI0\ (P))))$
proof −
have $R0T(R1T(R2T(CI0\ (P))))$
$= R0T(R1T(R2T(P \wedge \$tr\acute{} \geq_u \$tr \wedge$
$\$tr\acute{} - \$tr =_u Flat_u\ (\$tr_T\acute{}) - Flat_u\ (\$tr_T) \wedge Flat_u\ (\$tr_T\acute{}) \geq_u Flat_u\ (\$tr_T)$
)))
by (*simp add*:*CI0-def*)
also have ... $= R0T(R1T(R2T(P) \wedge \$tr\acute{} \geq_u \$tr$
$\wedge\ \$tr\acute{} - \$tr$
$=_u Flat_u\ (dif_T(\$tr_T\acute{},\$tr_T)) - Flat_u\ (\langle(\langle\rangle{:}_u\ 'a\ trace,\pi_2(last_u(\$tr_T)))_u\rangle)$
$\wedge\ Flat_u\ (dif_T(\$tr_T\acute{},\$tr_T)) \geq_u Flat_u\ (\langle(\langle\rangle{:}_u\ 'a\ trace,\pi_2(last_u(\$tr_T)))_u\rangle)$
))
by (*simp add*:*R0T-def R1T-def R2T-def CI0-def ExpandsT-def*, *pred-tac*)
also have ... $= R0T(R1T(R2T(P) \wedge \$tr\acute{} \geq_u \$tr$
$\wedge\ \$tr\acute{} - \$tr$
$=_u Flat_u\ (dif_T(\$tr_T\acute{},\$tr_T)) - (\langle\rangle{:}_u\ 'a\ trace)$
$\wedge\ Flat_u\ (dif_T(\$tr_T\acute{},\$tr_T)) \geq_u \langle\rangle$
))
by (*simp add*:*FlatU-empty-trace*)
also have ... $= R0T(R1T(R2T(P) \wedge \$tr\acute{} \geq_u \$tr \wedge \$tr\acute{} - \$tr =_u Flat_u\ (dif_T(\$tr_T\acute{},\$tr_T))))$
by (*simp add*:*FlatU-empty-prefix FlatU-minus-empty*)
also have ... $= R0T(R1T(R2T(P) \wedge \$tr\acute{} \geq_u \$tr \wedge \$tr\acute{} - \$tr =_u Flat_u\ (\$tr_T\acute{}) - Flat_u\ (\$tr_T)))$
apply (*simp add*:*R0T-def R1T-def*)
apply (*bind-tac*)
by (*metis ExpandsT-def binding-FlatU-difT-FlatU-minus-FlatU uconjE uexpr-eq*)
also have ... $= R0T(R1T(R2T(P) \wedge \$tr\acute{} \geq_u \$tr \wedge \$tr\acute{} - \$tr =_u Flat_u\ (\$tr_T\acute{}) - Flat_u\ (\$tr_T)$
$\wedge\ Flat_u\ (\$tr_T\acute{}) \geq_u Flat_u\ (\$tr_T)))$
apply (*subst R1T-implies-FlatU-prefix*)
by (*simp only*:*utp-pred.inf.assoc*)
also have ... $= R0T(R1T(CI0(R2T(P))))$
by (*simp add*:*CI0-def*)

finally show *?thesis* **..**
qed

lemma *CI0132-R0T-R1T-R2T-commute*:
$CI0132(R0T(R1T(R2T(P)))) = R0T(R1T(R2T(CI0132(P))))$
proof −
have $CI0132(R0T(R1T(R2T(P)))) = CI0(CI1(CI3(R0T(R1T(CI2(R2T(P)))))))$
by (*simp add*:*CI2-R0T-commute CI2-R1T-commute*)
also have ... = $CI0(CI1(CI3(R0T(R1T(R2T(CI2(P)))))))$
by (*simp add*:*CI2-R2T-commute*)
also have ... = $CI0(CI1(R0T(R1T(CI3(R2T(CI2(P)))))))$
by (*simp add*:*CI3-R0T-commute CI3-R1T-commute*)
also have ... = $CI0(CI1(R0T(R1T(R2T(CI3(CI2(P)))))))$
by (*simp add*:*CI3-R2T-commute*)
also have ... = $CI0(R0T(R1T(CI1(R2T(CI3(CI2(P)))))))$
by (*simp add*:*CI1-R0T-commute CI1-R1T-commute*)
also have ... = $CI0(R0T(R1T(R2T(CI1(CI3(CI2(P)))))))$
by (*simp add*:*R0T-R1T-CI1-R2T-commute*)
also have ... = $R0T(R1T(CI0(R2T(CI1(CI3(CI2(P)))))))$
by (*simp add*:*CI0-R0T-commute CI0-R1T-commute*)
also have ... = $R0T(R1T(R2T(CI0132(P))))$
by (*simp add*:*CI0-R2T-commute*)

finally show *?thesis* **.**
qed

9.4.1.11 Results on TR3

lemma *TR3-distr-disj*: $TR3(P \vee Q) = (TR3(P) \vee TR3(Q))$
by *pred-tac*

lemma *TR3-distr-conj*: $TR3(P \wedge Q) = (TR3(P) \wedge TR3(Q))$

by *pred-tac*

lemma *TR3-conj*: $TR3(P \wedge Q) = (TR3(P) \wedge Q)$
by *pred-tac*

9.4.1.12 Results on TR3 and CI

lemma *TR3-CI0-commute*: $TR3(CI0(P)) = CI0(TR3(P))$
by *pred-tac*

lemma *TR3-CI1-commute*: $TR3(CI1(P)) = CI1(TR3(P))$
by *pred-tac*

lemma *TR3-CI2-commute*: $TR3(CI2(P)) = CI2(TR3(P))$
by *pred-tac*

lemma *TR3-CI3-commute*: $TR3(CI3(P)) = CI3(TR3(P))$
by *pred-tac*

lemma *TR3-CI0132-commute*:$TR3(\ CI0132(P)) = CI0132\ (TR3(P))$
by *pred-tac*

lemma *R2loc$_T$-refines-P*:
assumes $\$tr_T \sharp P$ **and** $\$tr_T{}' \sharp P$
shows $P \sqsubseteq R2loc_T(P)$
using *assms*
by *pred-tac*

Theorem 3 in utp2016

lemma
assumes $\$tr_T \sharp P$ **and** $\$tr_T{}' \sharp P$
shows $P \sqsubseteq TR(P)$
using *assms*
by *pred-tac*

end
theory *super-theory*

imports
super-theory-alpha
super-theory-healths
begin

9.5 Super-theory results

Coupling invariants with **TR**

lemma *CI4-R0T-TR2-eq-R0T-R1T-CI4m*:
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $CI4(\ R0T(\ TR2(P))) = R0T(R1T(CI4_m(P)))$
proof −
have $CI4(\ R0T(\ TR2(P))) =$
$(P \wedge \$tr_T{}' \geq_u front_u(\$tr_T) \wedge$
$\#_u(\$tr_T) >_u 0 \wedge \#_u(\$tr_T{}') \geq_u \#_u(\$tr_T) \wedge$
$\$tr_C{}' \geq_u \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C{}' - \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) \geq_u \pi_1(last_u(\$tr_T)))$
apply (*simp add:CI4-def R0T-def TR2-def*)
by *pred-tac*
also have ... =
$(P \wedge \$tr_T{}' >_u front_u(\$tr_T) \wedge$
$\#_u(\$tr_T) >_u 0 \wedge \#_u(\$tr_T{}') \geq_u \#_u(\$tr_T) \wedge$
$\$tr_C{}' \geq_u \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C{}' - \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) \geq_u \pi_1(last_u(\$tr_T)))$
by (*metis t-gte-front-eq-gt-front utp-pred.inf.assoc*)
also have ... = $R0T(P \wedge \$tr_T{}' >_u front_u(\$tr_T) \wedge$
$\$tr_C{}' \geq_u \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C{}' - \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) \geq_u \pi_1(last_u(\$tr_T)))$
apply (*simp add:R0T-def*)
by *pred-tac*
also have ... = $R0T(P \wedge \$tr_T{}' >_u front_u(\$tr_T) \wedge$
$\$tr_C{}' \geq_u \$tr_C \wedge$
$\pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C{}' - \$tr_C \wedge Expands_T(\$tr_T,\$tr_T{}'))$

