An encoding of Interval Temporal Logic in Isabelle/HOL

Antonio Cau Ben Moszkowski David Smallwood

October 20, 2018

Abstract

These Isabelle theories introduce the semantics and syntax of Interval Temporal Logic (ITL). The ITL proof system, as introduced in [3], has been encoded and its soundness has been checked. An extensive library of ITL theorems, taken from [5], has been checked. The time reversal operator [4] has been defined and a collection of theorems has been provided.

Furthermore the new ITL operators first $\triangleright$ and last $\triangleleft$ (introduced using time reversal) have been defined together with an extensive library of theorems. These were used to introduce the Runtime Verification monitor language RV [6] together with the algebraic properties of this language.

We also provide an algebraic characterisation of ITL based on [2] and link it with the work on Kleene Algebras [1].

Contents

theory *Interval*
imports
Main
begin

1 Intervals

An interval is a sequence of elements of a particular type. Intervals are similar to list in Isabelle/HOL, the difference is that intervals have instead of nil a single element at the end. So an interval of length zero is a single element whereas for Isabelle's list we have that an empty list is nil (no element present).

The usual operations on intervals are defined: length (*intlen*), *prefix*, *suffix*, *sub*, *nth*, *intfirst*, *intlast*, *intapp* and *intrev*.

In order to define the semantics of the ITL chopstar we introduce *index-sequence* which is a sequence of chop (fuse) points. This sequence is again of type interval but the elements are natural numbers. Two functions *shift* and *shiftm* are introduced that are used to add (shift) and subtract a natural number of each element in the sequence of chop (fuse) points.

1.1 Definitions

datatype *'a interval* =
  St *'a* ($\lceil$-$\rceil$)
| *Cons* *'a* *'a interval* (**infixr** $\odot$ *65*)
for
  map: *map*
  rel: *interval-all2*
  pred: *interval-all*

type-synonym *index* = *nat interval*

syntax
  — interval Enumeration
  -interval :: *args* => *'a interval* ($\langle$(-)$\rangle$)

translations
  $\langle x, xs\rangle == x\odot\langle xs\rangle$
  $\langle x\rangle == \lceil x\rceil$

primrec (*nonexhaustive*) *intlen* :: *'a interval* ⇒ *nat* **where**
  intlen (*St x*) = *0*
| *intlen* ($x\odot xs$) = *1+* (*intlen xs*)

primrec (*nonexhaustive*) *nth* :: *'a interval* => *nat* => *'a* **where**
  nth (*St x*) *n* = *x*
| *nth* (*Cons x xs*) *n* = (*case n of 0* ⇒ *x* | *Suc k* ⇒ *nth xs k*)

primrec *prefix*:: *nat* ⇒ *'a interval* ⇒ *'a interval* **where**
  prefix n (*St x*) = (*St x*)
| *prefix n* (*Cons x xs*) = (*case n of 0* ⇒ (*St x*) | *Suc m* ⇒ (*Cons x* (*prefix m xs*)))

primrec *suffix*:: *nat* ⇒ *'a interval* ⇒ *'a interval* **where**
  suffix n (*St x*) = (*St x*)
| *suffix n* (*Cons x xs*) = (*case n of 0* ⇒ (*Cons x xs*) | *Suc m* ⇒ *suffix m xs*)

definition *sub*:: *nat* ⇒ *nat* ⇒ *'a interval* ⇒ *'a interval*
where
  sub n k xs = (*if k<n then prefix 0* (*suffix n xs*)
    else prefix (*k−n*) (*suffix n xs*)
)

primrec *intfirst* :: *'a interval* ⇒ *'a* **where**
  intfirst (*St x*) = *x*
| *intfirst* (*Cons x -*) = *x*

primrec *intlast* :: $'a$ *interval* $\Rightarrow$ $'a$ **where**
intlast $(St\ x) \quad = x$
| *intlast* $(Cons\ \text{-}\ xs) = intlast\ xs$

primrec *intapp* :: $'a$ *interval* $\Rightarrow$ $'a$ *interval* $\Rightarrow$ $'a$ *interval* (**infixr** $\ominus$ *65*) **where**
intapp-St: $(St\ x) \ominus ys = x \odot ys$ |
intapp-Cons: $(x \odot xs) \ominus ys = x \odot (xs \ominus ys)$

primrec *intrev* :: $'a$ *interval* $\Rightarrow$ $'a$ *interval* **where**
intrev $(St\ x) = (St\ x)$
| *intrev* $(Cons\ x\ xs) = (intrev\ xs) \ominus (St\ x)$

definition *index-sequence* :: *nat* $\Rightarrow$ *index* $\Rightarrow$ *bool* **where**
index-sequence $x\ idx \equiv (nth\ idx\ 0 = x) \wedge (\forall\ n.\ n<intlen\ idx \longrightarrow nth\ idx\ n < nth\ idx\ (Suc\ n))$

definition *shift* :: *nat* $\Rightarrow$ *nat* $\Rightarrow$ *nat* **where**
shift $k = (\lambda\ x.\ x+k)$

definition *shiftm* :: *nat* $\Rightarrow$ *nat* $\Rightarrow$ *nat* **where**
shiftm $k = (\lambda\ x.\ (if\ k>x\ then\ 0\ else\ (x-k)))$

1.2 Lemmas

Basic lemmas are introduced for each of the above operations on intervals.

1.2.1 Interval Length

lemma *interval-intlen-gr-zero* [*simp*]:
intlen $xs \geq 0$
by *auto*

lemma *interval-intlen-st* :
intlen $(St\ x) = 0$
by *simp*

lemma *interval-intlen-cons* [*simp*]:
$(intlen\ (x \odot xs)) = (intlen\ xs) + 1$
by *simp*

lemma *interval-intlen-cons-1* :
intlen $l > 0 \longleftrightarrow (\exists\ x\ ls.\ l = x \odot ls)$
by (*induct l*) *simp-all*

lemma *interval-intlen-map*:
intlen $(map\ f\ xs) = intlen\ xs$
by (*induct xs*) *simp-all*

1.2.2 nth

lemma *interval-nth-zero* [*simp*]:
  $nth\ (x \odot xs)\ 0\ = x$
by *simp*

lemma *interval-nth-Suc* [*simp*]:
  $nth\ (x \odot xs)\ (Suc\ n) = nth\ xs\ n$
by *auto*

lemma *interval-nth-last*:
  $nth\ (x \odot xs)\ (intlen\ (x \odot xs)) = nth\ xs\ (intlen\ xs)$
by *simp*

lemma *interval-nth-cons*:
 assumes $0 < i \wedge i < 1 + intlen(xs)$
 shows $nth(x \odot xs)\ i = nth\ xs\ (i - 1) \wedge$
   $nth(x \odot xs)\ (i+1) = nth\ xs\ ((i - 1)+1)$
by (*metis One-nat-def Suc-leI add.commute assms interval-nth-Suc le-add-diff-inverse2 plus-1-eq-Suc*)

lemma *interval-nth-zero-intfirst*:
  $nth\ xs\ 0 = intfirst\ xs$
by (*induct xs*) *simp-all*

lemma *interval-nth-intlen-intlast*:
  $nth\ xs\ (intlen\ xs) = intlast\ xs$
by (*induct xs*) *simp-all*

lemma *interval-st-intlen* :
  $(xs = (St\ x)) \longleftrightarrow intlen\ xs = 0 \wedge nth\ xs\ 0 = x$
by (*induct xs*) *simp-all*

lemma *interval-eq-nth-eq* :
  $(xs = ys) = (intlen\ xs = intlen\ ys \wedge (\forall\ i \leq intlen\ xs.\ nth\ xs\ i = nth\ ys\ i))$
apply (*induct xs arbitrary*: *ys*)
apply (*metis interval-st-intlen le-numeral-extra*(*3*))
apply (*case-tac ys*, *simp*)
by *fastforce*

lemma *interval-nth-map* :
  $nth\ (map\ f\ xs)\ i = f\ (nth\ xs\ i)$
apply (*induct xs arbitrary*: *i*, *simp*)
apply (*case-tac i*, *simp*, *simp*)
done

1.2.3 index sequence

lemma *interval-idx-less*:
  assumes *iseq*: *index-sequence x idx*
  shows $(n < intlen\ idx \wedge n+k < intlen\ idx) \longrightarrow nth\ idx\ n < nth\ idx\ (Suc(n+k))$
apply (*induct k*)

using *index-sequence-def iseq* **apply** *auto*[*1*]
using *index-sequence-def iseq* **by** *auto*

lemma *interval-idx-less-last* :
assumes *index-sequence x idx*
shows $(i<intlen\ idx \wedge i+(intlen\ idx - (i+1))< intlen\ idx)$
$\longrightarrow nth\ idx\ i < nth\ idx\ (Suc(i+(intlen\ idx\ -(i+1))))$
using *assms interval-idx-less* **by** *blast*

lemma *interval-idx-less-last-1*:
assumes *index-sequence x idx*
shows $i<intlen\ idx \longrightarrow nth\ idx\ i < nth\ idx\ (intlen\ idx)$
using *assms interval-idx-less-last* **by** *auto*

lemma *interval-idx-greater-first*:
assumes *index-sequence x idx*
shows $(i>0 \wedge i\leq intlen\ idx) \longrightarrow x < nth\ idx\ i$
apply (*induct i*, *simp*)
using *assms*
by (*metis One-nat-def Suc-le-lessD add-Suc index-sequence-def interval-idx-less less-le-trans plus-1-eq-Suc*)

lemma *interval-idx-cons*:
index-sequence 0 $(x\odot ls) =$
$(x=0 \wedge x<nth\ ls\ 0 \wedge index\text{-}sequence\ (nth\ ls\ 0)\ ls)$
apply (*simp add*: *index-sequence-def*)
using *less-Suc-eq-0-disj* **by** *auto*

lemma *interval-idx-shift-mono*:
mono (*shift k*)
by (*simp add*: *Interval.shift-def mono-def*)

lemma *interval-idx-expand*:
$index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs) \wedge 0\leq i \wedge i< (intlen\ l)$
$\Longrightarrow 0\leq (nth\ l\ i) \wedge (nth\ l\ i) \leq\ (nth\ l\ (i+1)) \wedge (nth\ l\ (i+1)) \leq (intlen\ xs)$
apply (*simp add*: *index-sequence-def*)
apply (*induct l*, *simp*)
by (*metis Suc-lessI eq-imp-le index-sequence-def interval-idx-less-last-1 less-imp-le-nat*)

lemma *interval-idx-shift-idx* [*simp*]:
$(\ index\text{-}sequence\ (x+k)\ (map\ (shift\ k)\ \ idx)) = (index\text{-}sequence\ x\ idx)$
by (*simp add*: *Interval.shift-def index-sequence-def interval-intlen-map interval-nth-map*)

lemma *interval-idx-shiftm* :
$(index\text{-}sequence\ k\ (lsk) \wedge ls = map\ (shiftm\ k)\ lsk) \Longrightarrow$
$index\text{-}sequence\ 0\ (ls) \wedge (intlen\ ls) = (intlen\ lsk)$
by (*simp add*: *interval-eq-nth-eq index-sequence-def shift-def shiftm-def interval-nth-map*)
(*smt Suc-leI diff-less-mono index-sequence-def interval-idx-greater-first interval-intlen-map le-less-trans less-Suc-eq-0-disj not-less order.asym*)

lemma *interval-lsk-ls* :
  (*index-sequence k* (*lsk*) ∧ *lsk* = *map* (*shift k*) *ls* ∧ *index-sequence 0* (*ls*)) =
   (*index-sequence k* (*lsk*) ∧ *ls* = *map* (*shiftm k*) *lsk* ∧ *index-sequence 0* (*ls*))
apply (*simp add*: *interval-eq-nth-eq index-sequence-def shift-def shiftm-def interval-nth-map*)
apply *rule*
apply (*metis* (*no-types*, *lifting*) *add-diff-cancel-right′ interval-intlen-map not-add-less2*)
by (*metis* (*no-types*, *lifting*) *Suc-eq-plus1 add.commute add-cancel-right-left add-diff-inverse-nat ex-least-nat-less interval-intlen-map le-SucE le-zero-eq not-less-zero order-refl*)

lemma *interval-idx-link-shiftm*:
  (*index-sequence k* (*lsk*) ∧ *ls* = *map* (*shiftm k*) *lsk*) =
   (*index-sequence k* (*lsk*) ∧ *ls* = *map* (*shiftm k*) *lsk* ∧
   index-sequence 0 (*ls*) ∧ (*intlen ls*) =(*intlen lsk*))
using *interval-idx-shiftm* **by** *blast*

lemma *interval-idx-link*:
  (*lsk* = *map* (*shift k*) *ls* ∧ *index-sequence 0* (*ls*)) =
   (*lsk* = *map* (*shift k*) *ls* ∧ *index-sequence k* (*lsk*) ∧ *index-sequence 0* (*ls*)∧
   (*intlen ls*) =(*intlen lsk*))
by (*metis Interval.shift-def add-diff-cancel-left′ diff-diff-cancel diff-is-0-eq′ interval-idx-shift-idx interval-idx-shift-mono interval-intlen-map le-numeral-extra*(*3*) *mono-def*)

lemma *interval-idx-bound-0* :
 assumes *index-sequence 0 ls* ∧ *Interval.nth ls* (*intlen ls*) = *intlen* (*suffix k xs*)
 shows ((*i*≤*intlen ls*) ⟶ ((*nth ls* (*i*)) ≤ (*intlen* (*suffix k xs*))))
using *assms*
by (*metis add.commute add-eq-if eq-iff interval-idx-less le-add-diff-inverse2 le-neq-implies-less lessI less-imp-le-nat*)

lemma *interval-idx-bound-1*:
 (*index-sequence 0* (*ls*) ∧ (*nth* (*ls*) (*intlen* (*ls*))) = (*intlen* (*suffix k xs*))) ⟷
   (*index-sequence 0* (*ls*) ∧ (*nth* (*ls*) (*intlen* (*ls*))) = (*intlen* (*suffix k xs*)) ∧
   (∀ *i*. (*i*≤*intlen ls*) ⟶ ((*nth ls* (*i*)) ≤ (*intlen* (*suffix k xs*)))))
using *interval-idx-bound-0* **by** *blast*

1.2.4 prefix, suffix and sub

lemma *interval-prefix-state* [*simp*]:
   prefix m (*St x*) = (*St x*)
by *simp*

lemma *interval-prefix-suc* [*simp*]:
   prefix (*Suc m*) (*x*⊙*xs*) = *x* ⊙ (*prefix m xs*)
by *auto*

lemma *interval-prefix-zero* [*simp*]:
   prefix 0 (*x*⊙*xs*) = *St x*
by *auto*

lemma *interval-prefix-zero-intfirst* [*simp*]:

prefix 0 xs = *St* (*intfirst xs*)
by (*induct xs*) *simp-all*

lemma *interval-intfirst-prefix* [*simp*]:
$i \leq$*intlen xs* $\Longrightarrow$ *intfirst* (*prefix i xs*) = *intfirst xs*
by (*induct xs arbitrary*: *i*, *auto*) (*case-tac i*, *auto*)

lemma *interval-prefix-intlen* [*simp*]:
(*prefix* (*intlen xs*) *xs*) = *xs*
by (*induct xs*) *simp-all*

lemma *interval-prefix-intlen-gr-1* [*simp*]:
(*prefix* ((*intlen xs*)+*i*) *xs*) = *xs*
by (*induct xs*) *simp-all*

lemma *interval-intlen-prefix-cons* [*simp*]:
intlen(*prefix* (*Suc i*) (*x*$\odot$*xs*)) = *1* + *intlen*(*prefix i xs*)
using *interval-intlen-cons* **by** *auto*

lemma *interval-prefix-length* :
intlen (*prefix i xs*) = (*if i*$\leq$ *intlen xs then i else intlen xs*)
by (*induct xs arbitrary*: *i*, *simp*) (*case-tac i*, *auto*)

lemma *interval-prefix-length-good* [*simp*]:
assumes *i*$\leq$ *intlen xs*
shows (*intlen* (*prefix i xs*)) = *i*
using *assms* **by** (*simp add*: *interval-prefix-length*)

lemma *interval-prefix-length-bad* [*simp*] :
assumes *i* > *intlen xs*
shows *intlen* (*prefix i xs*) = *intlen xs*
using *assms* **by** (*simp add*: *interval-prefix-length*)

lemma *interval-pref-intlen-bound* :
assumes *i* $\leq$ (*intlen xs*)
shows *intlen* (*prefix i xs*) $\leq$ *intlen xs*
using *assms* **by** (*induct xs*, *simp*) (*metis interval-prefix-length*)

lemma *interval-suffix-length*:
intlen (*suffix i xs*) = (*if i*$\leq$ *intlen xs then* (*intlen xs*)$-$*i else 0*)
by (*induct xs arbitrary*: *i*, *simp*) (*case-tac i*, *auto*)

lemma *interval-suffix-length-good* [*simp*]:
assumes *i*$\leq$ *intlen xs*
shows *intlen* (*suffix i xs*) = (*intlen xs*)$-$*i*
using *assms* **by** (*simp add*: *interval-suffix-length*)

lemma *interval-suffix-length-bad* [*simp*]:
assumes *i*> *intlen xs*
shows *intlen* (*suffix i xs*) = *0*

using *assms* **by** (*simp add*: *interval-suffix-length*)

lemma *interval-nth-prefix* [*simp*]:
$i \le intlen\ xs \wedge k \le i \Longrightarrow nth\ (prefix\ i\ xs)\ k = nth\ xs\ k$
apply (*induct xs arbitrary*: *i k*, *auto*)
apply (*case-tac i*, *auto*)
apply (*case-tac k*, *auto*)
done

lemma *interval-nth-suffix* [*simp*]:
$i \le intlen\ xs \wedge k \le intlen\ xs - i \Longrightarrow nth\ (suffix\ i\ xs)\ k = nth\ xs\ (i+k)$
by (*induct xs arbitrary*: *i k*, *auto*) (*case-tac i*, *auto*)

lemma *interval-suffix-prefix-help-1*:
assumes $ia+i \le intlen\ xs \wedge k \le ia$
shows $nth\ (prefix\ ia\ (suffix\ i\ xs))\ k = nth\ (suffix\ i\ (prefix\ (ia+i)\ xs))\ k$
proof −
have *1*: $nth\ (prefix\ ia\ (suffix\ i\ xs))\ k = nth\ (suffix\ i\ xs)\ k$
using *interval-nth-prefix assms* **by** (*metis interval-prefix-intlen-gr-1 le-cases le-iff-add*)
have *2*: $nth\ (suffix\ i\ xs)\ k = nth\ xs\ (i+k)$
using *interval-nth-suffix assms* **by** (*simp add*: *add-le-imp-le-diff*)
have *3*: $nth\ xs\ (i+k) = nth\ (prefix\ (ia+i)\ xs)\ (i+k)$
using *interval-nth-prefix assms* **by** *simp*
have *4*: $nth\ (prefix\ (ia+i)\ xs)\ (i+k) = nth\ (suffix\ i\ (prefix\ (ia+i)\ xs))\ k$
using *interval-nth-suffix assms* **by** *simp*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *interval-suffix-prefix-help-2*:
assumes $ia+i \le intlen\ xs$
shows $(\forall\ k \le ia\ .\ nth\ (prefix\ ia\ (suffix\ i\ xs))\ k = nth\ (suffix\ i\ (prefix\ (ia+i)\ xs))\ k)$
using *interval-suffix-prefix-help-1* **using** *assms* **by** *fastforce*

lemma *interval-suffix-prefix-help-3*:
assumes $ia+i \le intlen\ xs$
shows $intlen\ (prefix\ ia\ (suffix\ i\ xs)) = intlen\ (suffix\ i\ (prefix\ (ia+i)\ xs))$
using *assms interval-prefix-length-good interval-suffix-length-good* **by** *auto*

lemma *interval-suffix-prefix-swap*:
assumes $ia+i \le intlen\ xs$
shows $prefix\ ia\ (suffix\ i\ xs) = suffix\ i\ (prefix\ (ia+i)\ xs)$
by (*simp add*: *interval-eq-nth-eq interval-suffix-prefix-help-2 interval-suffix-prefix-help-3 assms*)

lemma *interval-prefix-prefix-zero* [*simp*]:
$prefix\ 0\ (\ prefix\ 0\ xs\ ) = prefix\ 0\ xs$
by (*induct xs*) *simp-all*

lemma *interval-pref-pref* [*simp*]:
$(prefix\ i\ (prefix\ i\ xs)) = prefix\ i\ xs$
by (*metis interval-prefix-intlen interval-prefix-intlen-gr-1 interval-prefix-length*

less-imp-add-positive not-less)

lemma *interval-pref-pref-3* [*simp*]:
  (*prefix i* (*prefix* (*i*+*k*) *xs*)) = *prefix i xs*
apply (*induct xs arbitrary*: *i k*, *simp*)
apply (*case-tac i*, *auto*)
by (*simp add*: *Nitpick.case-nat-unfold*)

lemma *interval-pref-help*:
 assumes *i*≤*intlen* (*prefix* (*intlen xs* − *Suc 0*) *xs*)
 shows (*prefix i* (*prefix* (*intlen xs* − *Suc 0*) *xs*)) = (*prefix i xs*)
using *assms*
by (*metis diff-le-self interval-pref-pref-3 interval-prefix-length*
  ordered-cancel-comm-monoid-diff-class.add-diff-inverse)

lemma *interval-pref-pref-help*:
 assumes *intlen xs* >*0* ∧ *ia*<*intlen* (*xs*)
 shows (*prefix ia* (*prefix* (*intlen xs* − *Suc 0*) *xs*)) = (*prefix ia xs*)
using *assms*
by (*metis Suc-leI Suc-le-mono Suc-pred diff-le-self interval-pref-help interval-prefix-length-good*)

lemma *interval-pref-pref-help-1*:
 assumes *i*>*0* ∧ *i*≤ *intlen xs*
 shows (*prefix* (*intlen* (*prefix i xs*) − *Suc 0*) (*prefix i xs*)) =
  (*prefix* (*intlen* (*prefix i xs*) − *Suc 0*) *xs*)
using *assms interval-pref-pref-3* **by** (*metis diff-le-self interval-prefix-length-good le-iff-add*)

lemma *interval-suffix-suc* [*simp*]:
  suffix (*Suc m*) (*x* ⊙ *xs*) = *suffix m xs*
by *auto*

lemma *interval-suffix-zero* [*simp*]:
  suffix 0 xs = *xs*
by (*induct xs*) *simp-all*

lemma *interval-suffix-intlen* [*simp*]:
  suffix (*intlen xs*) *xs* = (*St* (*nth xs* (*intlen xs*)))
by (*induct xs*) *simp-all*

lemma *interval-suffix-intlast* [*simp*]:
  suffix (*intlen xs*) *xs* = *St* (*intlast xs*)
by (*induct xs*) *simp-all*

lemma *interval-suffix-suffix* [*simp*]:
  suffix i (*suffix j xs*) = *suffix* (*i*+*j*) *xs*
apply (*induct xs arbitrary*: *i j*, *simp*)
apply (*case-tac i*, *auto*)
by (*simp add*: *Nitpick.case-nat-unfold*)

lemma *interval-prefix-suffix-intlen*:

$intlen\ (prefix\ ia\ (suffix\ i\ xs)) =$
$(if\ i \leq intlen\ xs\ then$
$(if\ ia \leq intlen\ xs\ -i\ \ then\ ia\ else\ (intlen\ xs)\ -i\ )$
$else\ 0)$
by (*metis interval-prefix-length interval-suffix-length le-zero-eq*)

lemma *interval-prefix-suffix-intlen-good* [*simp*]:
assumes $ia \leq intlen\ xs\ -i \wedge i \leq intlen\ xs$
shows $intlen\ (prefix\ ia\ (suffix\ i\ xs)) = ia$
using *assms* **by** (*simp add*: *interval-prefix-suffix-intlen*)

lemma *interval-prefix-suffix-intlen-bad-0* [*simp*]:
assumes $i > intlen\ xs$
shows $intlen\ (prefix\ ia\ (suffix\ i\ xs)) = 0$
using *assms* **by** (*simp add*: *interval-prefix-suffix-intlen*)

lemma *interval-prefix-suffix-intlen-bad-1* [*simp*] :
assumes $i \leq intlen\ xs \wedge ia > intlen\ xs\ -i$
shows $intlen\ (prefix\ ia\ (suffix\ i\ xs)) = (intlen\ xs)\ -i$
using *assms* **by** (*simp add*: *interval-prefix-suffix-intlen*)

lemma *interval-suffix-suffix-3*:
assumes $i>0 \wedge ia<i \wedge i \leq intlen\ xs$
shows $(suffix\ (i-ia)\ (suffix\ ((intlen\ xs)-i)\ xs)) = (suffix\ (((intlen\ xs)-ia))\ xs)$
using *assms* **by** *simp*

lemma *interval-sub-zero-prefix* :
$sub\ 0\ k\ xs = prefix\ k\ xs$
by (*simp add*: *Interval.sub-def*)

lemma *interval-sub-suffix* :
assumes $(i < j \wedge j \leq (intlen\ xs)-k)$
shows $(sub\ (i+k)\ (j+k)\ xs) = (sub\ i\ j\ (suffix\ k\ xs))$
using *assms* **by** (*simp add*: *Interval.sub-def*)

lemma *interval-sub-prefix-suffix-0*:
assumes $(0 \leq i \wedge ia+i \leq intlen\ xs)$
shows $(sub\ i\ (i+ia)\ xs) = (prefix\ (ia)\ (suffix\ i\ xs))$
using *assms* **by** (*simp add*: *Interval.sub-def*)

lemma *interval-sub-prefix-suffix*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq intlen\ xs$
shows $(sub\ i\ j\ xs) = (prefix\ (j-i)\ (suffix\ i\ xs))$
using *assms* **by** (*simp add*: *Interval.sub-def*)

1.2.5 Reverse

lemma *interval-intlen-intapp* [*simp*]:
$intlen\ (xs \ominus ys) = (intlen\ xs) + (intlen\ ys)\ +1$
by (*induct xs arbitrary*: *ys*) *simp-all*

lemma *interval-intrev-intlen* [*simp*]:
 $intlen\ (intrev\ xs) = intlen\ xs$
by (*induct xs*, *simp*, *simp*)

lemma *interval-suffix-intapp* [*simp*]:
 $suffix\ (Suc\ (intlen\ xs))\ (xs \ominus ys) = ys$
by (*induct xs*) *simp-all*

lemma *interval-suffix-intapp2* [*simp*]:
 $suffix\ (intlen\ xs\ -\ k)\ (\ xs) \ominus ys = suffix\ (intlen\ xs\ -\ k)\ (\ xs \ominus ys)$
by (*induct xs*, *simp*)
 (*metis Suc-diff-le diff-is-0-eq′ intapp-Cons interval-suffix-suc interval-suffix-zero intlen.simps*(*2*) *not-less-eq-eq plus-1-eq-Suc*)

lemma *interval-intapp-assoc* [*simp*]:
 $(xs \ominus ys) \ominus zs = xs \ominus (\ ys \ominus zs\ )$
by (*induct xs*) *simp-all*

lemma *interval-intapp-nth*:
 $nth\ (xs \ominus ys)\ k = (if\ k \leq intlen\ xs$
 $then\ (nth\ xs\ k)$
 $else\ (nth\ ys\ (k - (intlen\ xs)\ -1))\ )$
apply (*induct xs arbitrary*: *k*)
apply (*case-tac k*, *simp*, *simp*)
apply (*case-tac k*, *simp*, *simp*)
done

lemma *interval-rev-intapp* [*simp*]:
 $intrev\ (xs \ominus ys) = (intrev\ ys) \ominus (intrev\ xs)$
by (*induct xs*) *simp-all*

lemma *interval-rev-rev-ident* [*simp*]:
 $intrev\ (intrev\ xs) = xs$
by (*induct xs*) *auto*

lemma *interval-rev-swap* :
 $((intrev\ xs) = ys) = (xs = intrev\ ys)$
by *auto*

lemma *interval-intlast-intrev*:
 $intlast\ (intrev\ xs) = intfirst\ xs$
by (*induct xs*, *auto*)
 (*metis Suc-eq-plus1 add.right-neutral interval.inject*(*1*) *interval-intlen-intapp interval-intlen-st interval-suffix-intapp interval-suffix-intlast*)

lemma *interval-intfirst-intrev*:
 $intfirst\ (intrev\ xs) = intlast\ xs$
by (*induct xs*, *auto*)
 (*metis intapp-St interval-intlast-intrev interval-rev-intapp intlast.simps*(*2*) *intrev.simps*(*1*))

lemma *interval-intrev-nth*:
$k \leq$ *intlen* (*intrev xs*) $\Longrightarrow$ (*nth* (*intrev xs*) *k*) = (*nth xs* ((*intlen xs*) $-k$))
apply (*induct xs*, *simp*)
apply *simp*
apply (*case-tac k*)
apply (*simp add*: *interval-intapp-nth*)
by (*smt Interval.nth.simps*(*1*) *Suc-diff-Suc diff-Suc-Suc diff-is-0-eq' interval-intapp-nth interval-intrev-intlen le-SucE less-Suc-eq-le old.nat.simps*(*4*) *old.nat.simps*(*5*))

lemma *interval-intrev-prefix*:
$k \leq$ *intlen xs* $\Longrightarrow$ *intrev*(*prefix k xs*) = *suffix* ((*intlen xs*) $-$ *k*) (*intrev xs*)
apply (*induct xs arbitrary*: *k*, *simp*)
apply *simp*
apply (*case-tac k*)
apply (*metis diff-zero interval-intrev-intlen interval-suffix-intapp intrev.simps*(*1*) *old.nat.simps*(*4*))
by (*metis Suc-le-mono diff-Suc-Suc interval-intrev-intlen interval-suffix-intapp2 intrev.simps*(*2*) *old.nat.simps*(*5*))

lemma *interval-intrev-suffix*:
$k \leq$ *intlen xs* $\Longrightarrow$ *intrev*(*suffix k xs*) = *prefix* ((*intlen xs*) $-$ *k*) (*intrev xs*)
by (*induct xs arbitrary*: *k*, *simp*, *simp add*: *interval-intrev-prefix interval-rev-swap*)

lemma *interval-intrev-sub1*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq$ *intlen xs*
shows *intrev* (*sub i j xs*) = *intrev* (*prefix* ($j-i$) (*suffix i xs*))
using *assms interval-sub-prefix-suffix* **by** (*simp add*: *interval-sub-prefix-suffix*)

lemma *interval-intrev-sub2*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq$ *intlen xs*
shows *intrev* (*prefix* ($j-i$) (*suffix i xs*)) = *suffix* ((*intlen xs*) $-$ *j*) (*intrev* (*suffix i xs*))
using *assms interval-intrev-prefix*[*of* $j-i$ *suffix i xs*] **by** *auto*

lemma *interval-intrev-sub3*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq$ *intlen xs*
shows *suffix* ((*intlen xs*) $-$ *j*) (*intrev* (*suffix i xs*)) =
suffix ((*intlen xs*) $-j$) (*prefix* ((*intlen xs*) $-$ *i*) (*intrev xs*))
using *assms interval-intrev-suffix*[*of i xs*] **by** *auto*

lemma *interval-intrev-sub4*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq$ *intlen xs*
shows *suffix* ((*intlen xs*) $-j$) (*prefix* ((*intlen xs*) $-$ *i*) (*intrev xs*)) =
sub ((*intlen xs*) $-j$) ((*intlen xs*) $-$ *i*) (*intrev xs*)
using *assms* **by** (*simp add*: *diff-le-mono2 interval-sub-prefix-suffix interval-suffix-prefix-swap*)

lemma *interval-intrev-sub*:
assumes $0 \leq i \wedge i \leq j \wedge j \leq$ *intlen xs*
shows *intrev* (*sub i j xs*) = *sub* ((*intlen xs*) $-j$) ((*intlen xs*) $-$ *i*) (*intrev xs*)
using *assms*
by (*simp add*: *interval-intrev-sub1 interval-intrev-sub2 interval-intrev-sub3 interval-intrev-sub4*)

lemma *interval-intrev-idx-2*:
assumes *index-sequence 0 l* $\wedge$ (*nth l* (*intlen l*)) = (*intlen xs*) $\wedge$
$0 \leq i \wedge i <$ (*intlen l*)
shows (*intrev* (*sub* (*nth l i*) (*nth l* (*i+1*)) *xs*)) =
((*sub* ((*intlen xs*)− (*nth l* (*i+1*))) ((*intlen xs*) − (*nth l i*)) (*intrev xs*)))
using *assms interval-idx-expand interval-intrev-sub*[*of* (*nth l i*) (*nth l* (*i+1*)) *xs*]
by *blast*

lemma *interval-intrev-idx-3*:
assumes *index-sequence 0 l* $\wedge$ (*nth l* (*intlen l*)) = (*intlen xs*) $\wedge$
ls = *map* (λ *x*. (*intlen xs*) −*x*) (*intrev l*)
shows (*nth ls 0*) = *0* $\wedge$ (*nth ls* (*intlen ls*)) = (*intlen xs*) $\wedge$ *intlen ls* =*intlen l*
using *assms*
by (*metis diff-self-eq-0 diff-zero index-sequence-def interval-intfirst-intrev*
interval-intlast-intrev interval-intlen-map interval-intrev-intlen
interval-nth-intlen-intlast interval-nth-map interval-nth-zero-intfirst)

lemma *interval-intrev-idx-4*:
index-sequence 0 l $\wedge$ (*nth l* (*intlen l*)) = (*intlen xs*) $\wedge$
ls = *map* (λ *x*. (*intlen xs*) −*x*) (*intrev l*)
$\Longrightarrow$ *i*$\leq$ *intlen ls* $\longrightarrow$ (*nth ls i*) = (*intlen xs*) − (*nth l* ((*intlen l*)−*i*))
apply (*induct ls*)
apply (*metis diff-zero interval-intlen-st interval-intrev-idx-3 le-0-eq*)
by (*simp add*: *interval-intlen-map interval-intrev-nth interval-nth-map*)

lemma *interval-intrev-idx-5*:
assumes (*index-sequence 0 l* $\wedge$ (*nth l* (*intlen l*)) = (*intlen xs*))
shows (*i*< *intlen l* $\longrightarrow$
(*intlen xs*) − (*nth l* ((*intlen l*)−*i*)) < (*intlen xs*) − (*nth l* ((*intlen l*)−(*i+1*))))
using *assms*
by (*smt Suc-diff-Suc Suc-eq-plus1 add-gr-0 add-less-cancel-left diff-less*
index-sequence-def le-add-diff-inverse2 le-numeral-extra(*3*) *less-diff-conv*
less-imp-le-nat not-gr-zero interval-idx-expand)

lemma *interval-intrev-idx-6*:
assumes (*index-sequence 0 l* $\wedge$ (*nth l* (*intlen l*)) = (*intlen xs*) $\wedge$
ls = *map* (λ *x*. (*intlen xs*) −*x*) (*intrev l*))
shows (*i*< *intlen ls* $\longrightarrow$
((*nth ls i*) = (*intlen xs*) − (*nth l* ((*intlen l*)−*i*)) $\wedge$
(*nth ls* (*i+1*)) = (*intlen xs*) − (*nth l* ((*intlen l*)−(*i+1*))) $\wedge$
(*nth ls i*) < (*nth ls* (*i+1*))))
proof −
have *1*: (*i*< *intlen ls* $\longrightarrow$ (*nth ls i*) = (*intlen xs*) − (*nth l* ((*intlen l*)−*i*)))
using *assms interval-intrev-idx-4 less-imp-le-nat* **by** *blast*
have *2*: (*i*< *intlen ls* $\longrightarrow$ (*nth ls* (*i+1*)) = (*intlen xs*) − (*nth l* ((*intlen l*)−(*i+1*))))
using *assms* **by** (*simp add*: *interval-intrev-idx-4*)
have *3*: (*i*< *intlen ls* $\longrightarrow$
((*nth ls i*) = (*intlen xs*) − (*nth l* ((*intlen l*)−*i*)) $\wedge$
(*nth ls* (*i+1*)) = (*intlen xs*) − (*nth l* ((*intlen l*)−(*i+1*)))))

using *1 2* **by** *auto*
have *4*: $(i< intlen\ ls \longrightarrow$
$((nth\ ls\ i) = (intlen\ xs) - (nth\ l\ ((intlen\ l){-}i)) \wedge$
$(nth\ ls\ (i{+}1)) = (intlen\ xs) - (nth\ l\ ((intlen\ l){-}(i{+}1))) \wedge$
$(nth\ ls\ i) < (nth\ ls\ (i{+}1))))$
using *assms 3 index-sequence-def interval-intrev-idx-5*
by (*metis interval-intlen-map interval-intrev-intlen*)
from *4* **show** *?thesis* **by** *blast*
qed

lemma *interval-intrev-idx-7*:
assumes $(index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs) \wedge$
$ls = map\ (\lambda\ x.\ (intlen\ xs) - x)\ (intrev\ l))$
shows *index-sequence 0 ls*
using *assms interval-intrev-idx-6 interval-intrev-idx-3*
by (*metis Suc-eq-plus1 index-sequence-def*)

lemma *interval-intrev-idx-8*:
assumes $index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs) \wedge$
$ls = map\ (\lambda\ x.\ (intlen\ xs) - x)\ (intrev\ l) \wedge index\text{-}sequence\ 0\ ls$
shows $i{<}intlen\ ls \longrightarrow$
$(intlen\ xs){-}\ (nth\ l\ (i{+}1)) = nth\ ls\ ((intlen\ ls){-}(i{+}1)) \wedge$
$(intlen\ xs) - (nth\ l\ i) \quad = (nth\ ls\ ((intlen\ ls) - i))$
using *assms interval-intrev-idx-4*
by (*smt Suc-eq-plus1 Suc-leI add-diff-cancel-right′ assms diff-diff-cancel diff-diff-left diff-le-self interval-intrev-idx-3*)

lemma *interval-intrev-idx-9*:
assumes $index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs) \wedge$
$ls = map\ (\lambda\ x.\ (intlen\ xs) - x)\ (intrev\ l) \wedge index\text{-}sequence\ 0\ ls$
shows $i{<}intlen\ ls \longrightarrow$
$sub\ ((intlen\ xs){-}\ (nth\ l\ (i{+}1)))\ ((intlen\ xs) - (nth\ l\ i))\ (intrev\ xs) =$
$sub\ (nth\ ls\ ((intlen\ ls){-}(i{+}1)))\ ((nth\ ls\ ((intlen\ ls) - i))\ )\ (intrev\ xs)$

using *interval-intrev-idx-8* **using** *assms* **by** *fastforce*

lemma *interval-intrev-idx-11*:
assumes $(index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs))$
shows $i{\le}intlen\ l \longrightarrow$
$(nth\ l\ i) = (nth\ (map\ (\lambda\ x.(intlen\ xs){-}x)\ (intrev\ (map\ (\lambda\ x.(intlen\ xs){-}x)\ (intrev\ l))))\ i)$
using *assms index-sequence-def*
by (*smt diff-diff-cancel diff-is-0-eq diff-less diff-zero leD le-cases not-gr-zero interval-intrev-idx-3 interval-intrev-idx-6 interval-intrev-idx-7*)

lemma *interval-intrev-idx-12*:
assumes $(index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ xs))$
shows $l = map\ (\lambda\ x.\ (intlen\ xs) - x\ )\ (intrev\ (map\ (\lambda\ x.\ (intlen\ xs) - x)\ (intrev\ l)))$
using *assms interval-intrev-idx-11*
by (*simp add*: *interval-intrev-idx-11 interval-eq-nth-eq interval-intlen-map*)

end

theory *Syntax*
imports
Main
abbrevs &-*i* = $\wedge_i$ **and**
|-*i* = $\vee_i$ **and**
)-*i* = $\supset_i$ **and**
$-$-*i* = $\neg_i$ **and**
=-*i* = $\equiv_i$ **and**
false-i = *false*$_i$ **and**
true-i = *true*$_i$ **and**
if-i = *if*$_i$ **and**
-$*$ = *
begin

2 Syntax

The basic ITL syntax is introduced first followed by the derived operators including the time reversal operator ([4]).