apply (*simp add*:*ExpandsT-def*)
by *pred-tac*
also have ... = *R0T* (*R1T* (*P* ∧
$tr_C´ ≥_u $tr_C ∧
$\pi_1(head_u(\$tr_T´ - front_u(\$tr_T))) - \pi_1(last_u(\$tr_T)) =_u \$tr_C´ - \$tr_C$))
apply (*simp add*:*ExpandsT-def R1T-def*)
by *pred-tac*
also have ... = *R0T* (*R1T* (*CI4*$_m$ (*P*)))
by (*simp add*:*CI4*$_m$*-def*)

finally show *?thesis* **.**
qed

lemma *tots*: *R0T*(*R1T*(*x* =$_u$ #$_u$(*dif*$_T$($tr_T´,$tr_T))))
=
R0T(*R1T*(*x* =$_u$ (#$_u$($tr_T´)−#$_u$($tr_T) + *1*)))
apply (*simp-all add*:*R1T-def R0T-def ExpandsT-def*)
apply (*bind-tac*)
by (*metis* (*mono-tags*, *lifting*) *binding-length-dif*$_T$*-eq-length-t-minus-length-s-plus-one bop.rep-eq eq-upred-def*)

lemma *R0T-R1T-length-x*: *R0T*(*R1T*(*x* =$_u$ (#$_u$($tr_T´)−#$_u$($tr_T) + *x*)))
=
R0T(*R1T*(#$_u$($tr_T´) =$_u$ #$_u$($tr_T)))
apply (*simp-all add*:*R1T-def R0T-def ExpandsT-def*)
apply *bind-tac*
by (*meson lengtht-eq-lengths*)

R2T and TR3 commute under the conditions ensured by R0T and R1T

lemma *R0T-R1T-R2T-TR3-eq-R0T-R1T-TR3-R2T*:
fixes *P* :: ('*ϑ*,'*α*) *hrelation-st*
shows *R0T*(*R1T*(*R2T*(*TR3*(*P*)))) = *R0T*(*R1T*(*TR3*(*R2T*(*P*))))
proof −
have *R0T*(*R1T*(*R2T*(*TR3*(*P*)))) = *R0T*(*R1T*(*R2T* (*P* ∧ ($ok ∧ $wait$_T$ ⇒ #$_u$(tr_T$) =$_u$ #$_u$(tr_T$´) ∧ $wait$_T$´))))
by (*simp add*:*TR3-def*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧ *R2T*($ok ∧ $wait$_T$ ⇒ #$_u$(tr_T$) =$_u$ #$_u$(tr_T$´) ∧

$\$wait_T\acute{}$)))
by (*simp add*:*R2T-distr-conj*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧ ($\$ok$ ∧ $\$wait_T$ ⇒ *R2T*($\#_u(\$tr_T) =_u \#_u(\$tr_T\acute{})$)
∧ $\$wait_T\acute{}$)))
apply (*simp add*:*R2T-distr-disj R2T-distr-conj R2T-def*)
by (*subst-tac*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
($\#_u(\langle(\langle\rangle :: ('\vartheta\ trace, '\vartheta, '\alpha)\ huexpr\text{-}st, \pi_2(last_u(\$tr_T)))_u\rangle) =_u \#_u(dif_T(\$tr_T\acute{},\$tr_T))$)
∧ $\$wait_T\acute{}$)))
apply (*simp add*:*R2T-def*)
by (*subst-tac*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
(*1* $=_u \#_u(dif_T(\$tr_T\acute{},\$tr_T))$) ∧ $\$wait_T\acute{}$)))
by (*simp add*:*length-one*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
(*R0T*(*R1T*(*1* $=_u \#_u(dif_T(\$tr_T\acute{},\$tr_T))$)) ∧ $\$wait_T\acute{}$))
))
by (*smt R0T-R1T-commute R0T-distr-conj R0T-idempotent R0T-imp2 R1T-distr-conj R1T-idempotent R1T-imp2*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
(*R0T*(*R1T*(*1* $=_u (\#_u(\$tr_T\acute{})-\#_u(\$tr_T)$ + *1*))) ∧ $\$wait_T\acute{}$)))
by (*simp add*:*tots*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
(*R0T*(*R1T*($\#_u(\$tr_T\acute{})=_u\#_u(\$tr_T)$))) ∧ $\$wait_T\acute{}$)))
apply (*simp add*:*R1T-def R0T-def*)
apply *bind-tac*
by (*meson lengtht-eq-lengths*)
also have ... = *R0T*(*R1T*(*R2T* (*P*) ∧
($\$ok$ ∧ $\$wait_T$ ⇒
($\#_u(\$tr_T\acute{})=_u\#_u(\$tr_T)$ ∧ $\$wait_T\acute{}$))))
by (*smt R0T-R1T-commute R0T-distr-conj R0T-idempotent R0T-imp2 R1T-distr-conj R1T-idempotent R1T-imp2*)

also have ... = $R0T(R1T(TR3(R2T(P))))$
apply (*simp add*:*TR3-def*)
by (*simp add*:*utp-pred.eq-upred-sym*)

finally show *?thesis* **.**
qed

lemma $R2loc_T\text{-}R1_C\text{-}commute$: $R2loc_T(R1_C(P)) = R1_C(R2loc_T(P))$
apply (*simp add*:$R2loc_T$*-def R1C-def*)
by *subst-tac*

Corresponding to Lemma L.1.1.7 **Corresponding to: Lemma L.3.6.2**

lemma $CI\text{-}TR\text{-}eq\text{-}CI0132\text{-}R012T\text{-}TR3\text{-}CI4_m$:
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $CI(TR(P)) = CI0132\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (P))))))$
proof −
have $CI(TR(P)) = CI\ (TR0\ (TR1\ (TR2\ (TR3\ (TR4\ (R2loc_T\ (R1_C\ (P))))))))$
by (*simp add*:*CI-def TR-def*)
also have ... = $CI\ (TR0\ (TR1\ (TR2\ (TR3\ (TR4\ (R1_C\ (R2loc_T\ (P))))))))$
by (*simp add*:$R2loc_T\text{-}R1_C$*-commute*)
also have ... = $CI\ (R1_C\ (TR0\ (TR1\ (TR2\ (TR3\ (TR4\ (R2loc_T\ (P))))))))$
by (*simp add*:$R1_C$*-TR0123-commute* $R1_C$*-TR4-commute*)
also have ... = $CI\ (TR0\ (TR1\ (TR2\ (TR3\ (TR4\ (R2loc_T\ (P)))))))$
by (*simp add*:$CI\text{-}R1_C\text{-}eq\text{-}CI0132$)
also have ... = $CI\ (TR0\ (TR1\ (TR2\ (TR3\ (R2T\ (P))))))$
by (*metis TR0-TR4-R2loc-TR0-R2T TR0-inside*)
also have ... = $CI\ (R0T\ (TR2\ (TR3\ (R2T\ (P)))))$
by (*simp add*:*TR0-TR1-eq-R0T*)
also have ... = $CI0132\ (R0T\ (R1T\ (CI4_m\ (TR3\ (R2T\ (P))))))$
by (*simp add*:*CI-def CI4-R0T-TR2-eq-R0T-R1T-CI4m*)
also have ... = $CI0132\ (R0T\ (R1T\ (TR3\ (R2T\ (CI4_m\ (P))))))$
by (*simp add*:$CI4_m$*-R2T-commute* $CI4_m$*-TR3-commute*)
also have ... = $CI0132\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (P))))))$
by (*simp add*:*R0T-R1T-R2T-TR3-eq-R0T-R1T-TR3-R2T*)