2.1 Primitive formulae

datatype *'a pitl* =
false-d ($false_i$)
| *atom-d 'a* $\Rightarrow$ *bool* (($atom_i$ -))
| *implies-d 'a pitl 'a pitl* ((- $\supset_i$ -) [26,25] 25)
| *skip-d* (*skip*)
| *chop-d 'a pitl 'a pitl* ((- ; -) [84,84] 83)
| *chopstar-d 'a pitl* ((- *) [85] 85)

2.2 Derived Boolean Operators

definition *not-d* (($\neg_i$ -) [90] 90)
where
$\neg_i\ f \equiv f \supset_i false_i$

definition *true-d* ($true_i$)
where
$true_i \equiv \neg_i\ false_i$

definition *or-d* ((- $\vee_i$ -) [31,30] 30)
where
$f \vee_i g \equiv \neg_i\ f \supset_i g$

definition *and-d* ((- $\wedge_i$ -) [36,35] 35)
where

$f \wedge_i g \equiv \neg_i (\neg_i f \vee_i \neg_i g)$

definition *iff-d* ((- $\equiv_i$ -) [*21*,*20*] *20*)
where
$f \equiv_i g \equiv ((f \supset_i g) \wedge_i (g \supset_i f))$

2.3 Next and Previous Operators

definition *next-d* (($\bigcirc$ -) [*88*] *87*)
where
$\bigcirc f \equiv skip \ ; f$

definition *wnext-d* ((*wnext* -) [*88*] *87*)
where
$wnext\ f \equiv \neg_i (\bigcirc (\neg_i f))$

definition *prev-d* ((*prev* -) [*88*] *87*)
where
$prev\ f \equiv f;skip$

definition *wprev-d* ((*wprev* -) [*88*] *87*)
where
$wprev\ f \equiv \neg_i(prev(\neg_i f))$

2.4 More and Empty

definition *more-d* (*more*)
where
$more \equiv \bigcirc\ true_i$

definition *empty-d* (*empty*)
where
$empty \equiv \neg_i\ more$

2.5 Box and Diamond Operators

definition *sometimes-d* (($\Diamond$ -) [*88*] *87*)
where
$\Diamond f \equiv true_i \ ; f$

definition *always-d* (($\Box$ -) [*88*] *87*)
where
$\Box f \equiv \neg_i (\Diamond (\neg_i f))$

definition *di-d* ((*di* -) [*88*] *87*)
where
$di\ f \equiv f \ ; true_i$

definition *bi-d* ((*bi* -) [*88*] *87*)
where

$bi\ f \equiv \neg_i\ (di\ (\neg_i\ f))$

definition *da-d* ((*da* -) [*88*] *87*)
where
$da\ f \equiv true_i\ ;\ (f\ ;\ true_i)$

definition *ba-d* ((*ba* -) [*88*] *87*)
where
$ba\ f \equiv \neg_i\ (da\ (\neg_i\ f))$

definition *dm-d* ((*dm* -) [*88*] *87*)
where
$dm\ f \equiv \Diamond\ (more \wedge_i f)$

definition *bm-d* ((*bm* -) [*88*] *87*)
where
$bm\ f \equiv \neg_i\ (dm\ (\neg_i\ f))$

2.6 Initial and Final Operators

definition *init-d* ((*init* -) [*88*] *87*)
where
$init\ f \equiv ((empty \wedge_i f);\ true_i)$

definition *fin-d* ((*fin* -) [*88*] *87*)
where
$fin\ f \equiv \Box\ (\ empty \supset_i f)$

2.7 Programming Constructs

definition *halt-d* ((*halt* -) [*88*] *87*)
where
$halt\ f \equiv \Box\ (\ empty \equiv_i f)$

definition *keep-d* ((*keep* -) [*88*] *87*)
where
$keep\ f \equiv ba\ (\ skip \supset_i f)$

definition *yields-d* ((- *yields* -) [*88*,*88*] *87*)
where
$f\ yields\ g \equiv \neg_i\ (\ f;\ \neg_i\ g)$

definition *ifthenelse-d* ((*if* $_i$ - *then* - *else* -) [*88*,*88*,*88*] *87*)
where
$if_i\ f\ then\ g\ else\ h \equiv (f \wedge_i g) \vee_i (\ (\neg_i\ f) \wedge_i h)$

definition *ifthen-d* ((*if* $_i$ - *then* -) [*88*,*88*] *87*)
where
$if_i\ f\ then\ g\ \equiv if_i\ f\ then\ g\ else\ true_i$

definition *while-d* ((*while* - *do* -) [*88*,*88*] *87*)
where
$\textit{while f do g} \equiv (f \wedge_i g)^\star \wedge_i \textit{fin} (\neg_i f)$

definition *repeat-d* ((*repeat* - *until* -) [*88*,*88*] *87*)
where
$\textit{repeat f until g} \equiv f \;;\; (\textit{while} (\neg_i g) \textit{ do } f)$

primrec *len-d* :: *nat* ⇒ *'a pitl* ((*len* -) [*88*] *87*) **where**
(*len 0*) = *empty*
| (*len* (*Suc n*)) = (*skip* ; (*len n*))

primrec *power-chop-d* :: *'a pitl* ⇒ *nat* ⇒ *'a pitl* **where**
power-0 : *power-chop-d f 0* = *empty*
| *power-Suc*: *power-chop-d f* (*Suc n*) = $((f \wedge_i \textit{more});(\textit{power-chop-d f n}))$

primrec *power-d* :: *'a pitl* ⇒ *nat* ⇒ *'a pitl* ((*power* - -) [*88*,*88*] *87*) **where**
pow-0 : *power-d f 0* = *empty*
| *pow-Suc*: *power-d f* (*Suc n*) = ((*f*);(*power-d f n*))

2.8 Time reversal

primrec *rev-d* :: *'a pitl* ⇒ *'a pitl* ((- ^{r}) [*88*] *87*)
where
$\textit{false}_i{}^r = \textit{false}_i$
| $(\textit{atom}_i\ p)^r = \textit{fin}\ (\textit{atom}_i\ p)$
| $(f1 \supset_i f2)^r = (f1^r \supset_i f2^r)$
| $\textit{skip}^r = \textit{skip}$
| $(f1 \;;\; f2)^r = (f2^r \;;\; f1^r)$
| $(f^\star)^r = (f^r)^\star$

end

theory *Semantics*
imports
Interval
Syntax
begin

3 Semantics

The semantics of the basic ITL operators is defined. Then lemmas are introduced that define the derived ITL operators in terms of operations on intervals. Furthermore we prove the soundness of the ITL axioms. This is followed by the key time reversal lemma.

3.1 Semantics Primitive Operators

fun *semantics-pitl* :: [*'a interval*, *'a pitl*] ⇒ *bool* ((- ⊨ -) [*80*,*10*] *10*)

where
$(\sigma \models \mathit{false}_i) = \mathit{False}$
$\mid (\sigma \models \mathit{atom}_i\ p) = (p\ (\mathit{intfirst}\ \sigma))$
$\mid (\sigma \models f \supset_i g) = ((\sigma \models f) \longrightarrow (\sigma \models g))$
$\mid (\sigma \models \mathit{skip}) = (\mathit{intlen}\ \sigma = 1)$
$\mid (\sigma \models f\ ;\ g) =$
$(\exists\, i.\ ((0 \le i) \wedge (i \le (\mathit{intlen}\ \sigma)) \wedge ((\mathit{prefix}\ i\ \sigma) \models f) \wedge ((\mathit{suffix}\ i\ \sigma) \models g))\ )$
$\mid (\sigma \models f^{\star}) = (\exists\ (l{::}\mathit{index}).\ \mathit{index\text{-}sequence}\ 0\ l\ \wedge (\mathit{nth}\ l\ (\mathit{intlen}\ l)) = (\mathit{intlen}\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (\mathit{intlen}\ l)) \longrightarrow$
$((\mathit{sub}\ (\mathit{nth}\ l\ i)\ (\mathit{nth}\ l\ (i+1))\ \sigma) \models f)$
$)$
$)$

3.2 Semantics Boolean Operators

lemma *not-defs* [*simp*]:
$(\sigma \models \neg_i\ f\ ) = \mathit{Not}\ (\ \sigma \models f)$
by (*simp add*: *not-d-def*)

lemma *or-defs* [*simp*]:
$(\sigma \models f1 \vee_i f2) = (\ (\sigma \models f1) \vee (\sigma \models f2))$
by (*metis not-defs or-d-def semantics-pitl.simps*(*3*))

lemma *and-defs* [*simp*]:
$(\sigma \models f1 \wedge_i f2) = (\ (\sigma \models f1) \wedge (\sigma \models f2))$
by (*simp add*: *and-d-def*)

lemma *implies-defs* [*simp*]:
$(\sigma \models f1 \supset_i f2) = (\ (\sigma \models f1) \longrightarrow (\sigma \models f2))$
by *auto*

lemma *iff-defs* [*simp*]:
$(\sigma \models f1 \equiv_i f2) = (\ (\sigma \models f1) = (\sigma \models f2))$
by (*metis and-defs iff-d-def semantics-pitl.simps*(*3*))

lemma *true-defs* [*simp*]:
$(\sigma \models \mathit{true}_i)$
by (*simp add*: *true-d-def*)

3.3 Semantics Box and Diamond Operators

lemma *sometimes-defs* [*simp*]:
$(\sigma \models \Diamond\ f) = (\exists\ i \le \mathit{intlen}\ \sigma.\ \ (\mathit{suffix}\ i\ \sigma \models f))$
by (*simp add*: *sometimes-d-def*)

lemma *always-defs* [*simp*]:
$(\sigma \models \Box\ f) = (\forall\ i \le \mathit{intlen}\ \sigma.\ \ (\mathit{suffix}\ i\ \sigma \models f))$
by (*simp add*: *always-d-def*)

lemma *di-defs* [*simp*]:

$(\sigma \models di\ f) = (\exists\ i \le intlen\ \sigma.\ (prefix\ i\ \sigma \models f))$
by (*simp add*: *di-d-def*)

lemma *bi-defs* [*simp*]:
$(\sigma \models bi\ f) = (\forall\ i \le intlen\ \sigma.\ (prefix\ i\ \sigma \models f))$
by (*simp add*: *bi-d-def*)

lemma *da-defs* [*simp*]:
$(\sigma \models da\ f) = (\exists\ i\ ia\ .\ 0 \le i\ \wedge ia{+}i \le intlen\ \sigma \wedge (sub\ i\ (ia\ {+}i)\ \sigma \models f))$
apply (*simp add*: *da-d-def*)
using *interval-prefix-length-good interval-suffix-length-good*
by (*smt add.commute add-diff-cancel-left′ add-leD2 interval-sub-prefix-suffix-0 le-iff-add nat-add-left-cancel-le zero-le*)

lemma *ba-defs* [*simp*]:
$(\sigma \models ba\ f) = (\forall\ i\ ia\ .\ (0 \le i \wedge ia{+}i \le intlen\ \sigma) \longrightarrow (sub\ i\ (ia{+}i)\ \sigma \models f))$
by (*metis ba-d-def da-defs not-defs*)

3.4 Semantics Next and Previous Operators

lemma *skip-defs* :
$(\sigma\ \models skip) = (\ (intlen\ \sigma)\ {=}1)$
by *simp*

lemma *next-defs* [*simp*]:
$(\sigma \models \bigcirc\ f) = (\ (intlen\ \sigma) > 0 \wedge ((suffix\ 1\ \sigma) \models f))$
using *Suc-le-eq* **by** (*simp add*: *next-d-def*) *force*

lemma *wnext-defs* [*simp*]:
$(\sigma \models wnext\ f) = ((intlen\ \sigma)\ {=}0 \vee ((suffix\ 1\ \sigma) \models f))$
by (*simp add*: *wnext-d-def*)

lemma *prev-defs* [*simp*]:
$(\sigma \models prev\ f) = (\ (intlen\ \sigma) > 0 \wedge ((prefix\ ((intlen\ \sigma){-}1)\ \sigma) \models f))$
by (*simp add*: *prev-d-def*)
(*metis One-nat-def Suc-leI diff-diff-cancel diff-is-0-eq′ diff-le-self interval-suffix-length-good neq0-conv zero-neq-one*)

lemma *wprev-defs* [*simp*]:
$(\sigma \models wprev\ f) = (\ (intlen\ \sigma)\ {=}0 \vee ((prefix\ ((intlen\ \sigma){-}1)\ \sigma) \models f))$
by (*simp add*: *wprev-d-def*)

3.5 Semantics More and Empty

lemma *more-defs* [*simp*] :
$(\sigma \models more) = (intlen\ \sigma > 0)$
by (*simp add*: *more-d-def*)

lemma *empty-defs* [*simp*]:
$(\sigma \models empty) = (\ (intlen\ \sigma) = 0)$

by (*simp add*: *empty-d-def*)

3.6 Semantics Initial and Final Operators

lemma *init-defs* [*simp*]:
  $(\sigma \models init\ f) = (\ (St\ (intfirst\ \sigma)) \models f)$
by (*simp add*: *init-d-def*) *auto*

lemma *fin-defs* [*simp*]:
  $(\sigma \models fin\ f) = (\ (St\ (intlast\ \sigma\ )) \models f)$
by (*simp add*: *fin-d-def interval-nth-intlen-intlast*)

3.7 Semantics Programming Constructs

lemma *ifthenelse-defs* [*simp*]:
  $(\sigma \models if_i\ f0\ then\ f1\ else\ f2) =$
  $(\ ((\sigma \models f0) \wedge (\sigma \models f1)) \vee (\ \neg(\sigma \models f0) \wedge (\sigma \models f2)))$
by (*simp add*: *ifthenelse-d-def*)

lemma *len-defs* [*simp*]:
  $(\sigma \models len\ n) = ((intlen\ \sigma) = n)$
by (*induct n arbitrary*: σ, *simp*, *simp*) *fastforce*

3.8 Soundness Axioms

3.8.1 ChopAssoc

lemma *ChopAssocSemHelp*:
 $(\exists\, i\ ia\ .\ i \le intlen\ \sigma \wedge ia \le intlen\ \sigma - i \wedge (prefix\ i\ \sigma \models f) \wedge$
   $(prefix\ ia\ (suffix\ i\ \sigma) \models g) \wedge (suffix\ (ia + i)\ \sigma \models h)) =$
  $(\exists\, j\ ja\ .\ j \le intlen\ \sigma \wedge ja \le j \wedge (prefix\ ja\ (prefix\ j\ \sigma) \models f) \wedge$
   $(suffix\ ja\ (prefix\ j\ \sigma) \models g) \wedge (suffix\ j\ \sigma \models h))$
by (*smt Nat.le-diff-conv2 add-diff-cancel-left' interval-pref-pref-3 interval-suffix-prefix-swap*
    le-add1 le-add-diff-inverse2 le-trans)

lemma *ChopAssocSemHelp2*:
  $(\sigma \models\ f\ ; (g\ ;\ h)) = (\sigma \models (f;g);h)$
proof −
 have $(\sigma \models\ f\ ; (g\ ;\ h)) =$
    $((\exists\, i \le intlen\ \sigma.\ (prefix\ i\ \sigma \models f) \wedge (\exists\, ia \le intlen\ (suffix\ i\ \sigma).$
     $(prefix\ ia\ (suffix\ i\ \sigma) \models g) \wedge (suffix\ (ia + i)\ \sigma \models h))))$
 by *simp-all*
 also have ... =
      $(\exists\, i\ ia\ .\ i \le intlen\ \sigma \wedge ia \le intlen\ \sigma - i \wedge (prefix\ i\ \sigma \models f) \wedge$
    $(prefix\ ia\ (suffix\ i\ \sigma) \models g) \wedge (suffix\ (ia + i)\ \sigma \models h))$
 by *fastforce*
 also have ... =
     $(\exists\, j\ ja\ .\ j \le intlen\ \sigma \wedge ja \le j \wedge (prefix\ ja\ (prefix\ j\ \sigma) \models f) \wedge$
     $(suffix\ ja\ (prefix\ j\ \sigma) \models g) \wedge (suffix\ j\ \sigma \models h))$
 using *ChopAssocSemHelp*[*of* σ *f g h*] **by** *blast*
 also have ... =

$(\exists\, i \leq intlen\ \sigma.\ (\exists\, ia \leq intlen\ (prefix\ i\ \sigma).\ \ (prefix\ ia\ (prefix\ i\ \sigma) \models f) \wedge$
$(suffix\ ia\ (prefix\ i\ \sigma) \models g)) \wedge (suffix\ i\ \sigma \models h))$
by *fastforce*
also have ... =
$(\sigma \models (f;g);h)$ **by** *simp-all*
finally show $(\sigma \models\ f\ ;\ (g\ ;\ h)) = (\sigma \models (f;g);h)$ **.**
qed

lemma *ChopAssocSem*:
$(\sigma \models\ f\ ;\ (g\ ;\ h) \equiv_i\ (f;g);h)$
using *ChopAssocSemHelp2* **using** *iff-defs* **by** *blast*

3.8.2 OrChopImp

lemma *OrChopImpSem*:
$(\sigma \models\ (\ f \vee_i g);h\ \ \supset_i\ f;h \vee_i g;h\ )$
by *simp-all blast*

3.8.3 ChopOrImp

lemma *ChopOrImpSem*:
$(\sigma \models f;(g \vee_i h) \supset_i\ \ f;g \vee_i\ f;h\ )$
by *simp-all blast*

3.8.4 EmptyChop

lemma *EmptyChopSem*:
$(\sigma \models empty\ ;\ f \equiv_i f\ )$
by *simp-all auto*

3.8.5 ChopEmpty

lemma *ChopEmptySem*:
$(\sigma \models f;empty \equiv_i f\ )$
by *simp-all auto*

3.8.6 StateImpBi

lemma *StateImpBiSem*:
$(\sigma \models init\ f \supset_i\ \ bi\ (init\ f)\ )$
by *simp-all*

3.8.7 NextImpNotNextNot

lemma *NextImpNotNextNotSem*:
$(\sigma \models \bigcirc\ f \supset_i\ \neg_i\ (\bigcirc\ \neg_i\ f)\ )$
by *auto*

3.8.8 BiBoxChopImpChop

lemma *BiBoxChopImpChopSem*:
$(\sigma \models bi\ (\ f \supset_i f1) \wedge_i \Box(g \supset_i g1) \supset_i f;g \supset_i f1;g1\ )$

by *fastforce*

3.8.9 BoxInduct

lemma *box-induct-help-1* :
$(\sigma \models f) \wedge (\forall\, i.\ Suc\ 0 \leq intlen\ \sigma\ -\ i \longrightarrow$
$\quad i \leq intlen\ \sigma \longrightarrow (suffix\ i\ \sigma \models f) \longrightarrow (suffix\ (Suc\ i)\ \sigma \models f))$
$\quad \Longrightarrow (\forall\ j.\ j \leq intlen\ \sigma \longrightarrow (suffix\ j\ \sigma \models f))$
proof
fix *j*
show $(\sigma \models f) \wedge (\forall\, i.\ Suc\ 0 \leq intlen\ \sigma\ -\ i \longrightarrow$
$\quad i \leq intlen\ \sigma \longrightarrow (suffix\ i\ \sigma \models f) \longrightarrow (suffix\ (Suc\ i)\ \sigma \models f))$
$\quad \Longrightarrow\ j \leq intlen\ \sigma \longrightarrow (suffix\ j\ \sigma \models f)$
proof
(*induct j arbitrary*: σ)
case *0*
then show *?case* **by** *simp*
next
case (*Suc j*)
then show *?case*
by (*metis Nat.le-diff-conv2 One-nat-def Suc-eq-plus1-left Suc-leD*)
qed
qed

lemma *BoxInductSem*:
$(\sigma \models \Box\ (f \supset_i\ wnext\ f) \wedge_i f \supset_i \Box\ f)$
apply (*simp*)
using *box-induct-help-1* **by** (*metis One-nat-def diff-self-eq-0 not-one-le-zero*)

3.8.10 ChopStarEqv

lemma *chopstar-help-1*:
$(\ \exists\, l.\ l{=}\ \langle 0\rangle \wedge index\text{-}sequence\ 0\ l\ \wedge$
$\quad Interval.nth\ l\ (intlen\ l) = (intlen\ \sigma) \wedge$
$\quad (\forall\, i.\ (0{\leq}i \wedge i{<}\ (intlen\ l)) \longrightarrow$
$\qquad ((sub\ (nth\ l\ i)\ (nth\ l\ (i{+}1))\ \sigma) \models f)$
$\quad)) \longleftrightarrow (intlen\ \sigma\ {=}0)$
by (*simp add*: *index-sequence-def*)

lemma *chopstar-help-2*:
$(\forall\, i.\ (0{<}i \wedge i{<}\ 1{+}(intlen\ ls)) \longrightarrow$
$\qquad ((sub\ (nth\ ls\ (i{-}1))\ (nth\ ls\ ((i{-}1){+}1))\ \sigma) \models f)$
$\quad) =$
$(\forall\, i.\ (0{\leq}i \wedge i{<}\ (intlen\ ls)) \longrightarrow$
$\qquad ((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i){+}1))\ \sigma) \models f)$
$\quad)$
by (*metis Suc-eq-plus1 Suc-pred add-diff-cancel-right' add-less-cancel-left add-nonneg-pos le-add2 le-add-same-cancel2 plus-1-eq-Suc zero-less-one*)

lemma *chop-power-chain*:
$(\exists\ (l{::}index).\ (intlen\ l) = (Suc\ n) \wedge index\text{-}sequence\ 0\ l \wedge (nth\ l\ (intlen\ l)) = (intlen\ \sigma)\ \wedge$

$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ \sigma) \models f)$
$)$
$) =$
$(\exists\ k.\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma))$
$\wedge (\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ (suffix\ k\ \sigma)) \models f)$
$))$
$)$

proof −
have $(\exists\ (l{::}index).\ (intlen\ l) = (Suc\ n) \wedge index\text{-}sequence\ 0\ l \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ \sigma) \models f)$
$)$
$)$
$=$
$(\exists\ x\ ls\ l.\ (intlen\ l) = (Suc\ n) \wedge l{=}x{\odot}ls \wedge index\text{-}sequence\ 0\ l \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ \sigma) \models f)$
$)$
$)$
using *interval-intlen-cons-1* **by** (*metis zero-less-Suc*)
also have ... =
$(\exists\ x\ ls\ l.\ (intlen\ l) = (Suc\ n) \wedge l{=}x{\odot}ls \wedge index\text{-}sequence\ 0\ (x{\odot}ls) \wedge$
$(nth\ (x{\odot}ls)\ (intlen\ (x{\odot}ls))) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ (x{\odot}ls))) \longrightarrow$
$((sub\ (nth\ (x{\odot}ls)\ i)\ (nth\ (x{\odot}ls)\ (i+1))\ \sigma) \models f)$
$)$
$)$
by *auto*
also have ... =
$(\exists\ x\ ls\ .\ (intlen\ ls) = n \wedge index\text{-}sequence\ 0\ (x{\odot}ls) \wedge$
$(nth\ (x{\odot}ls)\ (intlen\ (x{\odot}ls))) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ (x{\odot}ls))) \longrightarrow$
$((sub\ (nth\ (x{\odot}ls)\ i)\ (nth\ (x{\odot}ls)\ (i+1))\ \sigma) \models f)$
$)$
$)$
by *auto*
also have ... =
$(\exists\ x\ ls\ .\ (intlen\ ls) = n \wedge x = 0 \wedge index\text{-}sequence\ 0\ (x \odot ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ \sigma) \wedge$
$((\forall\, i.\ (0 \leq i \wedge i < (intlen\ (x{\odot}ls))) \longrightarrow$
$((sub\ (nth\ (x{\odot}ls)\ i)\ (nth\ (x{\odot}ls)\ (i+1))\ \sigma) \models f))$
$)$
$)$

by (*simp add*: *index-sequence-def*)
also have ... =
(∃ x ls . (intlen ls) = n ∧ x =0 ∧ index-sequence (nth ls 0) (ls) ∧
(nth (ls) (intlen (ls))) = (intlen σ) ∧
(x < (nth ls 0) ∧
((∀ i. (0≤i ∧ i< (intlen (x⊙ls))) ⟶
((sub (nth (x⊙ls) i) (nth (x⊙ls) (i+1)) σ) ⊨ f))
)
)
)
using *interval-idx-cons* **by** *auto*
also have ... =
(∃ x ls . (intlen ls) = n ∧ x =0 ∧ index-sequence (nth ls 0) (ls) ∧
(nth (ls) (intlen (ls))) = (intlen σ) ∧
(x < (nth ls 0) ∧
((sub (nth (x⊙ls) 0) (nth (x⊙ls) (1)) σ) ⊨ f)
∧
((∀ i. (0<i ∧ i< 1+ (intlen (ls))) ⟶
((sub (nth (x⊙ls) i) (nth (x⊙ls) (i+1)) σ) ⊨ f))
)
)
)
by (*metis* (*no-types*, *lifting*) *One-nat-def add.right-neutral add-Suc add-Suc-right add-cancel-right-left interval-intlen-cons not-gr-zero zero-le zero-less-Suc*)
also have ... =
(∃ x ls . (intlen ls) = n ∧ x =0 ∧ index-sequence (nth ls 0) (ls) ∧
(nth (ls) (intlen (ls))) = (intlen σ) ∧
(x < (nth ls 0) ∧ (nth (x⊙ls) 0) = x ∧ (nth (x⊙ls) (1)) = (nth ls 0) ∧
((sub (nth (x⊙ls) 0) (nth (x⊙ls) (1)) σ) ⊨ f)
∧
((∀ i. (0<i ∧ i< 1+ (intlen (ls))) ⟶
((sub (nth (x⊙ls) i) (nth (x⊙ls) (i+1)) σ) ⊨ f))
)
)
)
by *auto*
also have ... =
(∃ x ls . (intlen ls) = n ∧ x =0 ∧ index-sequence (nth ls 0) (ls) ∧
(nth (ls) (intlen (ls))) = (intlen σ) ∧
(x < (nth ls 0) ∧ (nth (x⊙ls) 0) = x ∧ (nth (x⊙ls) (1)) = (nth ls 0) ∧
((sub x (nth ls 0) σ) ⊨ f)
∧
((∀ i. (0<i ∧ i< 1+ (intlen (ls))) ⟶
((sub (nth (x⊙ls) i) (nth (x⊙ls) (i+1)) σ) ⊨ f))
)
)
)
by *auto*
also have ... =
(∃ x ls . (intlen ls) = n ∧ x =0 ∧ index-sequence (nth ls 0) (ls) ∧

$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ \sigma) \wedge$
$(x < (nth\ ls\ 0) \wedge$
$((sub\ x\ (nth\ ls\ 0)\ \sigma) \models f)$
$\wedge$
$((\forall i.\ (0<i \wedge i< 1+\ (intlen\ (ls))) \longrightarrow$
$((sub\ (nth\ (x \odot ls)\ i)\ (nth\ (x \odot ls)\ (i+1))\ \sigma) \models f))$
$)$
$)$
$)$
by *auto*
also have ... =
$(\exists\ x\ ls\ .\ (intlen\ ls) = n \wedge x = 0 \wedge index\text{-}sequence\ (nth\ ls\ 0)\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ \sigma) \wedge$
$(\ x < (nth\ ls\ 0) \wedge$
$((sub\ x\ (nth\ ls\ 0)\ \sigma) \models f)$
$\wedge$
$(\forall i.\ (0<i \wedge i< 1+(intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i-1))\ (nth\ ls\ ((i-1)+1))\ \sigma) \models f)$
$)))$
using *interval-nth-cons* **by** *metis*
also have ... =
$(\exists\ x\ ls\ .\ (intlen\ ls) = n \wedge x = 0 \wedge index\text{-}sequence\ (nth\ ls\ 0)\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ \sigma) \wedge$
$(x < (nth\ ls\ 0) \wedge$
$((sub\ x\ (nth\ ls\ 0)\ \sigma) \models f))$
$\wedge\ (\forall i.\ (0 \leq i \wedge i< (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ \sigma) \models f)$
$)$
$)$
using *chopstar-help-2* **by** *metis*
also have ... =
$(\exists\ \ ls\ .\ (intlen\ ls) = n \wedge index\text{-}sequence\ (nth\ ls\ 0)\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ \sigma) \wedge$
$(0 < (nth\ ls\ 0) \wedge$
$((sub\ 0\ (nth\ ls\ 0)\ \sigma) \models f))$
$\wedge\ (\forall i.\ (0 \leq i \wedge i< (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ \sigma) \models f)$
$)$
$)$
by *simp*
also have ... =
$(\exists\ \ lsk\ .\ \ (intlen\ lsk) = n \wedge (nth\ lsk\ 0) \leq intlen\ \sigma \wedge (nth\ lsk\ 0) > 0 \wedge$
$((sub\ 0\ (nth\ lsk\ 0)\ \sigma) \models f) \wedge$
$index\text{-}sequence\ (nth\ lsk\ 0)\ (lsk) \wedge$
$(nth\ (lsk)\ (intlen\ (lsk))) = (intlen\ \sigma) \wedge$
$(\forall i.\ (0 \leq i \wedge i< (intlen\ lsk)) \longrightarrow$
$((sub\ (nth\ lsk\ (i))\ (nth\ lsk\ ((i)+1))\ \sigma) \models f)$
$)$
$)$
by (*metis Suc-eq-plus1 Suc-pred add.left-neutral eq-iff interval-idx-less-last*

interval-intlen-gr-zero le-neq-implies-less lessI less-imp-le-nat)
also have ... =
  ($\exists$ *k lsk*. (*intlen lsk*) = *n* $\wedge$ (*nth lsk 0*) $\leq$ *intlen* σ $\wedge$
    (*nth lsk 0*) > *0* $\wedge$ *k*=(*nth lsk 0*) $\wedge$
    (*sub 0* (*nth lsk 0*) σ $\models$ *f*) $\wedge$
    index-sequence (*nth lsk 0*) (*lsk*) $\wedge$
    (*nth* (*lsk*) (*intlen* (*lsk*))) = (*intlen* (σ)) $\wedge$
    ($\forall$ *i*. (*0*$\leq$*i* $\wedge$ *i*< (*intlen lsk*)) $\longrightarrow$
      ((*sub* ((*nth lsk* (*i*))) ((*nth lsk* ((*i*)+*1*))) (σ)) $\models$ *f*)
)
)
by *auto*
also have ... =
  ($\exists$ *k lsk*. (*intlen lsk*) = *n* $\wedge$ *0* $\leq$*k* $\wedge$ *k* $\leq$ *intlen* σ $\wedge$ *k* > *0* $\wedge$ *k*=(*nth lsk 0*) $\wedge$
    (*sub 0 k* σ $\models$ *f*) $\wedge$
    (*index-sequence k* (*lsk*) $\wedge$
      (*nth* (*lsk*) (*intlen* (*lsk*))) = ((*intlen* (*suffix k* σ))+*k*) $\wedge$
    ($\forall$ *i*. (*0*$\leq$*i* $\wedge$ *i*< (*intlen lsk*)) $\longrightarrow$
      ((*sub* ((*nth lsk* (*i*))) ((*nth lsk* ((*i*)+*1*))) (σ)) $\models$ *f*)
    ))
)
apply (*simp add*: *interval-prefix-suffix-intlen interval-suffix-length interval-prefix-length*)
by *auto*
also have ... =
  ($\exists$ *k lsk*. (*intlen lsk*) = *n* $\wedge$ *0* $\leq$*k* $\wedge$ *k* $\leq$ *intlen* σ $\wedge$ *k* > *0* $\wedge$
    (*sub 0 k* σ $\models$ *f*) $\wedge$
    (*index-sequence k* (*lsk*) $\wedge$
      (*nth* (*lsk*) (*intlen* (*lsk*))) = ((*intlen* (*suffix k* σ))+*k*) $\wedge$
    ($\forall$ *i*. (*0*$\leq$*i* $\wedge$ *i*< (*intlen lsk*)) $\longrightarrow$
      ((*sub* ((*nth lsk* (*i*))) ((*nth lsk* ((*i*)+*1*))) (σ)) $\models$ *f*)
    ))
)
using *index-sequence-def* **by** *auto*
also have ... =
  ($\exists$ *k*. *0* $\leq$*k* $\wedge$ *k* $\leq$ *intlen* σ $\wedge$ *k* > *0* $\wedge$
    (*sub 0 k* σ $\models$ *f*) $\wedge$
    ($\exists$ *ls lsk*. (*intlen lsk*) = *n* $\wedge$ *index-sequence k* (*lsk*) $\wedge$
      ls = *map* (*shiftm k*) *lsk* $\wedge$
      (*nth* (*lsk*) (*intlen* (*lsk*))) = ((*intlen* (*suffix k* σ))+*k*) $\wedge$
    ($\forall$ *i*. (*0*$\leq$*i* $\wedge$ *i*< (*intlen lsk*)) $\longrightarrow$
      ((*sub* ((*nth lsk* (*i*))) ((*nth lsk* ((*i*)+*1*))) (σ)) $\models$ *f*)
    ))
)
by *blast*
also have ... =
  ($\exists$ *k*. *0* $\leq$*k* $\wedge$ *k* $\leq$ *intlen* σ $\wedge$ *k* > *0* $\wedge$
    (*sub 0 k* σ $\models$ *f*) $\wedge$
    ($\exists$ *ls lsk*. (*intlen lsk*) = *n* $\wedge$ *index-sequence k* (*lsk*) $\wedge$
      ls = *map* (*shiftm k*) *lsk* $\wedge$
      index-sequence 0 (*ls*) $\wedge$ (*intlen ls*) = *n* $\wedge$