finally show *?thesis* **.**

qed

Corresponding to: Lemma L.3.6.1

lemma *CI-TR-R1-eq-CI-TR*:
fixes $P :: ('\vartheta, '\alpha)$ *hrelation-st*
shows $CI(TR(R1(P))) = CI(TR(P))$
proof −
have $CI(TR(R1(P))) = CI0132\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (R1(P)))))))$
by (*simp add*:*CI-TR-eq-CI0132-R012T-TR3-CI4*$_m$)
also have ... $= CI0132\ (R1\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (P)))))))$
by (*simp add*:*R0T-R1T-R2T-TR3-CI4*$_m$*-R1-commute*)
also have ... $= CI0132\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (P))))))$
by (*simp add*:*CI0132-R1*)
also have ... $= CI(TR(P))$
by (*simp add*:*CI-TR-eq-CI0132-R012T-TR3-CI4*$_m$)

finally show *?thesis* **.**
qed

lemma *CI2-R3c-eq-R3c-and-CI2m*: $CI2(R3c(P)) = (R3c(P) \wedge CI2_m(P))$
apply (*simp add*:*CI2-def R3c-def CI2*$_m$*-def*)
apply (*subst-tac*)
by (*simp add*:*cond-def*)

lemma *not-ok-and-CI2*$_m$:$(\neg\ \$ok \wedge CI2_m(P)) = (\neg\ \$ok)$
apply (*simp add*:*CI2*$_m$*-def*)
by *pred-tac*

lemma *II-and-wait-and-CI2*$_m$:
$((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \wedge \$wait \wedge CI2_m\ (P))$
$=$
$((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \wedge \$wait)$
apply (*simp add*:*skip-r-def CI2*$_m$*-def*)
by *pred-tac*

lemma II_r*-wait-CI2*$_m$: $(II_r \wedge \$wait \wedge CI2_m\ (P)) = (II_r \wedge \$wait)$
proof −

have $(II_r \wedge \$wait \wedge CI2_m\ (P)) = (((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \vee (\neg\ \$ok \wedge \$tr \leq_u \$tr')) \wedge \$wait \wedge CI2_m\ (P))$
by (*simp add:skip-rea-def*)
also have ... $= (((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \wedge \$wait \wedge CI2_m\ (P)) \vee (\neg\ \$ok \wedge \$tr \leq_u \$tr' \wedge \$wait \wedge CI2_m\ (P)))$
by (*simp add: utp-pred.inf.assoc utp-pred.inf-sup-distrib2*)
also have ... $= (((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \wedge \$wait \wedge CI2_m\ (P)) \vee (\neg\ \$ok \wedge \$tr \leq_u \$tr' \wedge \$wait))$
by (*smt not-ok-and-CI2*$_m$ *utp-pred.inf.commute utp-pred.inf.left-commute*)
also have ... $= (((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \wedge \$wait) \vee (\neg\ \$ok \wedge \$tr \leq_u \$tr' \wedge \$wait))$
by (*simp add: II-and-wait-and-CI2*$_m$)
also have ... $= (((\$ref' =_u \$ref \wedge \$tr' =_u \$tr \wedge \$ok' \wedge \$wait' =_u \$wait) \vee (\neg\ \$ok \wedge \$tr \leq_u \$tr')) \wedge \$wait)$
by (*simp add: utp-pred.inf.assoc utp-pred.inf-sup-distrib2*)
also have ... $= (II_r \wedge \$wait)$
by (*simp add:skip-rea-def*)

finally show *?thesis* **.**
qed

lemma *CI2-R3c-eq-R3c-P-and-CI2*$_m$: $CI2(R3c(P)) = R3c(P \wedge CI2_m(P))$
proof −
have $CI2(R3c(P)) = (R3c(P) \wedge CI2_m\ (P))$
by (*simp add:CI2-R3c-eq-R3c-and-CI2m*)
also have ... $= ((II_r \triangleleft \$wait \triangleright P) \wedge CI2_m\ (P))$
by (*simp add:R3c-def*)
also have ... $= ((II_r \wedge CI2_m\ (P)) \triangleleft \$wait \triangleright (P \wedge CI2_m\ (P)))$
by (*simp add:cond-def utp-pred.inf.assoc utp-pred.inf-sup-distrib2*)
also have ... $= (II_r \triangleleft \$wait \triangleright (P \wedge CI2_m\ (P)))$
by (*smt II*$_r$*-wait-CI2*$_m$ *cond-def utp-pred.inf.commute utp-pred.inf.left-commute*)
also have ... $= R3c(P \wedge CI2_m(P))$
by (*simp add:R3c-def*)

finally show *?thesis* **.**
qed

lemma *CI0132-R3c-eq-CI013-R3c-and-CI2*$_m$: $CI0132(R3c(P)) = CI013(R3c(P \wedge CI2_m(P)))$
by (*simp add*:*CI2-R3c-eq-R3c-P-and-CI2*$_m$)

lemma *WeakConjunctive-simp*: $WeakConjunctive(HC) \Longrightarrow (\forall\ P.\ \exists\ Q.\ HC(P) = (P \wedge Q))$
by (*simp add*:*WeakConjunctive-def*)

lemma $Q \sqsubseteq P \longrightarrow CI2(Q) \sqsubseteq CI2(P)$
apply (*simp add*:*CI2-def*)
by *pred-tac*

lemma $\forall\ Q\ P.\ `P \Rightarrow Q` \Longrightarrow `CI2(P) \Rightarrow CI2(Q)`$
apply (*simp add*:*CI2-def*)
by *pred-tac*

lemma *Monotonic*(*CI2*)
apply (*simp add*:*Monotonic-def*)
apply (*simp add*:*CI2-def*)
by *rel-tac*

lemma *WeakConjunctive*(*CI2*)
apply (*simp add*:*WeakConjunctive-def CI2-def Monotonic-def*)
by (*metis CI2*$_m$*-def CI2-distr-conj CI2-eq-P-and-CI2*$_m$ *eq-upred-def utp-pred.inf.absorb2 utp-pred.inf-le1*)

lemma *Conjunctive*(*CI2*)
apply (*simp add*:*Conjunctive-def*)
apply (*simp add*:*CI2-def*)
oops

lemma $CI013(R3c(P)) = (CI013(II_r) \lhd \$wait_T \rhd CI013(P))$
apply (*simp add*:*R3c-def cond-conj-distr CI0-distr-conj CI0-conj CI1-distr-conj CI1-conj CI3-distr-conj CI3-conj*)
oops

lemma *CI012-distr-cond*:
assumes $\$ok' \sharp b$ **and** $\$wait \sharp b$
shows $CI012(P \lhd b \rhd Q) = (CI012(P) \lhd b \rhd CI012(Q))$
using *assms*