$(nth\ (lsk)\ (intlen\ (lsk))) = ((intlen\ (suffix\ k\ \sigma))+k) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ lsk)) \longrightarrow$
$((sub\ ((nth\ lsk\ (i)))\ ((nth\ lsk\ ((i)+1)))\ (\sigma)) \models f)$
$))$
$)$
using *interval-idx-link-shiftm* **by** *blast*
also have ... =
$(\exists\ k.\ \ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls\ lsk.\ (intlen\ lsk) = n\ \wedge index\text{-}sequence\ k\ (lsk) \wedge$
$lsk = map\ (shift\ k)\ ls \wedge$
$index\text{-}sequence\ 0\ (ls) \wedge (intlen\ ls) = n\ \wedge$
$(nth\ (lsk)\ (intlen\ (lsk))) = ((intlen\ (suffix\ k\ \sigma))+k) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ lsk)) \longrightarrow$
$((sub\ ((nth\ lsk\ (i)))\ ((nth\ lsk\ ((i)+1)))\ (\sigma)) \models f)$
$))$
$)$
using *interval-lsk-ls* **by** *blast*
also have ... =
$(\exists\ k\ ls\ lsk\ .\ \ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\ (intlen\ lsk) = n\ \wedge lsk = map\ (shift\ k)\ ls \wedge$
$index\text{-}sequence\ 0\ (ls) \wedge$
$index\text{-}sequence\ k\ (lsk) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ ((nth\ ls\ (i))+k)\ ((nth\ ls\ ((i)+1))+k)\ (\sigma)) \models f)$
$))$
$)$
apply (*simp add*: *Interval.shift-def interval-intlen-map interval-nth-map*) **by** *blast*
also have ... =
$(\exists\ k\ ls\ lsk.\ \ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\ (intlen\ lsk) = n\ \wedge lsk = map\ (shift\ k)\ ls \wedge$
$(intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ ((nth\ ls\ (i))+k)\ ((nth\ ls\ ((i)+1))+k)\ (\sigma)) \models f)$
$))$
$)$
using *interval-idx-link* **by** *blast*
also have ... =
$(\exists\ k.\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ ((nth\ ls\ (i))+k)\ ((nth\ ls\ ((i)+1))+k)\ (\sigma)) \models f)$
$))$
$)$

by (*simp add*: *interval-intlen-map*)
also have ... =
$(\exists\ k\ .\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i \leq intlen\ ls.\ Interval.nth\ ls\ i \leq intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ ((nth\ ls\ (i))+k)\ ((nth\ ls\ ((i)+1))+k)\ (\sigma)) \models f)$
$)$
$)$
$)$
using *interval-idx-bound-1* **by** *blast*
also have ... =
$(\exists\ k.\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i \leq intlen\ ls.\ Interval.nth\ ls\ i \leq intlen\ (suffix\ k\ \sigma))$
$\wedge (\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ (suffix\ k\ \sigma)) \models f)$
$)$
$)$
$)$
by (*smt add.commute index-sequence-def interval-idx-expand interval-sub-suffix interval-suffix-length-good plus-1-eq-Suc*)
also have ... =
$(\exists\ k.\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma))$
$\wedge (\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ (suffix\ k\ \sigma)) \models f)$
$))$
$)$
using *interval-idx-bound-1* **by** *blast*
finally show $(\exists\ (l{::}index).\ (intlen\ l) = (Suc\ n) \wedge index\text{-}sequence\ 0\ l \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ \sigma) \models f)$
$)$
$) =$
$(\exists\ k.\ 0 \leq k \wedge k \leq intlen\ \sigma \wedge k > 0 \wedge (sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n\ \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \leq i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ (suffix\ k\ \sigma)) \models f)$
$)$
$)$
$)$

.
qed

lemma *chop-power-eqv-sem*:
$(\exists\ n.\ (\sigma \models (power\text{-}chop\text{-}d\ f\ n))) =$
$((\sigma \models empty) \vee\ (\exists\ n.\ (\sigma \models (f \wedge_i more);\ (power\text{-}chop\text{-}d\ f\ n))))$
by (*metis not0-implies-Suc power-chop-d.power-0 power-chop-d.power-Suc*)

lemma *chopstar-eqv-power-chop-help*:
$(\ \sigma \models power\text{-}chop\text{-}d\ f\ n) =$
$(\exists\ (l{::}index).\ intlen(l) = n \wedge\ index\text{-}sequence\ 0\ l\ \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ (\ \sigma)) \wedge$
$(\forall i.\ (0{\le}i \wedge i{<}\ (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i{+}1))\ (\sigma)) \models f)$
)
)
proof
(*induct n arbitrary*: σ)
case *0*
then show *?case* **using** *index-sequence-def chopstar-help-1* **by** *fastforce*
next
case (*Suc n*)
then show *?case*
proof –
have *1*: $(\sigma \models power\text{-}chop\text{-}d\ f\ (Suc\ n)) = (\sigma \models ((f\ \wedge_i\ more);(power\text{-}chop\text{-}d\ f\ n)))$
by *simp*
have *2*: $(\sigma \models ((f\ \wedge_i\ more);(power\text{-}chop\text{-}d\ f\ n))) =$
$(\exists\ k.\ 0\ {\le}k \wedge k \le intlen\ (\sigma) \wedge k > 0\ \wedge$
$(prefix\ k\ (\sigma) \models f) \wedge$
$(suffix\ k\ (\sigma) \models power\text{-}chop\text{-}d\ f\ n)$
)

by *auto*
have *3*: $(\exists\ k.\ 0\ {\le}k \wedge k \le intlen\ (\sigma) \wedge k > 0\ \wedge$
$(prefix\ k\ (\sigma) \models f) \wedge$
$(suffix\ k\ (\sigma) \models power\text{-}chop\text{-}d\ f\ n)$
$) =$
$(\exists\ k.\ 0\ {\le}k \wedge k \le intlen\ (\sigma) \wedge k > 0\ \wedge$
$(sub\ 0\ k\ (\sigma) \models f) \wedge$
$(suffix\ k\ (\sigma) \models power\text{-}chop\text{-}d\ f\ n)$
)
by (*simp add*: *interval-sub-zero-prefix*)
have *4*: $(\exists\ k.\ 0\ {\le}k \wedge k \le intlen\ (\sigma) \wedge k > 0\ \wedge$
$(sub\ 0\ k\ (\sigma) \models f) \wedge$
$(suffix\ k\ (\sigma) \models power\text{-}chop\text{-}d\ f\ n)$
$) =$
$(\exists\ k.\ 0\ {\le}k \wedge k \le intlen\ (\sigma) \wedge k > 0\ \wedge$
$(sub\ 0\ k\ (\sigma) \models f) \wedge$
$(\exists\ (l{::}index).\ intlen(l) = n \wedge\ index\text{-}sequence\ 0\ l\ \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ (suffix\ k\ \sigma)) \wedge$

$(\forall\, i.\ (0 \le i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ (suffix\ k\ \sigma)) \models f)$
)
)
)
using *Suc.hyps* **by** *auto*
have *5*:
$(\exists\ (l::index).\ (intlen\ l) = (Suc\ n) \wedge index\text{-}sequence\ 0\ l \wedge$
$(nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ l)) \longrightarrow$
$((sub\ (nth\ l\ i)\ (nth\ l\ (i+1))\ \sigma) \models f)$
)
) =
$(\exists\ k.\ 0 \le k \wedge k \le intlen\ \sigma \wedge k > 0 \wedge$
$(sub\ 0\ k\ \sigma \models f) \wedge$
$(\exists\ ls.\ (intlen\ ls) = n \wedge index\text{-}sequence\ 0\ (ls) \wedge$
$(nth\ (ls)\ (intlen\ (ls))) = (intlen\ (suffix\ k\ \sigma)) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ ls)) \longrightarrow$
$((sub\ (nth\ ls\ (i))\ (nth\ ls\ ((i)+1))\ (suffix\ k\ \sigma)) \models f)$
)
)
)
using *chop-power-chain* **by** *simp*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed
qed

lemma *chopstar-eqv-power-chop*:
$(\sigma \models f^\star) = (\exists\ k.\ (\sigma \models power\text{-}chop\text{-}d\ f\ k))$
by (*simp add*: *chopstar-eqv-power-chop-help*)

lemma *ChopstarEqvSem*:
$(\sigma \models f^\star \equiv_i\ empty \vee_i\ (f \wedge_i more);\ f^\star\ )$
using *chopstar-eqv-power-chop*
by (*smt chop-power-eqv-sem iff-defs or-defs semantics-pitl.simps*(*5*))

3.9 Time Reversal

lemma *time-reverse-help-1*:
assumes $index\text{-}sequence\ 0\ l\ \wedge (nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge$
$ls = map\ (\lambda\ x.\ (intlen\ \sigma) - x)\ (intrev\ l) \wedge index\text{-}sequence\ 0\ ls$
shows $(\forall\ i.\ 0 \le i \wedge i < intlen\ ls \longrightarrow$
$(sub\ (nth\ ls\ ((intlen\ ls) - (i+1)))\ ((nth\ ls\ ((intlen\ ls) - i))\ )\ (intrev\ \sigma) \models f^r))$
=
$(\forall\ i.\ 0 \le i \wedge i < intlen\ ls \longrightarrow (sub\ (nth\ ls\ (i))\ ((nth\ ls\ (i+1))\ )\ (intrev\ \sigma) \models f^r))$
by (*smt Suc-diff-Suc Suc-eq-plus1 add-gr-0 diff-diff-cancel diff-less le-add-diff-inverse2 le-simps*(*1*) *zero-less-one zero-order*(*1*))

lemma *TimeReverseSem*:
$(\sigma \models f) \longleftrightarrow ((intrev\ \sigma) \models f^r)$

```
proof
 (induct f arbitrary: σ)
 case false-d
 then show ?case by auto
 next
 case (atom-d x)
 then show ?case by (simp add: interval-intlast-intrev)
 next
 case (implies-d f1 f2)
 then show ?case by simp
 next
 case skip-d
 then show ?case by simp
 next
 case (chop-d f1 f2)
 then show ?case using interval-intrev-prefix interval-intrev-suffix
 by (smt interval-intlen-gr-zero interval-intrev-intlen interval-prefix-length
        interval-rev-rev-ident interval-suffix-length-good order-refl rev-d.simps(5)
        semantics-pitl.simps(5))
 next
 case (chopstar-d f)
 then show ?case
  proof −
   have (σ ⊨  f ⋆) =
          (∃ l. index-sequence 0 l  ∧ (nth l (intlen l)) = (intlen σ) ∧
                   (∀ i. (0≤i ∧ i< (intlen l)) ⟶
                       ((sub (nth l i) (nth l (i+1)) σ) ⊨ f)
                   )
                  )
   by simp
   also have  ... =
            (∃ l. index-sequence 0 l  ∧ (nth l (intlen l)) = (intlen σ) ∧
                   (∀ i. (0≤i ∧ i< (intlen l)) ⟶
                       (intrev (sub (nth l i) (nth l (i+1)) σ) ⊨ f^r)
                   )
                  )
   using chopstar-d.hyps by simp
   also have  ... =
           (∃ l. index-sequence 0 l  ∧ (nth l (intlen l)) = (intlen σ) ∧
                   (∀ i. (0≤i ∧ i< (intlen l)) ⟶
              ( (sub ((intlen σ)− (nth l (i+1))) ((intlen σ) − (nth l i)) (intrev σ)) ⊨ f^r)
                   )
                  )
   using interval-intrev-idx-2 by metis
   also have  ... =
          (∃ l ls. ls = map (λ x. (intlen σ) −x) (intrev l) ∧ index-sequence 0 ls ∧
                index-sequence 0 l  ∧ (nth l (intlen l)) = (intlen σ) ∧
                   (∀ i. (0≤i ∧ i< (intlen l)) ⟶
              ( (sub ((intlen σ)− (nth l (i+1))) ((intlen σ) − (nth l i)) (intrev σ)) ⊨ f^r)
                   )
```

)
using *interval-intrev-idx-7* **by** *auto*
also have ... =
$(\exists\ l\ ls.\ ls = map\ (\lambda\ x.\ (intlen\ \sigma)\ -x)\ (intrev\ l) \wedge$
$index\text{-}sequence\ 0\ ls \wedge index\text{-}sequence\ 0\ l$
$\wedge\ (nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge (nth\ ls\ (intlen\ ls)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ ls)) \longrightarrow$
$(\ (sub\ ((intlen\ \sigma) -\ (nth\ l\ (i+1)))\ ((intlen\ \sigma)\ -\ (nth\ l\ i))\ (intrev\ \sigma)) \models f^r)$
)
)
by (*metis interval-intrev-idx-3*)
also have ... =
$(\exists\ l\ ls.\ ls = map\ (\lambda\ x.\ (intlen\ \sigma)\ -x)\ (intrev\ l) \wedge$
$index\text{-}sequence\ 0\ ls \wedge index\text{-}sequence\ 0\ l$
$\wedge\ (nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge (nth\ ls\ (intlen\ ls)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ ls)) \longrightarrow$
$(\ sub\ (nth\ ls\ ((intlen\ ls) -(i+1)))\ ((nth\ ls\ ((intlen\ ls)\ -i))\ )\ (intrev\ \sigma) \models f^r)$
)
)

using *interval-intrev-idx-9* **by** *metis*
also have ... =
$(\exists\ l\ ls.\ ls = map\ (\lambda\ x.\ (intlen\ \sigma)\ -x)\ (intrev\ l) \wedge$
$index\text{-}sequence\ 0\ ls \wedge index\text{-}sequence\ 0\ l$
$\wedge\ (nth\ l\ (intlen\ l)) = (intlen\ \sigma) \wedge (nth\ ls\ (intlen\ ls)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ ls)) \longrightarrow$
$(\ (sub\ (nth\ ls\ (i))\ ((nth\ ls\ (i+1))\ )\ (intrev\ \sigma)) \models f^r)$
)
)

using *time-reverse-help-1* **by** *metis*
also have ... =
$(\exists\ ls.\ \ index\text{-}sequence\ 0\ ls \wedge$
$(nth\ ls\ (intlen\ ls)) = (intlen\ \sigma) \wedge$
$(\forall\, i.\ (0 \le i \wedge i < (intlen\ ls)) \longrightarrow$
$(\ (sub\ (nth\ ls\ (i))\ ((nth\ ls\ (i+1))\ )\ (intrev\ \sigma)) \models f^r)$
)
)
using *interval-intrev-idx-12* **by** (*smt interval-intrev-idx-3 interval-intrev-idx-7*)
also have ... = $(intrev\ \sigma \models (\ f^r\ ^\star)\ )$ **by** *simp*
also have ... = $(intrev\ \sigma \models (\ f\ ^\star)\ ^r)$ **by** *simp*
finally show $(\sigma \models\ f\ ^\star) = (intrev\ \sigma \models (\ f\ ^\star)\ ^r)$ **.**
qed
qed

end

theory *ITL*
imports

Semantics
begin

4 Axioms and Rules

The ITL axiom and proof rules are introduced (taken from [3]) together with the validity operation. The soundness of the rules and axioms are checked using the lemmas of Semantics.thy.

definition *valid* :: *'a pitl* $\Rightarrow$ *bool* (($\vdash$ -) *5*)
where ($\vdash f$) = ($\forall$ σ. ($\sigma \models f$))

lemma *itl-valid* [*simp*] :
($\vdash f$) = ($\forall$ σ. ($\sigma \models f$))
by (*simp add*: *valid-def*)

lemma *itl-eq*:
($\vdash f \equiv_i g$) = ($\forall$ σ. ($\sigma \models f$) = ($\sigma \models g$))
by *simp*

lemma *EqvReverseReverse*:
$\vdash$ $(f^r)^r \equiv_i f$
using *TimeReverseSem* **by** (*metis interval-rev-rev-ident itl-eq*)

lemma *ReverseEqv*:
($\vdash f$) $\longleftrightarrow$($\vdash f^r$)
by (*metis TimeReverseSem interval-rev-swap valid-def*)

4.1 Rules

lemma *MP* :
assumes $\vdash f \supset_i g$
$\vdash f$
shows $\vdash g$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *BoxGen* :
assumes $\vdash f$
shows $\vdash \Box f$
using *assms* **by** *auto*

lemma *BiGen*:
assumes $\vdash f$
shows $\vdash$ *bi f*
using *assms* **by** *auto*

4.2 Axioms

lemma *ChopAssoc* :
$\vdash$ f ; (g ; h) $\equiv_i$ (f;g);h
using *ChopAssocSem valid-def* **by** *blast*

lemma *OrChopImp* :
$\vdash (f \lor_i g);h \quad \supset_i \ f;h \lor_i g;h$
using *OrChopImpSem valid-def* **by** *blast*

lemma *ChopOrImp* :
$\vdash f;(g \lor_i h) \supset_i \ f;g \lor_i \ f;h$
using *ChopOrImpSem valid-def* **by** *blast*

lemma *EmptyChop* :
$\vdash empty ; f \equiv_i f$
using *EmptyChopSem valid-def* **by** *blast*

lemma *ChopEmpty* :
$\vdash f;empty \equiv_i f$
using *ChopEmptySem valid-def* **by** *blast*

lemma *StateImpBi* :
$\vdash init\ f \supset_i \ bi\ (init\ f)$
using *StateImpBiSem valid-def* **by** *blast*

lemma *NextImpNotNextNot* :
$\vdash \bigcirc f \supset_i \neg_i (\bigcirc \neg_i f)$
using *NextImpNotNextNotSem valid-def* **by** *blast*

lemma *BiBoxChopImpChop* :
$\vdash bi\ (f \supset_i f1) \land_i \Box(g \supset_i g1) \supset_i f;g \supset_i f1;g1$
using *BiBoxChopImpChopSem valid-def* **by** *blast*

lemma *BoxInduct* :
$\vdash \Box\ (f \supset_i wnext\ f) \land_i f \supset_i \Box\ f$
using *BoxInductSem valid-def* **by** *blast*

lemma *ChopstarEqv* :
$\vdash f^\star \equiv_i \ empty \lor_i \ (f \land_i more); f^\star$
using *ChopstarEqvSem valid-def* **by** *blast*

end

theory *Theorems*
imports
ITL
begin

5 ITL theorems

We give the proofs of a list of ITL theorems. These proofs and theorems were from [5].

5.1 Propositional reasoning

This is a list of propositional logic theorems used in the proofs of the ITL theorems.

lemma *itl-prop*:
$\vdash (f \equiv_i f) \equiv_i true_i$
$\vdash (\neg_i\ true_i) \equiv_i false_i$
$\vdash (\neg_i\ false_i) \equiv_i true_i$
$\vdash (\neg_i \neg_i f) \equiv_i f$
$\vdash (\neg_i f \equiv_i f) \equiv_i false_i$
$\vdash (f \equiv_i \neg_i f) \equiv_i false_i$
$\vdash (\neg_i (f \equiv_i g)) \equiv_i (f \equiv_i \neg_i g)$
$\vdash (true_i \equiv_i f) \equiv_i f$
$\vdash (f \equiv_i true_i) \equiv_i f$
$\vdash (true_i \supset_i f) \equiv_i f$
$\vdash (false_i \supset_i f) \equiv_i true_i$
$\vdash (f \supset_i true_i) \equiv_i true_i$
$\vdash (f \supset_i f) \equiv_i true_i$
$\vdash (f \supset_i false_i) \equiv_i \neg_i f$
$\vdash (f \supset_i \neg_i f) \equiv_i \neg_i f$
$\vdash (f \wedge_i true_i) \equiv_i f$
$\vdash (true_i \wedge_i f) \equiv_i f$
$\vdash (f \wedge_i false_i) \equiv_i false_i$
$\vdash (false_i \wedge_i f) \equiv_i false_i$
$\vdash (f \wedge_i f) \equiv_i f$
$\vdash (f \wedge_i \neg_i f) \equiv_i false_i$
$\vdash (\neg_i f \wedge_i f) \equiv_i false_i$
$\vdash (f \vee_i true_i) \equiv_i true_i$
$\vdash (true_i \vee_i f) \equiv_i true_i$
$\vdash (f \vee_i false_i) \equiv_i f$
$\vdash (false_i \vee_i f) \equiv_i f$
$\vdash (f \vee_i f) \equiv_i f$
$\vdash (f \vee_i \neg_i f) \equiv_i true_i$
$\vdash (\neg_i f \vee_i f) \equiv_i true_i$
$(\vdash f \equiv_i f1) = (\vdash f1 \equiv_i f)$
$(\vdash f \equiv_i f1) = (\ (\vdash f \supset_i f1) \wedge\ (\vdash f1 \supset_i f)\ )$
$(\vdash f \supset_i (\ f1 \wedge_i f2)) = (\ (\vdash f \supset_i f1) \wedge (\vdash f \supset_i f2))$
$(\vdash f \equiv_i f1) = (\vdash \neg_i f \equiv_i \neg_i f1)$
$(\vdash \neg_i (\ f \vee_i f1)) = (\vdash \neg_i f \wedge_i \neg_i f1)$
$(\vdash\ (\ f \supset_i f1)) = (\vdash\ (\ \neg_i f \vee_i f1)\ )$
by *auto*

lemma *prop01*:
assumes $\vdash f \equiv_i g$
shows $\vdash \neg_i f \equiv_i \neg_i g$
using *assms itl-prop*(*33*) **by** *blast*

lemma *prop02*:
assumes $\vdash f \supset_i g$
$\vdash g \supset_i h$
shows $\vdash f \supset_i h$

using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop03*:
 assumes $\vdash f \equiv_i g$
 $\vdash g \equiv_i h$
 shows $\vdash f \equiv_i h$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop04*:
 $\vdash \neg_i (f \wedge_i g \wedge_i \neg_i h) \equiv_i (f \supset_i (g \supset_i h))$
by *auto*

lemma *prop05*:
 assumes $\vdash f \equiv_i g$
 shows $\vdash h \wedge_i f \equiv_i h \wedge_i g$
using *assms* **by** *auto*

lemma *prop06*:
 assumes $\vdash f \equiv_i g$
 shows $\vdash f \wedge_i h \equiv_i g \wedge_i h$
using *assms* **by** *auto*

lemma *prop07*:
 assumes $\vdash f \equiv_i f1$
 $\vdash g \equiv_i g1$
 shows $\vdash (if_i\ \ (init\ w)\ \ then\ f\ \ else\ g) \equiv_i (if_i\ \ (init\ w)\ \ then\ f1\ \ else\ g1)$
using *assms*(*1*) *assms*(*2*) **by** *simp*

lemma *prop08*:
 assumes $\vdash f \wedge_i g \supset_i h$
 shows $\vdash f \wedge_i g \wedge_i f1 \supset_i h$
using *assms* **by** *auto*

lemma *prop09*:
 assumes $\vdash f \wedge_i g \wedge_i h \supset_i f1$
 $\vdash f \wedge_i h \wedge_i g \supset_i \neg_i f1$
 shows $\vdash \neg_i (f \wedge_i h \wedge_i g)$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop10*:
 assumes $\vdash f \wedge_i f1 \supset_i h$
 shows $\vdash f \wedge_i g \wedge_i f1 \supset_i h$
using *assms* **by** *auto*

lemma *prop11*:
 $\vdash (if_i\ \ g\ \ then\ \ f\ \ else\ f1) \supset_i ((g \supset_i f) \wedge_i (\neg_i g \supset_i f1))$
by *simp*

lemma *prop12*:
 assumes $\vdash f \supset_i g$

```
 shows    ⊢ h ∧ᵢ f ⊃ᵢ h ∧ᵢ g
using assms by auto

lemma prop13:
 assumes ⊢ f ⊃ᵢ ¬ᵢ g ∨ᵢ h
 shows   ⊢ g ∧ᵢ f ⊃ᵢ h
using assms by auto

lemma prop14:
 assumes ⊢ f ≡ᵢ g ∨ᵢ h
         ⊢ h ⊃ᵢ h1
 shows   ⊢ f ⊃ᵢ g  ∨ᵢ h1
using assms(1) assms(2) by auto

lemma prop15:
 assumes ⊢ f ≡ᵢ g
         ⊢ h ⊃ᵢ g
 shows   ⊢ h ⊃ᵢ f
using assms(1) assms(2) by auto

lemma prop16:
 assumes ⊢ f ⊃ᵢ g ∨ᵢ h
         ⊢ g ⊃ᵢ h1 ∨ᵢ h
 shows   ⊢ f ⊃ᵢ h1 ∨ᵢ h
using assms(1) assms(2) by auto

lemma prop17:
 assumes ⊢ f ⊃ᵢ g
         ⊢ f1 ⊃ᵢ g
  shows  ⊢ f ∨ᵢ f1  ⊃ᵢ g
using assms(1) assms(2) by auto


lemma prop18:
 assumes ⊢ g ∧ᵢ h ⊃ᵢ h1
         ⊢ f ≡ᵢ g
 shows   ⊢ f ∧ᵢ h ⊃ᵢ h1
using assms(1) assms(2) by auto

lemma prop19:
  assumes ⊢ f ≡ᵢ g ∨ᵢ h
  shows    ⊢ h ⊃ᵢ f
using assms by auto


lemma prop20:
 assumes ⊢ f ≡ᵢ g ∨ᵢ h
 shows   ⊢ ¬ᵢ f ≡ᵢ ¬ᵢg ∧ᵢ ¬ᵢ h
using assms by auto
```

lemma *prop21*:
 assumes $\vdash f \equiv_i h$
 $\vdash f \equiv_i h1$
 shows $\vdash h1 \equiv_i h$
using *assms*(1) *assms*(2) **by** *auto*

lemma *prop22*:
 assumes $\vdash f \equiv_i h$
 $\vdash g \equiv_i g1$
 shows $\vdash f \wedge_i g \supset_i h \wedge_i g1$
using *assms*(1) *assms*(2) **by** *auto*

lemma *prop23*:
 assumes $\vdash f \equiv_i g \vee_i h$
 shows $\vdash f \wedge_i f1 \equiv_i (g \wedge_i f1) \vee_i (h \wedge_i f1)$
 using *assms* **by** *auto*

lemma *prop24*:
 assumes $\vdash f \equiv_i g \vee_i (h \wedge_i h1)$
 $\vdash g \equiv_i g1$
 $\vdash h \equiv_i h2 \wedge_i h3$
 $\vdash h3 \wedge_i h1 \equiv_i h4$
 shows $\vdash f \equiv_i g1 \vee_i (h2 \wedge_i h4)$
using *assms*(1) *assms*(2) *assms*(3) *assms*(4) **by** *auto*

lemma *prop25*:
 assumes $\vdash f \supset_i false_i$
 shows $\vdash g \vee_i f \equiv_i g$
using *assms* **by** *auto*

lemma *prop26*:
 assumes $\vdash f \supset_i g$
 shows $\vdash f \supset_i h \vee_i g$
using *assms* **by** *auto*

lemma *prop27*:
 assumes $\vdash f \supset_i g$
 shows $\vdash \neg_i g \supset_i \neg_i f$
using *assms* **by** *auto*

lemma *prop28*:
 assumes $\vdash f \equiv_i g \vee_i h$
 $\vdash h \equiv_i h1$
 shows $\vdash f \equiv_i g \vee_i h1$
using *assms*(1) *assms*(2) **by** *auto*

lemma *prop29*:
 assumes $\vdash f \supset_i g \vee_i h$
 shows $\vdash f \wedge_i \neg_i g \supset_i h$
using *assms* **by** *auto*

lemma *prop30*:
assumes $\vdash f0 \supset_i g$
$\vdash f1 \supset_i g$
shows $\vdash f0 \vee_i f1 \supset_i g$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop31*:
$\vdash (f \supset_i (g \equiv_i h)) \equiv_i ((f \wedge_i g) \equiv_i (f \wedge_i h))$
by *auto*

lemma *prop32*:
assumes $\vdash f \wedge_i g \supset_i h$
shows $\vdash g \supset_i (f \supset_i h)$
using *assms* **by** *auto*

lemma *prop33*:
assumes $\vdash f0 \wedge_i g \supset_i h$
$\vdash f1 \wedge_i g \supset_i h$
shows $\vdash (f0 \vee_i f1) \wedge_i g \supset_i h$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop34*:
assumes $\vdash f \wedge_i g \supset_i h$
$\vdash f$
shows $\vdash g \supset_i h$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop35*:
assumes $\vdash f \supset_i g \vee_i h$
$\vdash h \supset_i h1$
shows $\vdash f \supset_i g \vee_i h1$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop36*:
assumes $\vdash f \wedge_i g \supset_i h$
shows $\vdash f \supset_i (g \supset_i h)$
using *assms* **by** *auto*

lemma *prop37*:
assumes $\vdash f1 \supset_i f$
$\vdash f \wedge_i f1 \wedge_i g \supset_i h$
shows $\vdash f1 \wedge_i g \supset_i h$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop38*:
assumes $\vdash f \supset_i g$
shows $\vdash f \equiv_i f \wedge_i g$
using *assms* **by** *auto*

lemma *prop39*:
assumes $\vdash f \equiv_i f1$
$\vdash g \equiv_i g1$
shows $\vdash (f \supset_i g) \equiv_i (f1 \supset_i g1)$
using *assms*(*1*) *assms*(*2*) **by** *auto*

lemma *prop40*:
assumes $\vdash f \equiv_i f1$
$\vdash g \equiv_i g1$
$\vdash h \equiv_i h1$
shows $\vdash (f \equiv_i g \wedge_i h) \equiv_i (f1 \equiv_i g1 \wedge_i h1)$
using *assms*(*1*) *assms*(*2*) *assms*(*3*) **by** *auto*

5.2 State formulas

The *init* operator denotes state formulas, i.e., ITL formula that only constrain the first state of an interval.

lemma *Initprop* :
$\vdash ((init\ f) \wedge_i (init\ g)) \equiv_i init(f \wedge_i g)$
$\vdash (\neg_i (init\ f)) \equiv_i init\ (\neg_i f)$
$\vdash ((init\ f) \vee_i (init\ g)) \equiv_i init\ (f \vee_i g)$
$\vdash init\ true_i$
by *auto*

lemma *Finprop* :
$\vdash (true_i;(f \wedge_i empty)) \wedge_i (true_i;(g \wedge_i empty)) \equiv_i (true_i;(f \wedge_i g \wedge_i empty))$
$\vdash (true_i;(f \wedge_i empty)) \vee_i (true_i;(g \wedge_i empty)) \equiv_i (true_i;((f \vee_i g) \wedge_i empty))$
$\vdash (true_i;(true_i \wedge_i empty))$
$\vdash \neg_i (true_i;(f \wedge_i empty)) \equiv_i (true_i;(\neg_i f \wedge_i empty))$
apply *simp-all*
apply *auto*[*1*]
apply *auto*[*2*]
using *dual-order.order-iff-strict* **by** *fastforce*

5.3 Basic Theorems

lemma *BiChopImpChop* :
$\vdash bi\ (f \supset_i f1) \supset_i f;g \supset_i f1;g$
proof −
have *1*: $\vdash g \supset_i g$ **by** *auto*
hence *2*: $\vdash \Box\ (g \supset_i g)$ **by** (*rule BoxGen*)
have *3*: $\vdash bi\ (f \supset_i f1) \wedge_i \Box(g \supset_i g) \supset_i f;g \supset_i f1;g$ **by** (*rule BiBoxChopImpChop*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *AndChopA*:
$\vdash (f \wedge_i f1);g \supset_i f;g$
proof −
have *1*: $\vdash f \wedge_i f1 \supset_i f$ **by** *auto*
hence *2*: $\vdash bi\ (f \wedge_i f1 \supset_i f)$ **by** (*rule BiGen*)
have *3*: $\vdash bi\ (f \wedge_i f1 \supset_i f) \supset_i (f \wedge_i f1);g \supset_i f;g$ **by** (*rule BiChopImpChop*)

from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *AndChopB*:
$\vdash (f \wedge_i f1);g \supset_i f1;g$
proof −
have *1*: $\vdash f \wedge_i f1 \supset_i f1$ **by** *auto*
hence *2*: $\vdash$ *bi* $(f \wedge_i f1 \supset_i f1)$ **by** (*rule BiGen*)
have *3*: $\vdash$ *bi* $(f \wedge_i f1 \supset_i f1) \supset_i (f \wedge_i f1);g \supset_i f1;g$ **by** (*rule BiChopImpChop*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *NextChop*:
$\vdash (\bigcirc f);g \equiv_i \bigcirc(f;g)$
proof −
have *1*: $\vdash$ *skip*;$(f;g) \equiv_i ($*skip*;$f);g$ **by** (*rule ChopAssoc*)
show *?thesis* **by** (*metis 1 itl-prop*(*30*) *next-d-def*)
qed

lemma *BoxChopImpChop* :
$\vdash \Box (g \supset_i g1) \supset_i f;g \supset_i f;g1$
proof −
have *1*: $\vdash g \supset_i g$ **by** *auto*
hence *2*: $\vdash$ *bi* $(g \supset_i g)$ **by** (*rule BiGen*)
have *3*: $\vdash$ *bi* $(f \supset_i f) \wedge_i \Box(g \supset_i g1) \supset_i f;g \supset_i f;g1$ **by** (*rule BiBoxChopImpChop*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *LeftChopImpChop*:
assumes $\vdash f \supset_i f1$
shows $\vdash f;g \supset_i f1;g$
proof −
have *1*: $\vdash f \supset_i f1$ **using** *assms* **by** *auto*
hence *2*: $\vdash$ *bi* $(f \supset_i f1)$ **by** (*rule BiGen*)
have *3*: $\vdash$ *bi* $(f \supset_i f1) \supset_i f;g \supset_i f1;g$ **by** (*rule BiChopImpChop*)
from *2 3* **show** *?thesis* **using** *MP* **by** *blast*
qed

lemma *RightChopImpChop*:
assumes $\vdash g \supset_i g1$
shows $\vdash f;g \supset_i f;g1$
proof −
have *1*: $\vdash g \supset_i g1$ **using** *assms* **by** *auto*
hence *2*: $\vdash \Box (g \supset_i g1)$ **by** (*rule BoxGen*)
have *3*: $\vdash \Box (g \supset_i g1) \supset_i f;g \supset_i f;g1$ **by** (*rule BoxChopImpChop*)
from *2 3* **show** *?thesis* **using** *MP* **by** *blast*
qed

lemma *RightChopEqvChop*:
assumes $\vdash g \equiv_i g1$

shows $\vdash f;g \equiv_i f;g1$
proof $-$
have *1*: $\vdash g \equiv_i g1$ **using** *assms* **by** *auto*
have *2*: $(\vdash g \supset_i g1) \Longrightarrow (\vdash f;g \supset_i f;g1)$ **by** (*rule RightChopImpChop*)
have *3*: $(\vdash g1 \supset_i g) \Longrightarrow (\vdash f;g1 \supset_i f;g)$ **by** (*rule RightChopImpChop*)
from *1 2 3* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *ChopOrEqv*:
$\vdash f;(g \vee_i g1) \equiv_i f;g \vee_i f;g1$
proof $-$
have *1*: $\vdash g \supset_i g \vee_i g1$ **by** *auto*
hence *2*: $\vdash f;g \supset_i f;(g \vee_i g1)$ **by** (*rule RightChopImpChop*)
have *3*: $\vdash g1 \supset_i g \vee_i g1$ **by** *auto*
hence *4*: $\vdash f;g1 \supset_i f;(g \vee_i g1)$ **by** (*rule RightChopImpChop*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *OrChopEqv*:
$\vdash (f \vee_i f1);g \equiv_i f;g \vee_i f1;g$
proof $-$
have *1*: $\vdash f \supset_i f \vee_i f1$ **by** *auto*
hence *2*: $\vdash f;g \supset_i (f \vee_i f1);g$ **by** (*rule LeftChopImpChop*)
have *3*: $\vdash f1 \supset_i f \vee_i f1$ **by** *auto*
hence *4*: $\vdash f1;g \supset_i (f \vee_i f1);g$ **by** (*rule LeftChopImpChop*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *OrChopImpRule*:
assumes $\vdash f \supset_i f1 \vee_i f2$
shows $\vdash f;g \supset_i (f1;g) \vee_i (f2;g)$
proof $-$
have *1*: $\vdash f \supset_i f1 \vee_i f2$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;g \supset_i (f1 \vee_i f2);g$ **by** (*rule LeftChopImpChop*)
have *3*: $\vdash (f1 \vee_i f2);\, g \equiv_i f1;g \vee_i f2;g$ **by** (*rule OrChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *LeftChopEqvChop*:
assumes $\vdash f \equiv_i f1$
shows $\vdash f;g \equiv_i f1;g$
proof $-$
have *1*: $\vdash f \equiv_i f1$ **using** *assms* **by** *auto*
hence *2*: $\vdash f \supset_i f1$ **by** *auto*
hence *3*: $\vdash f;g \supset_i f1;g$ **by** (*rule LeftChopImpChop*)
from *1* **have** $\vdash f1 \supset_i f$ **by** *auto*
hence *4*: $\vdash f1;g \supset_i f;g$ **by** (*rule LeftChopImpChop*)
from *3 4* **show** *?thesis* **by** *auto*
qed

lemma *OrChopEqvRule*:
assumes $\vdash f \equiv_i f1 \vee_i f2$
shows $\vdash f;g \equiv_i (f1;g) \vee_i (f2;g)$
proof –
have *1*: $\vdash f \equiv_i f1 \vee_i f2$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;g \equiv_i (f1 \vee_i f2);g$ **by** (*rule LeftChopEqvChop*)
have *3*: $\vdash (f1 \vee_i f2);g \equiv_i f1;g \vee_i f2;g$ **by** (*rule OrChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *NextImpNext*:
assumes $\vdash f \supset_i g$
shows $\vdash \bigcirc f \supset_i \bigcirc g$
proof –
have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \Box (f \supset_i g)$ **by** (*rule BoxGen*)
have *3*: $\vdash \Box (f \supset_i g) \supset_i (skip;f) \supset_i (skip;g)$ **by** (*rule BoxChopImpChop*)
have *4*: $\vdash (skip;f) \supset_i (skip;g)$ **by** (*metis 2 3 MP*)
from *4* **show** *?thesis* **by** (*metis next-d-def*)
qed