by (*simp add*:*CI0-distr-cond CI1-distr-cond CI2-distr-cond*)

lemma *CI013-distr-cond*:
$CI013(P \lhd b \rhd Q) = (CI013(P) \lhd b \rhd CI013(Q))$
by (*simp add*:*CI0-distr-cond CI1-distr-cond CI3-distr-cond*)

lemma *bool-value-eq*: $(x \wedge x =_u y) = (y \wedge x =_u y)$
by *pred-tac*

lemma *bool-neg-value-eq*: $(\neg\ x \wedge x =_u y) = (\neg\ y \wedge x =_u y)$
by *pred-tac*

lemma *CI013-wait-II*$_r$: $CI013(\$wait \wedge II_r) = CI013(\$wait_T \wedge II_r)$
apply (*simp add*:*skip-rea-def CI3-def utp-pred.inf.assoc*)
by (*smt bool-value-eq utp-pred.inf.assoc utp-pred.inf.left-commute*)

lemma *CI013-not-wait-P*: $CI013(\neg\ \$wait \wedge P) = CI013(\neg\ \$wait_T \wedge P)$
apply (*simp add*:*skip-rea-def CI3-def utp-pred.inf.assoc*)
by (*smt bool-neg-value-eq utp-pred.inf.assoc utp-pred.inf.left-commute*)

lemma *CI013-conj*: $CI013(P \wedge Q) = (CI013(P) \wedge Q)$
by (*simp add*: *CI0-is-Conjunctive CI1-conj CI3-conj Conjunctive-conj*)

lemma *CI013-R3c*: $CI013(R3c(P)) = (CI013(II_r) \lhd \$wait_T \rhd CI013(P))$
proof −
have $CI013(R3c(P)) = (CI013(II_r) \lhd \$wait \rhd CI013(P))$
by (*simp add*:*R3c-def CI013-distr-cond*)
also have ... $= (CI013(II_r \wedge \$wait) \vee CI013(P \wedge \neg\ \$wait))$
by (*simp add*:*cond-def CI013-conj utp-pred.inf.commute*)
also have ... $= (CI013(\$wait \wedge II_r) \vee CI013(\neg\ \$wait \wedge P))$
by (*simp add*:*utp-pred.inf.commute*)
also have ... $= (CI013(\$wait_T \wedge II_r) \vee CI013(\neg\ \$wait_T \wedge P))$
by (*simp add*:*CI013-wait-II*$_r$ *CI013-not-wait-P*)
also have ... $= ((CI013(II_r) \wedge \$wait_T) \vee (CI013(P) \wedge \neg\ \$wait_T))$
by (*simp add*:*CI013-conj utp-pred.inf.commute*)
also have ... $= (CI013(II_r) \lhd \$wait_T \rhd CI013(P))$
by (*simp add*:*cond-def utp-pred.inf.commute*)

finally show *?thesis* **.**
qed

lemma *not-wait$_T$-and-TR3*: $(\neg\ \$wait_T \wedge\ TR3(P)) = (\neg\ \$wait_T \wedge P)$
apply (*simp add*:*TR3-def*)
by (*smt impl-alt-def utp-pred.compl-inf utp-pred.inf.commute utp-pred.inf.left-commute utp-pred.inf-sup*

lemma *TR3-CI013-R3*: $TR3(CI013(R3c(P))) = (TR3(CI013(II_r)) \lhd \$wait_T \rhd CI013(P))$
proof −
have $TR3(CI013(R3c(P))) = (TR3(CI013(II_r) \lhd \$wait_T \rhd CI013(P)))$
by (*simp add*:*CI013-R3c*)
also have ... $= (TR3(CI013(II_r)) \lhd \$wait_T \rhd TR3(CI013(P)))$
apply (*simp add*:*cond-def TR3-def*)
by (*simp add*: *utp-pred.inf.assoc utp-pred.inf-sup-distrib2*)
also have ... $= (TR3(CI013(II_r)) \lhd \$wait_T \rhd CI013(P))$
by (*simp add*:*cond-def not-wait$_T$-and-TR3*)

finally show *?thesis* **.**
qed

lemma *R0T-distr-cond*: $R0T(P \lhd b \rhd Q) = (R0T(P) \lhd b \rhd R0T(Q))$
by (*simp add*:*cond-def R0T-distr-disj R0T-conj utp-pred.inf.commute*)

lemma *R1T-distr-cond*: $R1T(P \lhd b \rhd Q) = (R1T(P) \lhd b \rhd R1T(Q))$
by (*simp add*:*cond-def R1T-distr-disj R1T-conj utp-pred.inf.commute*)

lemma *CI0-tr-eq-tr'*:$CI0(\$tr\acute{}=_u\$tr) = CI0(Flat_u(\$tr_T\acute{})=_u Flat_u(\$tr_T))$
apply (*simp add*:*CI0-def*)
apply *pred-tac*
using *prefixeq-diff-minus* **apply** *fastforce*
using *prefixeq-diff-minus* **by** *fastforce*

lemma *CI1-ref-eq-ref'*:$CI1(\$ref\acute{}=_u\$ref) = CI1(\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T\acute{})))$
apply (*simp add*:*CI1-def*)
by *pred-tac*

lemma *CI013-II$_r$*: $CI013(II_r) = CI013(\neg\ \$ok$
$\vee$
$(\$ok' \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T)$
$\wedge\ \$wait'=_u \$wait_T))$

proof −

have $CI013(II_r) = CI013((\neg\ \$ok \wedge \$tr \leq_u \$tr') \vee (\$ok' \wedge \$ref'=_u \$ref \wedge \$tr'=_u \$tr \wedge \$wait'=_u \$wait))$

apply (*simp add:skip-rea-def utp-pred.inf.commute*)

by (*simp add*: *utp-pred.inf.commute utp-pred.inf.left-commute utp-pred.sup.commute*)

also have ... $= CI013((\neg\ \$ok \wedge CI0(\$tr \leq_u \$tr'))$
$\vee$
$(\$ok' \wedge \$ref'=_u \$ref \wedge CI0(\$tr'=_u \$tr) \wedge \$wait'=_u \$wait))$

by (*simp add*: *CI0-CI1-commute CI0-CI3-commute CI0-distr-conj CI0-distr-disj CI0-idempotent*)

also have ... $= CI013((\neg\ \$ok \wedge CI0(true))$
$\vee$
$(\$ok' \wedge \$ref'=_u \$ref \wedge CI0(Flat_u(\$tr_T')=_u Flat_u(\$tr_T)) \wedge \$wait'=_u \$wait))$

by (*smt CI0-def CI0-tr-eq-tr' utp-pred.inf.left-idem utp-pred.inf-top.left-neutral*)

also have ... $= CI013((\neg\ \$ok\ \wedge CI0(true))$
$\vee$
$(\$ok' \wedge \$ref'=_u \$ref \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T) \wedge \$wait'=_u \$wait))$

by (*simp add:CI0-CI1-commute CI0-CI3-commute CI0-distr-conj CI0-distr-disj CI0-idempotent CI0-conj*)

also have ... $= CI013((\neg\ \$ok\ \wedge true)$
$\vee$
$(\$ok' \wedge \$ref'=_u \$ref \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T) \wedge \$wait'=_u \$wait))$

by (*smt CI0-CI1-commute CI0-CI3-commute CI0-distr-conj CI0-distr-disj CI0-idempotent*)

also have ... $= CI013((\neg\ \$ok)$
$\vee$
$(\$ok' \wedge \$ref'=_u \$ref \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T) \wedge \$wait'=_u \$wait))$

by *pred-tac*

also have ... $= CI013((\neg\ \$ok)$
$\vee$
$(\$ok' \wedge CI1(\$ref'=_u \$ref) \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T) \wedge \$wait'=_u \$wait))$

by (*simp add*: *CI1-CI3-commute CI1-distr-conj CI1-distr-disj CI1-idempotent*)

also have ... $= CI013((\neg\ \$ok)$
$\vee$
$(\$ok' \wedge CI1(\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T'))) \wedge Flat_u(\$tr_T')=_u Flat_u(\$tr_T)$