lemma *ChopOrImpRule*:
assumes $\vdash g \supset_i g1 \vee_i g2$
shows $\vdash f;g \supset_i (f;g1) \vee_i (f;g2)$
proof –
have *1*: $\vdash g \supset_i g1 \vee_i g2$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;g \supset_i f;(g1 \vee_i g2)$ **by** (*rule RightChopImpChop*)
have *3*: $\vdash f;(g1 \vee_i g2) \equiv_i f;g1 \vee_i f;g2$ **by** (*rule ChopOrEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *NextImpDist*:
$\vdash \bigcirc (f \supset_i g) \supset_i \bigcirc f \supset_i \bigcirc g$
proof –
have *1*: $\vdash \neg_i (f \supset_i g) \equiv_i f \wedge_i \neg_i g$ **by** *auto*
hence *2*: $\vdash skip;\neg_i (f \supset_i g) \equiv_i skip;(f \wedge_i \neg_i g)$ **by** (*rule RightChopEqvChop*)
have *3*: $\vdash f \supset_i g \vee_i (f \wedge_i \neg_i g)$ **by** *auto*
hence *4*: $\vdash skip;f \supset_i (skip;g) \vee_i (skip;(f \wedge_i \neg_i g))$ **by** (*rule ChopOrImpRule*)
hence *5*: $\vdash \neg_i (skip;(f \wedge_i \neg_i g)) \supset_i (skip;f) \supset_i (skip;g)$ **by** *auto*
have *6*: $\vdash \neg_i (skip;\neg_i(f \supset_i g)) \supset_i (skip;f) \supset_i (skip;g)$ **by** *auto*
hence *7*: $\vdash \neg_i (\bigcirc\neg_i(f \supset_i g)) \supset_i (\bigcirc f) \supset_i (\bigcirc g)$ **by** (*simp add*: *next-d-def*)
have *8*: $\vdash \bigcirc(f \supset_i g) \supset_i \neg_i (\bigcirc\neg_i(f \supset_i g))$ **by** (*rule NextImpNotNextNot*)
from *7 8* **show** *?thesis* **by** *auto*
qed

lemma *ChopImpDiamond*:
$\vdash f;g \supset_i \Diamond g$
proof –
have *1*: $\vdash f \supset_i true_i$ **by** *auto*

hence *2*: $\vdash f;g \supset_i true_i;g$ **by** (*rule LeftChopImpChop*)
from *2* **show** *?thesis* **by** *auto*
qed

lemma *NowImpDiamond*:
$\vdash f \supset_i \diamond f$
proof −
have *1*: $\vdash empty;f \equiv_i f$ **by** (*rule EmptyChop*)
have *2*: $\vdash empty \supset_i true_i$ **by** *auto*
hence *3*: $\vdash empty;f \supset_i true_i;f$ **by** (*rule LeftChopImpChop*)
have *4*: $\vdash f \supset_i true_i;f$ **using** *1 3* **by** *auto*
from *4* **show** *?thesis* **by** *auto*
qed

lemma *BoxElim*:
$\vdash \Box f \supset_i f$
proof −
have *1*: $\vdash \neg_i f \supset_i \diamond \neg_i f$ **by** (*rule NowImpDiamond*)
hence *2*: $\vdash \neg_i (\diamond \neg_i f) \supset_i f$ **by** *auto*
from *2* **show** *?thesis* **by** (*metis always-d-def*)
qed

lemma *NextDiamondImpDiamond*:
$\vdash \bigcirc (\diamond f) \supset_i \diamond f$
proof −
have *1*: $\vdash skip;(true_i;f) \equiv_i (skip;true_i);f$ **by** (*rule ChopAssoc*)
hence *2*: $\vdash (skip;true_i);f \equiv_i skip;(true_i;f)$ **by** *auto*
hence *3*: $\vdash (skip;true_i);f \equiv_i \bigcirc(\diamond f)$ **by** (*simp add*: *next-d-def*)
have *4*: $\vdash (skip;true_i);f \supset_i \diamond f$ **by** (*rule ChopImpDiamond*)
from *3 4* **show** *?thesis* **by** *auto*
qed

lemma *BoxImpNowAndWeakNext*:
$\vdash \Box f \supset_i (f \wedge_i wnext\ (\Box f))$
proof −
have *1*: $\vdash \neg_i f \supset_i \diamond \neg_i f$ **by** (*rule NowImpDiamond*)
hence *2*: $\vdash \neg_i (\diamond \neg_i f) \supset_i f$ **by** *auto*
hence *3*: $\vdash \Box f \supset_i f$ **by** (*metis always-d-def*)
have *4*: $\vdash \bigcirc (\diamond \neg_i f) \supset_i \diamond (\neg_i f)$ **by** (*rule NextDiamondImpDiamond*)
have *5*: $\vdash \neg_i \neg_i (\diamond \neg_i f) \supset_i \diamond(\neg_i f)$ **by** *auto*
hence *6*: $\vdash \bigcirc (\neg_i \neg_i (\diamond \neg_i f)) \supset_i \bigcirc (\diamond(\neg_i f))$ **by** (*rule NextImpNext*)
have *7*: $\vdash \bigcirc (\neg_i \neg_i (\diamond \neg_i f)) \supset_i \diamond (\neg_i f)$ **using** *4 6* **by** *auto*
hence *8*: $\vdash \bigcirc (\neg_i(\Box f)) \supset_i \diamond (\neg_i f)$ **by** (*simp add*: *always-d-def*)
hence *9*: $\vdash \neg_i (\diamond (\neg_i f)) \supset_i \neg_i (\bigcirc (\neg_i(\Box f)))$ **by** *auto*
hence *10*: $\vdash \Box f \supset_i wnext\ (\Box f)$ **by** (*simp add*: *always-d-def wnext-d-def*)
from *3 10* **show** *?thesis* **using** *itl-prop*(*32*) **by** *blast*
qed

lemma *BoxImpBoxRule*:

assumes $\vdash f \supset_i g$
shows $\vdash \Box f \supset_i \Box g$
proof –
have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i g \supset_i \neg_i f$ **by** *auto*
hence *3*: $\vdash \Box(\neg_i g \supset_i \neg_i f)$ **by** (*rule BoxGen*)
have *4*: $\vdash \Box(\neg_i g \supset_i \neg_i f) \supset_i (true_i;\neg_i g) \supset_i (true_i;\neg_i f)$ **by** (*rule BoxChopImpChop*)
have *5*: $\vdash (true_i;\neg_i g) \supset_i (true_i;\neg_i f)$ **using** *3 4 MP* **by** *auto*
hence *6*: $\vdash \Diamond \neg_i g \supset_i \Diamond\neg_i f$ **by** (*simp add*: *sometimes-d-def*)
hence *7*: $\vdash \neg_i (\Diamond\neg_i f) \supset_i \neg_i(\Diamond \neg_i g)$ **by** *auto*
from *7* **show** *?thesis* **by** (*simp add*: *always-d-def*)
qed

lemma *BoxImpDist*:
$\vdash \Box(f \supset_i g) \supset_i \Box f \supset_i \Box g$
proof –
have *1*: $\vdash (f \supset_i g) \supset_i (\neg_i g \supset_i \neg_i f)$ **by** *auto*
hence *2*: $\vdash \Box(f \supset_i g) \supset_i \Box(\neg_i g \supset_i \neg_i f)$ **by** (*rule BoxImpBoxRule*)
have *3*: $\vdash \Box(\neg_i g \supset_i \neg_i f) \supset_i (true_i; \neg_i g) \supset_i (true_i; \neg_i f)$ **by** (*rule BoxChopImpChop*)
have *4*: $\vdash \Box(f \supset_i g) \supset_i (true_i; \neg_i g) \supset_i (true_i; \neg_i f)$ **using** *2 3 prop02* **by** *blast*
hence *5*: $\vdash \Box(f \supset_i g) \supset_i \Diamond\neg_i g \supset_i \Diamond\neg_i f$ **by** (*simp add*: *sometimes-d-def*)
hence *6*: $\vdash \Box(f \supset_i g) \supset_i \neg_i(\Diamond\neg_i f) \supset_i \neg_i(\Diamond\neg_i g)$ **by** *auto*
from *6* **show** *?thesis* **by** (*simp add*: *always-d-def*)
qed

lemma *DiamondEmpty*:
$\vdash \Diamond\, empty$
proof –
have *1*: $\vdash true_i$ **by** *auto*
have *2*: $\vdash true_i;\ empty \equiv_i true_i$ **by** (*rule ChopEmpty*)
have *3*: $\vdash true_i;\ empty$ **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *sometimes-d-def*)
qed

lemma *NextEqvNext*:
assumes $\vdash f \equiv_i g$
shows $\vdash \bigcirc f \equiv_i \bigcirc g$
proof –
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash skip;f \equiv_i skip;g$ **by** (*rule RightChopEqvChop*)
from *1* **show** *?thesis* **by** (*simp add*: *next-d-def*)
qed

lemma *NextAndNextImpNextRule*:
assumes $\vdash (f \wedge_i g) \supset_i h$
shows $\vdash (\bigcirc f \wedge_i \bigcirc g) \supset_i \bigcirc h$
using *assms* **by** *auto*

lemma *NextAndNextEqvNextRule*:
assumes $\vdash f \wedge_i g \equiv_i h$

shows $\vdash \bigcirc f \wedge_i \bigcirc g \equiv_i \bigcirc h$
using *assms* **by** *auto*

lemma *WeakNextEqvWeakNext*:
assumes $\vdash f \equiv_i g$
shows $\vdash wnext\ \ f \equiv_i wnext\ \ g$
using *assms* **by** *auto*

lemma *DiamondImpDiamond*:
assumes $\vdash f \supset_i g$
shows $\vdash \diamond f \supset_i \diamond g$
using *assms* **by** *auto*

lemma *DiamondEqvDiamond*:
assumes $\vdash f \equiv_i g$
shows $\vdash \diamond f \equiv_i \diamond g$
using *assms* **by** *auto*

lemma *BoxEqvBox*:
assumes $\vdash f \equiv_i g$
shows $\vdash \Box f \equiv_i \Box g$
using *assms* **by** *auto*

lemma *BoxAndBoxImpBoxRule*:
assumes $\vdash f \wedge_i g \supset_i h$
shows $\vdash \Box f \wedge_i \Box g \supset_i \Box h$
using *assms* **by** *auto*

lemma *BoxAndBoxEqvBoxRule*:
assumes $\vdash f \wedge_i g \equiv_i h$
shows $\vdash \Box f \wedge_i \Box g \equiv_i \Box h$
using *assms* **by** *auto*

lemma *ImpBoxRule*:
assumes $\vdash\ \ f \supset_i g$
shows $\vdash \Box f \supset_i \Box g$
using *assms* **by** *auto*

lemma *BoxIntro*:
assumes $\vdash\ \ f \supset_i g$
$\vdash\ \ more\ \wedge_i f \supset_i \bigcirc f$
shows $\vdash f \supset_i \Box g$
proof $-$
have *1*: $\vdash\ \ more\ \wedge_i f \supset_i \bigcirc f$ **using** *assms* **by** *auto*
hence *2*: $\vdash f \supset_i (empty \vee_i \bigcirc f)$ **by** *auto*
hence *3*: $\vdash f \supset_i wnext\ f$ **by** *auto*
hence *4*: $\vdash \Box(f \supset_i wnext\ f)$ **by** (*rule BoxGen*)
have *5*: $\vdash (\Box\ (f \supset_i wnext\ f)) \wedge_i f \supset_i \Box f$ **by** (*rule BoxInduct*)
hence *6*: $\vdash (\Box\ (f \supset_i wnext\ f)) \supset_i (f \supset_i \Box f)$ **using** *prop36* **by** *blast*
have *7*: $\vdash f \supset_i \Box f$ **using** *4 6 MP* **by** *blast*

have *8*: $\vdash \Box f \supset_i f$ **by** (*rule BoxElim*)
have *9*: $\vdash f \equiv_i \Box f$ **using** *7 8 itl-prop*(*31*) **by** *blast*
have *10*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
hence *11*: $\vdash \Box f \supset_i \Box g$ **by** (*rule ImpBoxRule*)
from *7 9 11* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *NextLoop*:
assumes $\vdash f \supset_i \bigcirc f$
shows $\vdash \neg_i f$
proof −
have *1*: $\vdash f \supset_i \bigcirc f$ **using** *assms* **by** *auto*
hence *2*: $\vdash f \supset_i (more \wedge_i wnext\ f)$ **by** *auto*
hence *3*: $\vdash f \supset_i wnext\ f$ **by** *auto*
hence *4*: $\vdash \Box(f \supset_i wnext\ f)$ **by** (*rule BoxGen*)
have *5*: $\vdash \Box (f \supset_i wnext\ f) \wedge_i f \supset_i \Box f$ **by** (*rule BoxInduct*)
hence *6*: $\vdash \Box (f \supset_i wnext\ f) \supset_i (f \supset_i \Box f)$ **using** *prop36* **by** *blast*
have *7*: $\vdash f \supset_i \Box f$ **using** *4 6 MP* **by** *blast*
have *8*: $\vdash \Box f \supset_i f$ **by** (*rule BoxElim*)
have *9*: $\vdash f \equiv_i \Box f$ **using** *7 8 itl-prop*(*31*) **by** *blast*
have *10*: $\vdash f \supset_i more$ **using** *2* **by** *auto*
hence *11*: $\vdash \Box f \supset_i \Box more$ **by** (*rule ImpBoxRule*)
have *12*: $\vdash \neg_i(\Box more)$ **by** *auto*
from *7 9 11 12* **show** *?thesis* **by** (*metis not-d-def prop02*)
qed

lemma *WnextEqvEmptyOrNext*:
$\vdash wnext\ f \equiv_i empty \vee_i \bigcirc f$
by *auto*

lemma *NotEmptyAndNext*:
$\vdash \neg_i(empty \wedge_i \bigcirc f)$
by *auto*

lemma *BoxEqvAndWnextBox*:
$\vdash \Box f \equiv_i f \wedge_i wnext\ (\Box f)$
proof −
have *1*: $\vdash \Box f \supset_i f \wedge_i wnext\ (\Box f)$
    using *BoxImpNowAndWeakNext* **by** *blast*
have *2*: $\vdash f \wedge_i wnext\ (\Box f) \supset_i f$
    by *simp*
have *3*: $\vdash more \wedge_i (f \wedge_i wnext\ (\Box f)) \supset_i \bigcirc (f \wedge_i wnext\ (\Box f))$
    by (*metis 1 NextImpNext WnextEqvEmptyOrNext empty-d-def prop10 prop13 prop14*)
have *4*: $\vdash f \wedge_i wnext\ (\Box f) \supset_i \Box f$
    using *2 3 BoxIntro* **by** *blast*
from *1 4* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *BoxEqvAndEmptyOrNextBox*:
$\vdash \Box f \equiv_i f \wedge_i (empty \vee_i \bigcirc(\Box f))$

using *BoxEqvAndWnextBox WnextEqvEmptyOrNext* **using** *prop03 prop05* **by** *blast*

lemma *BoxEqvBoxBox*:
$\vdash \Box f \equiv_i \Box\ (\Box\ f)$
by *auto*

lemma *BoxBoxImpBox*:
$\vdash \Box(\Box h) \supset_i \Box\ h$
using *BoxEqvBoxBox itl-prop*(*31*) **by** *blast*

lemma *BoxImpBoxBox*:
$\vdash \Box\ h\ \supset_i\ \Box(\Box h)$
by *simp*

lemma *DiamondIntro*:
assumes $\vdash (f \wedge_i \neg_i\ g) \supset_i \bigcirc f$
shows $\vdash f \supset_i \Diamond\ g$
proof −
have *1*: $\vdash f \wedge_i \neg_i\ g \supset_i \bigcirc f$
  using *assms* **by** *auto*
hence *2*: $\vdash f \wedge_i \neg_i\ g \wedge_i (\Box \neg_i g) \supset_i (\bigcirc f) \wedge_i (\Box \neg_i g)$
  by *auto*
have *3*: $\vdash (\Box \neg_i g) \supset_i \neg_i\ g$
  by (*rule BoxElim*)
hence *4*: $\vdash \Box \neg_i g \equiv_i (\Box \neg_i g) \wedge_i \neg_i\ g$
  using *BoxImpBoxBox BoxBoxImpBox itl-prop*(*31*) *itl-prop*(*32*) *prop02 prop26 prop29* **by** *blast*
have *5*: $\vdash f \wedge_i (\Box \neg_i g) \supset_i \bigcirc f \wedge_i \Box \neg_i g$
  using *2 4* **by** *auto*
have *6*: $\vdash \Box \neg_i g \equiv_i (\neg_i g) \wedge_i$ *wnext*$(\Box \neg_i g)$
  using *BoxEqvAndWnextBox* **by** *blast*
have *7*: $\vdash \bigcirc f \wedge_i \Box \neg_i g \supset_i\ \bigcirc f \wedge_i$ *wnext*$(\Box \neg_i g)$
  using *6* **by** *auto*
have *8*: $\vdash f \wedge_i (\Box \neg_i g) \supset_i \bigcirc f \wedge_i$ *wnext*$(\Box \neg_i g)$
  using *5 7* **by** *auto*
hence *9*: $\vdash f \wedge_i (\Box \neg_i g) \supset_i$ *more* $\wedge_i$ *wnext f* $\wedge_i$ *wnext*$(\Box \neg_i g)$
  by *auto*
hence *10*: $\vdash f \wedge_i (\Box \neg_i g) \supset_i$ *wnext f* $\wedge_i$ *wnext*$(\Box \neg_i g)$
  by *auto*
hence *11*: $\vdash f \wedge_i (\Box \neg_i g) \supset_i$ *wnext* $(f \wedge_i \Box \neg_i g)$
  by *auto*
hence *12*: $\vdash \Box(f \wedge_i (\Box \neg_i g) \supset_i$ *wnext* $(f \wedge_i \Box \neg_i g))$
  by (*rule BoxGen*)
have *13*: $\vdash \Box(f \wedge_i (\Box \neg_i g) \supset_i$ *wnext* $(f \wedge_i \Box \neg_i g)) \wedge_i f \wedge_i (\Box \neg_i g) \supset_i \Box(f \wedge_i (\Box \neg_i g))$
  by (*rule BoxInduct*)
hence *14*: $\vdash \Box(f \wedge_i (\Box \neg_i g) \supset_i$ *wnext* $(f \wedge_i \Box \neg_i g)) \supset_i ((f \wedge_i (\Box \neg_i g)) \supset_i \Box(f \wedge_i (\Box \neg_i g)))$
  using *prop36* **by** *blast*
have *15*: $\vdash ((f \wedge_i (\Box \neg_i g)) \supset_i \Box(f \wedge_i (\Box \neg_i g)))$
  using *12 14 MP* **by** *blast*
have *16*: $\vdash \Box(f \wedge_i (\Box \neg_i g)) \supset_i (f \wedge_i (\Box \neg_i g))$
  by (*rule BoxElim*)