$\wedge$ $\$wait' =_u \$wait$))

apply (*simp add*:*CI1-def*)

by *pred-tac*

also have ... = *CI013*((¬ $\$ok$)

$\vee$

($\$ok' \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T') =_u Flat_u(\$tr_T)$

$\wedge$ $\$wait' =_u \$wait$))

by (*simp add*: *CI1-CI3-commute CI1-distr-conj CI1-distr-disj CI1-idempotent*)

also have ... = *CI013*((¬ $\$ok$)

$\vee$

($\$ok' \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T') =_u Flat_u(\$tr_T)$

$\wedge$ *CI3*($\$wait' =_u \$wait$)))

by (*simp add*: *CI3-distr-conj CI3-distr-disj CI3-idempotent*)

also have ... = *CI013*((¬ $\$ok$)

$\vee$

($\$ok' \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T') =_u Flat_u(\$tr_T)$

$\wedge$ *CI3*($\$wait' =_u \$wait_T$)))

apply (*simp add*:*CI3-def*)

by *pred-tac*

also have ... = *CI013*(¬ $\$ok$

$\vee$

($\$ok' \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T') =_u Flat_u(\$tr_T)$

$\wedge$ $\$wait' =_u \$wait_T$))

by (*simp add*: *CI3-distr-conj CI3-distr-disj CI3-idempotent*)

finally show *?thesis* **.**

qed

lemma (¬ $\$ok \vee (\$ok' \wedge \$wait' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))$

=

(¬ $\$ok \vee (\$ok \wedge \$ok' \wedge \$wait' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))$

by (*simp add*: *utp-pred.sup-inf-distrib1*)

lemma $R0T(R1T(\#_u(\$tr_T) =_u \#_u(\$tr_T{}') \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T)$
$=_u Flat_u(\$tr_T{}')))$

$=$

$R0T(R1T(\$tr_T =_u \$tr_T{}'))$

apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)

apply *pred-tac*

by (*smt FlatLP-eq-length-eq-front-prefix-and-snd-last-eq butlast-prefix-eq-butlast length-greater-0-conv*)

Corresponding to: Lemma L.4.1.9

lemma *TR3-CI013-wait-and-II*$_r$: $TR3(CI013(\$wait_T \wedge II_r))$

$=$

$CI013(\$wait_T \wedge (\neg\ \$ok$

$\vee$

$(\$ok{}' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T{}')$

$\wedge\ \$wait_T{}' =_u \$wait_T$

$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T{}'))))$

proof −

have $TR3(CI013(\$wait_T \wedge II_r)) = TR3(CI013(II_r) \wedge \$wait_T)$

by (*simp add*:*CI013-conj utp-pred.inf.commute*)

also have ... $= TR3(\$wait_T \wedge CI013(\neg\ \$ok \vee (\$ok{}' \wedge \$wait{}' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T{}'))))$

by (*simp add*:*CI013-II*$_r$ *utp-pred.inf.commute utp-pred.inf.left-commute eq-upred-sym*)

also have ... $= TR3(\$wait_T \wedge CI013(\neg\ \$ok \vee (\$ok \wedge \$ok{}' \wedge \$wait{}' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T{}'))))$

by (*simp add*: *utp-pred.sup-inf-distrib1*)

also have ... $= TR3(CI013(\$wait_T \wedge$

$(\neg\ \$ok \vee (\$ok \wedge \$ok{}' \wedge \$wait{}' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T{}')))))$

by (*smt CI013-conj utp-pred.inf.commute*)

also have ... $= TR3(CI013((\$wait_T \wedge \neg\ \$ok) \vee (\$ok \wedge \$ok{}' \wedge \$wait_T \wedge \$wait{}' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T{}')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T{}'))))$

by (*simp add*: *utp-pred.inf.left-commute utp-pred.inf-sup-distrib1*)

also have ... = ($TR3(CI013(\$wait_T \wedge \neg \$ok))$

$\vee$

$TR3(CI013(\$ok \wedge \$ok' \wedge \$wait_T \wedge \$wait' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$

by (*simp add*:*CI3-distr-disj CI1-distr-disj CI0-distr-disj TR3-distr-disj*)

also have ... = ($CI013(TR3(\$wait_T \wedge \neg \$ok))$

$\vee$

$TR3(CI013(\$ok \wedge \$ok' \wedge \$wait_T \wedge \$wait' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$

by (*simp add*: *TR3-CI0-commute TR3-CI1-commute TR3-CI3-commute*)

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$

$\vee$

$TR3(CI013(\$ok \wedge \$ok' \wedge \$wait_T \wedge \$wait' =_u \$wait_T$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$

apply (*simp add*:*TR3-def*)

by *pred-tac*

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$

$\vee$

$TR3(\$ok \wedge \$ok' \wedge \$wait_T \wedge CI013(\$wait' =_u \$wait_T)$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))$

by (*smt CI013-conj utp-pred.inf.left-commute*)

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$

$\vee$

$TR3(\$ok' \wedge \$ok \wedge \$wait_T \wedge CI013(\$wait' =_u \$wait_T)$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))$

by (*simp add*: *utp-pred.inf.left-commute*)

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$

$\vee$

$(\$ok' \wedge TR3(\$ok \wedge \$wait_T) \wedge CI013(\$wait' =_u \$wait_T)$

$\wedge$

$\pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))$

by (*smt TR3-conj utp-pred.inf.assoc utp-pred.inf.left-commute*)

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$
$\vee$
$(\$ok' \wedge \$ok \wedge \$wait_T \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T'$
$\wedge\ CI013(\$wait' =_u \$wait_T)$
$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u$
$Flat_u(\$tr_T')))$

apply (*simp add:TR3-def*)

by *pred-tac*

also have ... = ($CI013(\$wait_T \wedge \neg \$ok)$
$\vee$
$(\$ok' \wedge \$ok \wedge \$wait_T \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T'$
$\wedge\ CI013(\$wait_T' =_u \$wait_T)$
$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u$
$Flat_u(\$tr_T')))$

apply (*simp add:CI3-def*)

by *pred-tac*

also have ... = $CI013((\$wait_T \wedge \neg \$ok)$
$\vee$
$(\$ok' \wedge \$ok \wedge \$wait_T \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T'$
$\wedge\ \$wait_T' =_u \$wait_T$
$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u$
$Flat_u(\$tr_T')))$

apply (*simp add:CI3-distr-disj CI1-distr-disj CI0-distr-disj*)

by *pred-tac*

also have ... = $CI013((\$wait_T \wedge \neg \$ok)$
$\vee$
$(\$ok' \wedge \$ok \wedge \$wait_T \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T')$
$\wedge\ \$wait_T' =_u \$wait_T$
$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u$
$Flat_u(\$tr_T')))$

by *pred-tac*

also have ... = $CI013(\$wait_T \wedge ((\neg \$ok)$
$\vee$
$(\$ok' \wedge \$ok \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T')$
$\wedge\ \$wait_T' =_u \$wait_T$
$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u$
$Flat_u(\$tr_T'))))$

by *pred-tac*
also have ... = $CI013(\$wait_T \wedge ((\neg\ \$ok)$
$\vee$
$(\$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T' =_u \$wait_T \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$
by *pred-tac*
also have ... = $CI013(\$wait_T \wedge (\neg\ \$ok$
$\vee$
$(\$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T' =_u \$wait_T \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$
by *pred-tac*

finally show *?thesis* **.**
qed

Corresponding to: Lemma L.4.1.10

lemma *R0T-R1T-TR3-CI01r-wait-and-II$_r$*: $R0T(R1T(TR3(CI013(\$wait_T \wedge II_r)))) = R0T(R1T(CI013 \wedge II_T)))$
proof −
have $R0T(R1T(TR3(CI013(\$wait_T \wedge II_r))))$
$=$
$R0T(R1T(CI013(\$wait_T \wedge (\neg\ \$ok$
$\vee$
$(\$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T' =_u \$wait_T \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))))$
by (*simp add*:*TR3-CI013-wait-and-II$_r$*)
also have ... = $R0T(R1T(CI013((\$wait_T \wedge \neg\ \$ok)$
$\vee$
$(\$wait_T \wedge \$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T') \wedge \$wait_T' =_u \$wait_T \wedge \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))))$
by (*simp add*: *utp-pred.inf-sup-distrib1*)
also have ... = $R0T(R1T(CI013(\$wait_T \wedge \neg\ \$ok)$

$\vee$

$CI013(\$wait_T \wedge \$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T')$

$\wedge\ \$wait_T' =_u \$wait_T$

$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T'))))$

by (*smt CI013-conj utp-pred.inf-sup-distrib1*)

also have ... = $(R0T(R1T(CI013(\$wait_T \wedge \neg\ \$ok)))$

$\vee$

$R0T(R1T(CI013(\$wait_T \wedge \$ok' \wedge \#_u(\$tr_T) =_u \#_u(\$tr_T')$

$\wedge\ \$wait_T' =_u \$wait_T$

$\wedge\ \pi_2(last_u(\$tr_T)) =_u \pi_2(last_u(\$tr_T')) \wedge Flat_u(\$tr_T) =_u Flat_u(\$tr_T')))))$

by (*simp add*:*R0T-distr-disj R1T-distr-disj*)

also have ... = $(R0T(R1T(CI013(\$wait_T \wedge \neg\ \$ok)))$

$\vee$

$R0T(R1T(CI013(\$wait_T \wedge \$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T'))))$

apply (*simp add*:*R0T-def R1T-def ExpandsT-def CI0-def CI1-def CI3-def*)

apply *pred-tac*

by (*smt FlatLP-eq-length-eq-front-prefix-and-snd-last-eq butlast-prefix-eq-butlast length-greater-0-conv*)

also have ... = $(R0T(R1T(CI013(R1T(\$wait_T \wedge \neg\ \$ok))))$

$\vee$

$R0T(R1T(CI013(\$wait_T \wedge \$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T'))))$

by (*simp add*:*R1T-def ExpandsT-def CI013-conj*)

also have ... = $(R0T(R1T(CI013(\$wait_T \wedge R1T(\neg\ \$ok))))$

$\vee$

$R0T(R1T(CI013(\$wait_T \wedge \$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T'))))$

by (*smt R1T-conj utp-pred.inf.commute*)

also have ... = $R0T(R1T(CI013(\$wait_T \wedge R1T(\neg\ \$ok))$

$\vee$

$CI013(\$wait_T \wedge \$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T')))$

by (*simp add*:*R0T-distr-disj R1T-distr-disj*)

also have ... = $R0T(R1T(CI013((\$wait_T \wedge R1T(\neg\ \$ok))$

$\vee$

$(\$wait_T \wedge \$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T'))))$

by (*smt CI013-conj utp-pred.inf-sup-distrib1*)

also have ... = $R0T(R1T(CI013(\$wait_T \wedge (R1T(\neg\ \$ok)$

$\vee$

$(\$ok' \wedge \$wait_T' =_u \$wait_T \wedge \$tr_T =_u \$tr_T')))))$

by (*simp add*: *utp-pred.inf-sup-distrib1*)
also have ... = $R0T(R1T(CI013(\$wait_T \wedge II_T)))$
by (*simp add*:II_T*-def*)

finally show *?thesis* **.**
qed

Corresponding to: Lemma L.4.1.7

lemma *R0T-R1T-TR3-CI013-R3-eq-R0T-R1T-CI013-R3T*:
$R0T(R1T(TR3(CI013(R3c(P))))) = R0T(R1T(CI013(R3T(P))))$
proof −
have $R0T(R1T(TR3(CI013(R3c(P))))) = R0T(R1T(TR3(CI013(II_r)) \lhd \$wait_T \rhd CI013(P)))$
by (*simp add*:*TR3-CI013-R3*)
also have ... = $R0T(R1T((\$wait_T \wedge TR3(CI013(II_r))) \lhd \$wait_T \rhd CI013(P)))$
by (*simp add*:*cond-def*)
also have ... = $R0T(R1T((TR3(CI013(\$wait_T \wedge II_r))) \lhd \$wait_T \rhd CI013(P)))$
apply (*simp add*: *TR3-conj CI013-conj utp-pred.inf.commute*)
by (*simp add*: *TR3-def utp-pred.inf.commute utp-pred.inf.left-commute*)
also have ... = $(R0T(R1T(TR3(CI013(\$wait_T \wedge II_r)))) \lhd \$wait_T \rhd R0T(R1T(CI013(P))))$
by (*simp add*:*R1T-distr-cond R0T-distr-cond*)
also have ... = $(R0T(R1T(CI013(\$wait_T \wedge II_T))) \lhd \$wait_T \rhd R0T(R1T(CI013(P))))$
by (*simp add*:*R0T-R1T-TR3-CI01r-wait-and-*II_r)
also have ... = $(R0T(R1T(CI013((\$wait_T \wedge II_T) \lhd \$wait_T \rhd P))))$
apply (*simp add*: *CI013-conj R1T-distr-cond R0T-distr-cond*)
by (*simp add*: *CI013-conj CI013-distr-cond R0T-distr-cond R1T-distr-cond*)
also have ... = $(R0T(R1T(CI013(II_T \lhd \$wait_T \rhd P))))$
by (*simp add*:*cond-def*)
also have ... = $(R0T(R1T(CI013(R3T(P)))))$
by (*simp add*:*R3T-def*)

finally show *?thesis* **.**
qed

Corresponding to: Lemma L.4.6.10

lemma *R0T-R1T-TR3-CI0132-R3*: $R0T(R1T(TR3(CI0132(R3c(P))))) = R0T(R1T(CI013(R3T(P \wedge CI2_m(P)))))$
by (*simp add*:*R0T-R1T-TR3-CI013-R3-eq-R0T-R1T-CI013-R3T CI0132-R3c-eq-CI013-R3c-and-*$CI2_m$)

lemma *CI0132-CI4*$_m$*-commute*: $CI0132(CI4_m(P)) = CI4_m(CI0132(P))$
apply (*simp add*:*CI0-def CI1-def CI2-def CI3-def CI4*$_m$*-def*)
by *pred-tac*

lemma $CI2(R3c(P)) = R3c(P \wedge CI2_m(P))$
oops

lemma $TR3(CI0132(R3c(P))) = CI012(R3T(P \wedge CI2_m(P)))$
apply (*simp add*:*CI0132-R3c-eq-CI013-R3c-and-CI2*$_m$ *CI2-R3c-eq-R3c-P-and-CI2*$_m$)
oops

lemma *length-single-here*: $length\ [e] = 1$
by *simp*

lemma *R0T-R1T-R2T-eq-R0T-R1T-R2T-R0T-R1T*:
$R0T\ (R1T\ (R2T(P))) = R0T\ (R1T\ (R2T\ (R0T\ (R1T\ (P)))))$
proof −
have $R0T\ (R1T\ (R2T\ (R0T\ (R1T\ (P)))))$
=
$R0T(R1T(R2T(P \wedge \#_u(\$tr_T) >_u 0 \wedge \#_u(\$tr_T) \leq_u \#_u(\$tr_T{}') \wedge$
$front_u(\$tr_T) <_u \$tr_T{}' \wedge \pi_1(last_u(\$tr_T)) \leq_u \pi_1(head_u(\$tr_T{}' - front_u(\$tr_T)))$
$)))$
by (*simp add*:*R0T-def R1T-def ExpandsT-def*, *pred-tac*)
also have ... $= R0T(R1T(R2T(P) \wedge$
$R2T(\#_u(\$tr_T) >_u 0) \wedge$
$R2T(\#_u(\$tr_T) \leq_u \#_u(\$tr_T{}')) \wedge$
$R2T(front_u(\$tr_T) <_u \$tr_T{}') \wedge$
$R2T(\pi_1(last_u(\$tr_T)) \leq_u \pi_1(head_u(\$tr_T{}' - front_u(\$tr_T))))$
$))$
by (*simp add*:*R2T-distr-conj*)
also have ... $= R0T(R1T(R2T(P) \wedge$
$\#_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle) >_u 0 \wedge$
$\#_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle) \leq_u \#_u(dif_T(\$tr_T{}', \$tr_T)) \wedge$
$front_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle) <_u dif_T(\$tr_T{}', \$tr_T) \wedge$
$\pi_1(last_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle)) \leq_u \pi_1(head_u(dif_T(\$tr_T{}', \$tr_T) -$
$front_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle))))$
$))$

by (*simp add*:*R2T-def*, *subst-tac*)
also have ... = $R0T(R1T(R2T(P) \wedge$
$1 \leq_u \#_u(dif_T(\$tr_T{}', \$tr_T)) \wedge$
$front_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle) <_u dif_T(\$tr_T{}', \$tr_T) \wedge$
$\pi_1(last_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle)) \leq_u \pi_1(head_u(dif_T(\$tr_T{}', \$tr_T) -$
$front_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle))))$
))
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
by *pred-tac*
also have ... = $R0T(R1T(R2T(P) \wedge$
$1 \leq_u \#_u(dif_T(\$tr_T{}', \$tr_T)) \wedge$
$\pi_1(last_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle)) \leq_u \pi_1(head_u(dif_T(\$tr_T{}', \$tr_T) -$
$front_u(\langle(\langle\rangle :_u 'a\ trace, \pi_2(last_u(\$tr_T)))_u\rangle))))$
))
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
apply *pred-tac*
by (*simp add*: *Suc-le-eq prefix-bot.bot.not-eq-extremum*)
also have ... = $R0T(R1T(R2T(P) \wedge$
$1 \leq_u \#_u(dif_T(\$tr_T{}', \$tr_T))))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
by *pred-tac*
also have ... = $R0T(R1T(R2T(P) \wedge$
$1 \leq_u (\#_u(\$tr_T{}') - \#_u(\$tr_T) + 1)))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
apply *pred-tac*
by (*simp add*:*length-difLP-eq-length-dif-plus-one*)
also have ... = $R0T(R1T(R2T(P)))$
apply (*simp add*:*R0T-def R1T-def ExpandsT-def*)
by *pred-tac*

finally show *?thesis* **..**
qed

lemma $CI4_m$-*R0T-commute*: $CI4_m(R0T(P)) = R0T(CI4_m(P))$
by *pred-tac*

lemma $CI4_m$-*R1T-commute*: $CI4_m(R1T(P)) = R1T(CI4_m(P))$

by (*simp add*:*R1T-def ExpandsT-def*, *pred-tac*)

Corresponding to: Lemma L.4.6.9

lemma *CI-TR-R3c*:
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $CI(TR(R3c(P))) = R0T\ (R1T\ (R2T\ (CI4_m(CI013(R3T(P \wedge CI2_m(P)))))))$
proof −
have $CI(TR(R3c(P))) = CI0132\ (R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (R3c(P)))))))$
by (*simp add*:*CI-TR-eq-CI0132-R012T-TR3-CI4*$_m$)
also have ... $= R0T\ (R1T\ (R2T\ (CI0132\ (TR3\ (CI4_m\ (R3c(P)))))))$
by (*simp add*:*CI0132-R0T-R1T-R2T-commute*)
also have ... $= R0T\ (R1T\ (R2T\ (TR3\ (CI0132\ (CI4_m\ (R3c(P)))))))$
by (*simp add*:*TR3-CI0132-commute*)
also have ... $= R0T\ (R1T\ (R2T\ (TR3\ (CI4_m\ (CI0132\ (R3c(P)))))))$
by (*simp add*:*CI0132-CI4*$_m$*-commute*)
also have ... $= R0T\ (R1T\ (R2T\ (R0T\ (R1T\ (TR3\ (CI4_m\ (CI0132\ (R3c(P))))))))))$
by (*subst R0T-R1T-R2T-eq-R0T-R1T-R2T-R0T-R1T*, *auto*)
also have ... $= R0T\ (R1T\ (R2T\ (CI4_m\ (R0T\ (R1T\ (TR3\ (CI0132\ (R3c(P))))))))))$
by (*simp add*: *CI4*$_m$*-R0T-commute CI4*$_m$*-R1T-commute CI4*$_m$*-TR3-commute*)
also have ... $= R0T\ (R1T\ (R2T\ (CI4_m\ (R0T(R1T(CI013(R3T(P \wedge CI2_m(P)))))))))$
by (*simp add*:*R0T-R1T-TR3-CI0132-R3*)
also have ... $= R0T\ (R1T\ (R2T\ (R0T(R1T(CI4_m\ (CI013(R3T(P \wedge CI2_m(P)))))))))$
by (*simp add*: *CI4*$_m$*-R0T-commute CI4*$_m$*-R1T-commute CI4*$_m$*-TR3-commute*)
also have ... $= R0T\ (R1T\ (R2T\ (CI4_m(CI013(R3T(P \wedge CI2_m(P)))))))$
by (*subst R0T-R1T-R2T-eq-R0T-R1T-R2T-R0T-R1T*, *auto*)

finally show *?thesis* **.**
qed

Corresponding to: Lemma L.4.6.8

lemma *CI-TR-R3c-with-CI0134*:
fixes $P :: ('\vartheta,'\alpha)$ *hrelation-st*
shows $CI(TR(R3c(P))) = R0T(R1T(R2T(CI0134(R3T(P \wedge CI2_m(P))))))$
proof −
have $CI(TR(R3c(P))) = R0T\ (R1T\ (R2T\ (CI4_m(CI013(R3T(P \wedge CI2_m(P)))))))$
by (*simp add*: *CI-TR-R3c*)
also have ... $= R0T\ (R1T\ (R2T\ (R0T\ (R1T\ (CI4_m(CI013(R3T(P \wedge CI2_m(P)))))))))$
by (*subst R0T-R1T-R2T-eq-R0T-R1T-R2T-R0T-R1T*, *auto*)

also have ... = *R0T* (*R1T* (*R2T* (*R0T* (*R1T* (*CI0134*(*R3T*(*P* ∧ *CI2*$_m$(*P*))))))))
apply (*simp add*:*CI4-def CI4*$_m$*-def R0T-def R1T-def ExpandsT-def*)
by (*simp add*: *CI013-conj utp-pred.inf-commute utp-pred.inf-left-commute*)
also have ... = *R0T* (*R1T* (*R2T* (*CI0134*(*R3T*(*P* ∧ *CI2*$_m$(*P*))))))
by (*subst R0T-R1T-R2T-eq-R0T-R1T-R2T-R0T-R1T*, *auto*)

finally show *?thesis* **.**
qed

Corresponding to: Lemma L.3.6.3

lemma *CI-TR-RH*:
fixes *P* :: ('ϑ,'α) *hrelation-st*
shows *CI*(*TR*(*RH*(*P*))) = *R0T*(*R1T*(*R2T*(*CI0134*(*R3T*(*R2*(*P*) ∧ *CI2*$_m$(*R2*(*P*)))))))
proof −
have *CI*(*TR*(*RH*(*P*))) = *CI*(*TR*(*R1*(*R2*(*R3c*(*P*)))))
by (*simp add*:*RH-def*)
also have ... = *CI*(*TR*(*R2*(*R3c*(*P*))))
by (*simp add*:*CI-TR-R1-eq-CI-TR*)
also have ... = *CI*(*TR*(*R3c*(*R2*(*P*))))
by (*simp add*:*R2-R3c-commute*)
also have ... = *R0T*(*R1T*(*R2T*(*CI0134*(*R3T*(*R2*(*P*) ∧ *CI2*$_m$(*R2*(*P*)))))))
by (*simp add*:*CI-TR-R3c-with-CI0134*)

finally show *?thesis* **.**
qed

lemma *R2s-design*: *R2s*(*P* ⊢ *Q*) = ((¬ *R2s*(¬*P*)) ⊢ *R2s*(*Q*))
apply (*simp add*:*design-def R2s-def*)
by (*subst-tac*)

lemma *CI2*$_m$*-R1-eq-CI2*$_m$:
shows *CI2*$_m$(*R1*(*P*)) = *CI2*$_m$(*P*)
proof −
have *CI2*$_m$(*R1*(*P*)) = ((¬ $*wait*´ ∧ ¬(*R1*(*P*))$^f_f$ ∧ $*ok* ∧ $*ok*´ ∧ $*tr*´ =$_u$ $*tr*) ⇒ #$_u$($*tr*$_T$) =$_u$ #$_u$($*tr*$_T$´))
by (*simp add*:*CI2*$_m$*-def*)
also have ... = ((¬ $*wait*´ ∧ ¬(*P* ∧ $*tr* ≤$_u$ $*tr*´)$^f_f$ ∧ $*ok* ∧ $*ok*´ ∧ $*tr*´ =$_u$ $*tr*) ⇒

$\#_u(\$tr_T) =_u \#_u(\$tr_T{}')$)
by (*simp add*:*R1-def*)
also have ... = $((\neg\ \$wait' \wedge (\neg {P^f}_f \vee \neg \$tr \leq_u \$tr') \wedge \$ok \wedge \$ok' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T) =_u \#_u(\$tr_T{}'))$
by *subst-tac*
also have ... = $((\neg\ \$wait' \wedge \neg {P^f}_f \wedge \$ok \wedge \$ok' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T) =_u \#_u(\$tr_T{}'))$
by *pred-tac*
also have ... = $CI2_m(P)$
by *pred-tac*

finally show *?thesis* **.**
qed

lemma *R1-CI2*$_m$*-R2s*: $R1(CI2_m(R2s(P))) = R1(R2s(CI2_m(P)))$
apply (*simp add*:*CI2*$_m$*-def R2s-def R1-def*)
apply *subst-tac*
apply *pred-tac*
apply (*metis alpha-d.update-convs*(*1*) *alpha-rp.surjective alpha-rp.update-convs*(*1*) *alpha-rp.update-convs*
using *prefixeq-diff-minus* **apply** *fastforce*
by (*metis alpha-d.update-convs*(*1*) *alpha-rp.surjective alpha-rp.update-convs*(*1*) *alpha-rp.update-convs*(*2*))

lemma *R1-CI2*$_m$*-R2-eq-R2-CI2*$_m$:
shows $R1(CI2_m(R2(P))) = R2(CI2_m(P))$
proof −
have $R1(CI2_m(R2(P))) = R1(CI2_m(R2s(P)))$
by (*simp add*:*R2-def CI2*$_m$*-R1-eq-CI2*$_m$)
also have ... = $R1(R2s(CI2_m(P)))$
by (*simp add*:*R1-CI2*$_m$*-R2s*)
also have ... = $R2(CI2_m(P))$
by (*simp add*:*R2-def*)

finally show *?thesis* **.**
qed

lemma *CI2*$_m$*-design*: $CI2_m(P \vdash Q) = CI2_m(\neg\ P)$
apply (*simp add*:*CI2*$_m$*-def design-def*)
apply *subst-tac*

by *pred-tac*

lemma *CI2$_m$-and-design*:
assumes $\$ok' \sharp P$ $\$wait \sharp P$
shows $((P \vdash Q) \wedge CI2_m(\neg P)) = (P \vdash (Q \wedge ((\neg\ \$wait' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T') =_u \#_u(\$tr_T))))$
using *assms*
apply (*simp add*:*design-def CI2$_m$-def*)
apply (*subst-tac*)
by *pred-tac*

Corresponding to: Lemma L.3.6.7

lemma
fixes $P :: ('\vartheta, '\alpha)$ *hrelation-st*
assumes $\$ok \sharp P$ $\$ok' \sharp P$ $\$ok \sharp Q$ $\$ok' \sharp Q$ $\$wait \sharp P$
shows $CI(TR(RH(P \vdash Q)))$
$=$
$R0T(R1T(R2T(CI0134(R3T(R2((P \vdash (Q \wedge ((\neg\ \$wait' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T') =_u \#_u(\$tr_T))))$
$))))))$
proof −
have $CI(TR(RH(P \vdash Q)))$
$=$
$R0T(R1T(R2T(CI0134(R3T(R2(P \vdash Q) \wedge CI2_m(R2(P \vdash Q)))))))$
by (*simp add*:*CI-TR-RH*)
also have ... $= R0T(R1T(R2T(CI0134(R3T(R2(P \vdash Q) \wedge R2(CI2_m(P \vdash Q)))))))$
apply (*simp add*:*R2-def*)
by (*metis* (*no-types*, *lifting*) *CI2$_m$-R1-eq-CI2$_m$ R1-CI2$_m$-R2s R1-conj R1-extend-conj R1-idem*)
also have ... $= R0T(R1T(R2T(CI0134(R3T(R2((P \vdash Q) \wedge CI2_m(P \vdash Q)))))))$
apply (*simp add*:*R2-def*)
by (*metis R2-conj R2-def*)
also have ... $= R0T(R1T(R2T(CI0134(R3T(R2((P \vdash Q) \wedge CI2_m(\neg P)))))))$
by (*simp add*:*CI2$_m$-design*)
also have ... $= R0T(R1T(R2T(CI0134(R3T(R2($
$(P \vdash (Q \wedge ((\neg\ \$wait' \wedge \$tr' =_u \$tr) \Rightarrow \#_u(\$tr_T') =_u \#_u(\$tr_T))))$
$))))))$
using *assms*

by (*simp add*:*CI2*$_m$-*and-design*)

finally show *?thesis* **.**
qed

end