have *17*: $\vdash \Box(f \wedge_i (\Box \neg_i g)) \equiv_i (f \wedge_i (\Box \neg_i g))$
using *16 15 itl-prop*(*31*) **by** *blast*
have *18*: $\vdash (f \wedge_i (\Box \neg_i g)) \supset_i$ *more*
using *9* **by** *auto*
hence *19*: $\vdash \Box(f \wedge_i (\Box \neg_i g)) \supset_i \Box$ *more*
by (*rule ImpBoxRule*)
have *20*: $\vdash \neg_i(\Box$ *more*)
by *auto*
have *21*: $\vdash \neg_i(f \wedge_i (\Box \neg_i g))$
using *17 19 20* **by** *auto*
hence *22*: $\vdash \neg_i f \vee_i \neg_i (\Box \neg_i g)$
by *auto*
have *23*: $\vdash \neg_i (\Box \neg_i g) \equiv_i \Diamond g$
by *auto*
from *22 23* **show** *?thesis* **by** *auto*
qed

lemma *DiamondIntroB*:
assumes $\vdash (f \wedge_i \neg_i g) \supset_i \bigcirc (f \wedge_i \neg_i g)$
shows $\vdash f \supset_i \Diamond g$
proof −
have *1*: $\vdash (f \wedge_i \neg_i g) \supset_i \bigcirc (f \wedge_i \neg_i g)$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i(f \wedge_i \neg_i g)$ **by** (*rule NextLoop*)
hence *3*: $\vdash f \supset_i g$ **by** *auto*
have *4*: $\vdash g \supset_i \Diamond g$ **by** (*rule NowImpDiamond*)
from *3 4* **show** *?thesis* **by** *auto*
qed

lemma *NextContra* :
assumes $\vdash (f \wedge_i \neg_i g) \supset_i (\bigcirc f \wedge_i \neg_i(\bigcirc g))$
shows $\vdash f \supset_i g$
proof −
have *1*: $\vdash (f \wedge_i \neg_i g) \supset_i (\bigcirc f \wedge_i \neg_i(\bigcirc g))$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i(f \supset_i g) \supset_i \bigcirc (\neg_i(f \supset_i g))$ **by** *auto*
hence *3*: $\vdash \neg_i \neg_i(f \supset_i g)$ **by** (*rule NextLoop*)
from *3* **show** *?thesis* **by** *auto*
qed

lemma *DiamondDiamondEqvDiamond*:
$\vdash \Diamond(\Diamond f) \equiv_i \Diamond f$
proof −
have *1*: $\vdash true_i;true_i \equiv_i true_i$ **by** *auto*
hence *2*: $\vdash (true_i;true_i);f \equiv_i true_i;f$ **using** *LeftChopEqvChop* **by** *blast*
have *3*: $\vdash (true_i;true_i);f \equiv_i true_i;(true_i;f)$ **using** *ChopAssoc itl-prop*(*30*) **by** *blast*
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *WeakNextDiamondInduct*:
assumes $\vdash$ *wnext* $(\Diamond f) \supset_i f$

shows $\vdash f$
proof −
have *1*: $\vdash$ *wnext* $(\diamond f) \supset_i f$ **using** *assms* **by** *blast*
hence *2*: $\vdash \neg_i f \supset_i \neg_i($ *wnext* $(\diamond f))$ **using** *prop27* **by** *blast*
hence *3*: $\vdash \neg_i f \supset_i \bigcirc(\neg_i (\diamond f))$ **by** *auto*
have *4*: $\vdash f \supset_i \diamond f$ **by** (*rule NowImpDiamond*)
hence *5*: $\vdash \neg_i(\diamond f) \supset_i \neg_i f$ **by** *auto*
have *6*: $\vdash \neg_i f \supset_i \bigcirc(\neg_i f)$ **using** *3 5* **using** *NextImpNext prop02* **by** *blast*
hence *7*: $\vdash \neg_i \neg_i f$ **by** (*rule NextLoop*)
from *7* **show** *?thesis* **by** *auto*
qed

lemma *EmptyNextInducta*:
assumes $\vdash$ *empty* $\supset_i f$
$\vdash \bigcirc f \supset_i f$
shows $\vdash f$
proof −
have *1*: $\vdash$ *empty* $\supset_i f$ **using** *assms* **by** *auto*
have *2*: $\vdash \bigcirc f \supset_i f$ **using** *assms* **by** *blast*
have *3*: $\vdash$ (*empty* $\vee_i \bigcirc f) \supset_i f$ **using** *1 2 prop17* **by** *blast*
have *4*: $\vdash$ *wnext* $f \equiv_i$ (*empty* $\vee_i \bigcirc f)$ **by** (*rule WnextEqvEmptyOrNext*)
hence *5*: $\vdash$ *wnext* $f \supset_i f$ **using** *3* **using** *itl-prop*(*31*) *prop02* **by** *blast*
hence *6*: $\vdash \neg_i f \supset_i \neg_i$ (*wnext* f) **by** *auto*
hence *7*: $\vdash \neg_i f \supset_i \bigcirc(\neg_i f)$ **by** *auto*
hence *8*: $\vdash \neg_i \neg_i f$ **by** (*rule NextLoop*)
from *8* **show** *?thesis* **by** *auto*
qed

lemma *EmptyNextInductb*:
assumes $\vdash$ *empty* $\wedge_i f \supset_i g$
$\vdash \bigcirc(f \supset_i g) \wedge_i f \supset_i g$
shows $\vdash f \supset_i g$
proof −
have *1*: $\vdash$ *empty* $\wedge_i f \supset_i g$ **using** *assms* **by** *auto*
have *2*: $\vdash \bigcirc(f \supset_i g) \wedge_i f \supset_i g$ **using** *assms* **by** *blast*
have *3*: $\vdash$ (*empty* $\vee_i \bigcirc(f \supset_i g)) \wedge_i f \supset_i g$ **using** *1 2 prop33* **by** *blast*
hence *4*: $\vdash$ *wnext* $(f \supset_i g) \wedge_i f \supset_i g$ **using** *prop36* **by** *auto*
hence *5*: $\vdash$ *wnext* $(f \supset_i g) \supset_i (f \supset_i g)$ **using** *prop36* **by** *blast*
hence *6*: $\vdash \neg_i (f \supset_i g) \supset_i \neg_i$ (*wnext* $(f \supset_i g))$ **using** *prop27* **by** *blast*
hence *7*: $\vdash \neg_i (f \supset_i g) \supset_i \bigcirc (\neg_i(f \supset_i g))$ **by** *simp*
hence *8*: $\vdash \neg_i \neg_i (f \supset_i g)$ **by** (*rule NextLoop*)
from *8* **show** *?thesis* **by** *auto*
qed

lemma *FinImpFin*:
assumes $\vdash f \supset_i g$
shows $\vdash$ *fin* $f \supset_i$ *fin* g
using *ImpBoxRule assms* **by** *auto*

lemma *FinEqvFin*:
assumes $\vdash f \equiv_i g$
shows $\vdash$ *fin* $f \equiv_i$ *fin* g
using *FinImpFin assms itl-prop*(*31*) **by** *blast*

lemma *FinAndFinImpFinRule*:
assumes $\vdash f \wedge_i g \supset_i h$
shows $\vdash$ *fin* $f \wedge_i$ *fin* $g \supset_i$ *fin* h
proof −
have $\vdash f \wedge_i g \supset_i h$ **using** *assms* **by** *auto*
then show *?thesis* **by** *simp*
qed

lemma *FinAndFinEqvFinRule*:
assumes $\vdash f \wedge_i g \equiv_i h$
shows $\vdash$ *fin* $f \wedge_i$ *fin* $g \equiv_i$ *fin* h
by (*meson FinAndFinImpFinRule FinImpFin assms itl-prop*(*31*) *itl-prop*(*32*))

lemma *HaltEqvHalt*:
assumes $\vdash f \equiv_i g$
shows $\vdash$ *halt* $f \equiv_i$ *halt* g
proof −
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash$ (*empty* $\equiv_i f$) $\equiv_i$ (*empty* $\equiv_i g$) **by** *auto*
hence *3*: $\vdash \Box$(*empty* $\equiv_i f$) $\equiv_i \Box$ (*empty* $\equiv_i g$) **by** (*rule BoxEqvBox*)
from *3* **show** *?thesis* **by** (*simp add*: *halt-d-def*)
qed

lemma *BiImpDiImpDi*:
$\vdash$ *bi* ($f \supset_i g$) $\supset_i$ *di* $f \supset_i$ *di* g
proof −
have *1*: $\vdash$ *bi* ($f \supset_i g$) $\supset_i$ (f; *true*$_i$) $\supset_i$ (g; *true*$_i$) **by** (*rule BiChopImpChop*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiImpDi*:
assumes $\vdash f \supset_i g$
shows $\vdash$ *di* $f \supset_i$ *di* g
proof −
have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash f$; *true*$_i \supset_i g$; *true*$_i$ **by** (*rule LeftChopImpChop*)
from *2* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *BiImpBiRule*:
assumes $\vdash f \supset_i g$
shows $\vdash$ *bi* $f \supset_i$ *bi* g

proof −
have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i\ g \supset_i \neg_i\ f$ **by** *auto*
hence *3*: $\vdash\ di\ \neg_i\ g \supset_i\ di\ \neg_i\ f$ **by** (*rule DiImpDi*)
hence *4*: $\vdash \neg_i\ (\ di\ \neg_i\ f) \supset_i \neg_i\ (\ di\ \neg_i\ g)$ **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *bi-d-def*)
qed

lemma *DiEqvDi*:
assumes $\vdash f \equiv_i g$
shows $\vdash\ di\ f \equiv_i\ di\ g$
proof −
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ true_i \equiv_i g;\ true_i$ **by** (*rule LeftChopEqvChop*)
from *2* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *BiEqvBi*:
assumes $\vdash f \equiv_i g$
shows $\vdash\ bi\ f \equiv_i\ bi\ g$
proof −
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i\ f \equiv_i \neg_i\ g$ **by** *auto*
hence *3*: $\vdash\ di\ \neg_i\ f \equiv_i\ di\ \neg_i\ g$ **by** (*rule DiEqvDi*)
hence *4*: $\vdash \neg_i\ (di\ \neg_i\ f) \equiv_i \neg_i\ (\ di\ \neg_i\ g)$ **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *bi-d-def*)
qed

lemma *LeftChopChopImpChopRule*:
assumes $\vdash (f;\ g) \supset_i g$
shows $\vdash (f;\ g);\ h \supset_i (g;\ h)$
proof −
have *1*: $\vdash (f;\ g) \supset_i g$ **using** *assms* **by** *blast*
hence *2*: $\vdash (f;\ g);\ h \supset_i g;\ h$ **by** (*rule LeftChopImpChop*)
have *3*: $\vdash f;\ (g;\ h) \equiv_i (f;\ g);\ h$ **by** (*rule ChopAssoc*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *AndChopCommute* :
$\vdash (f \wedge_i f1);\ g \equiv_i (f1 \wedge_i f);\ g$
proof −
have *1*: $\vdash f \wedge_i f1 \equiv_i f1 \wedge_i f$ **by** *auto*
from *1* **show** *?thesis* **by** (*rule LeftChopEqvChop*)
qed

lemma *BiAndChopImport*:
$\vdash\ bi\ f \wedge_i (f1;\ g) \supset_i (f \wedge_i f1);\ g$
proof −
have *1*: $\vdash f \supset_i (f1 \supset_i f \wedge_i f1)$ **by** *auto*
hence *2*: $\vdash bi\ f \supset_i bi\ (f1 \supset_i f \wedge_i f1)$ **by** (*rule BiImpBiRule*)

have *3*: $\vdash$ *bi* (*f1* $\supset_i$ (*f* $\wedge_i$ *f1*)) $\supset_i$ *f1*; *g* $\supset_i$ (*f* $\wedge_i$ *f1*); *g* **by** (*rule BiChopImpChop*)
from *1 3* **show** *?thesis* **using** *MP* **by** *auto*
qed

lemma *StateAndChopImport*:
$\vdash$ (*init w*) $\wedge_i$ (*f*; *g*) $\supset_i$ ((*init w*) $\wedge_i$ *f*); *g*
proof −
have *1*: $\vdash$ (*init w*)$\supset_i$ *bi* (*init w*) **by** (*rule StateImpBi*)
hence *2*: $\vdash$ (*init w*) $\wedge_i$ (*f*; *g*) $\supset_i$ *bi* (*init w*) $\wedge_i$ (*f*; *g*) **by** *auto*
have *3*: $\vdash$ *bi* (*init w*) $\wedge_i$ (*f*; *g*) $\supset_i$ ((*init w*) $\wedge_i$ *f*); *g* **by** (*rule BiAndChopImport*)
from *2 3* **show** *?thesis* **using** *MP* **by** *auto*
qed

5.4 Further Properties Di and Bi

lemma *ImpDi*:
$\vdash$ *f* $\supset_i$ *di f*
proof −
have *1*: $\vdash$ *f*; *empty* $\equiv_i$ *f* **by** (*rule ChopEmpty*)
have *2*: $\vdash$ *empty* $\supset_i$ $true_i$ **by** *auto*
hence *3*: $\vdash$ *f*; *empty* $\supset_i$ *f*; $true_i$ **by** (*rule RightChopImpChop*)
have *4* : $\vdash$ *f* $\supset_i$ *f*; $true_i$ **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiState*:
$\vdash$ *di* (*init w*) $\equiv_i$ (*init w*)
proof −
have *0*: $\vdash$ (*init* $\neg_i$*w*) $\supset_i$ *bi* (*init* $\neg_i$*w*) **using** *StateImpBi* **by** *fastforce*
hence *1*: $\vdash$ $\neg_i$(*init w*) $\supset_i$ *bi* $\neg_i$ (*init w*) **using** *Initprop* **by** *auto*
hence *2*: $\vdash$ $\neg_i$ (*init w*)$\supset_i$ $\neg_i$ (*di* $\neg_i$ $\neg_i$ (*init w*)) **by** (*simp add*: *bi-d-def*)
have *3*: $\vdash$ ($\neg_i$ (*init w*) $\supset_i$$\neg_i$ (*di* $\neg_i$ $\neg_i$ (*init w*))) $\supset_i$ (*di* $\neg_i$ $\neg_i$ (*init w*) $\supset_i$ (*init w*)) **by** *auto*
have *4*: $\vdash$ *di* $\neg_i$ $\neg_i$ (*init w*) $\supset_i$ (*init w*) **using** *2 3 MP* **by** *blast*
have *5*: $\vdash$ (*init w*) $\supset_i$ $\neg_i$ $\neg_i$ (*init w*) **by** *auto*
hence *6*: $\vdash$ *di* (*init w*) $\supset_i$ *di* $\neg_i$ $\neg_i$ (*init w*) **by** (*rule DiImpDi*)
have *7*: $\vdash$ *di* (*init w*) $\supset_i$ (*init w*) **using** *6 4 MP* **using** *prop02* **by** *blast*
have *8*: $\vdash$ (*init w*) $\supset_i$ *di* (*init w*) **by** (*rule ImpDi*)
from *7 8* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *StateChop*:
$\vdash$ (*init w*); *f* $\supset_i$ (*init w*)
using *DiState* **by** *auto*

lemma *StateChopExportA*:
$\vdash$ ((*init w*) $\wedge_i$ *f*); *g* $\supset_i$ (*init w*)
using *DiState* **by** *auto*

lemma *StateAndChop*:
$\vdash$ ((*init w*) $\wedge_i$ *f*); *g* $\equiv_i$(*init w*) $\wedge_i$ (*f*; *g*)

using *StateAndChopImport StateChopExportA AndChopB itl-prop*(*31*) *itl-prop*(*32*) **by** *blast*

lemma *StateAndChopImpChopRule*:
 assumes $\vdash$ (*init w*) $\wedge_i$ *f* $\supset_i$ *f1*
 shows $\vdash$ (*init w*) $\wedge_i$ (*f*; *g*) $\supset_i$ (*f1*; *g*)
proof −
 have *1*: $\vdash$ (*init w*) $\wedge_i$ *f* $\supset_i$ *f1* **using** *assms* **by** *auto*
 hence *2*: $\vdash$ ((*init w*) $\wedge_i$ *f*); *g* $\supset_i$ *f1*; *g* **by** (*rule LeftChopImpChop*)
 have *3*: $\vdash$ ((*init w*) $\wedge_i$ *f*); *g* $\equiv_i$ (*init w*) $\wedge_i$ (*f*; *g*) **by** (*rule StateAndChop*)
 from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *StateImpChopEqvChop* :
 assumes $\vdash$(*init w*) $\supset_i$ (*f* $\equiv_i$ *f1*)
 shows $\vdash$ (*init w*) $\supset_i$ ((*f*; *g*) $\equiv_i$ (*f1*; *g*))
proof −
 have *1*: $\vdash$ (*init w*) $\supset_i$ (*f*$\equiv_i$ *f1*) **using** *assms* **by** *auto*
 hence *2*: $\vdash$ (*init w*) $\wedge_i$ *f* $\supset_i$ *f1* **by** *auto*
 hence *3*: $\vdash$ (*init w*) $\wedge_i$ (*f*; *g*) $\supset_i$ (*f1*; *g*) **by** (*rule StateAndChopImpChopRule*)
 have *4*: $\vdash$ (*init w*) $\wedge_i$ *f1* $\supset_i$ *f* **using** *1* **by** *auto*
 hence *5*: $\vdash$ (*init w*) $\wedge_i$ (*f1*; *g*) $\supset_i$ (*f*; *g*) **by** (*rule StateAndChopImpChopRule*)
 from *3 5* **show** *?thesis* **by** *auto*
qed

lemma *ChopEqvStateAndChop*:
  assumes $\vdash$ *f* $\equiv_i$ (*init w*) $\wedge_i$ *f1*
  shows $\vdash$ (*f*; *g*) $\equiv_i$ (*init w*) $\wedge_i$ (*f1*; *g*)
proof −
 have *1*: $\vdash$ *f* $\equiv_i$ (*init w*) $\wedge_i$ *f1* **using** *assms* **by** *auto*
 hence *2*: $\vdash$ *f*; *g* $\equiv_i$ ((*init w*) $\wedge_i$ *f1*); *g* **by** (*rule LeftChopEqvChop*)
 have *3*: $\vdash$ ((*init w*) $\wedge_i$ *f1*); *g* $\equiv_i$ (*init w*) $\wedge_i$ (*f1*; *g*) **by** (*rule StateAndChop*)
 from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *DiIntro*:
$\vdash$ *f* $\supset_i$ *di f*
proof −
  have *1*: $\vdash$ *f*; *empty* $\equiv_i$ *f* **by** (*rule ChopEmpty*)
  have *2*: $\vdash$ *empty* $\supset_i$ *true$_i$* **by** *auto*
  hence *3*: $\vdash$ $\Box$(*empty* $\supset_i$ *true$_i$*) **by** (*rule BoxGen*)
  have *4*: $\vdash$ $\Box$(*empty* $\supset_i$ *true$_i$*) $\supset_i$ (*f*; *empty* $\supset_i$ *f*; *true$_i$*) **by** (*rule BoxChopImpChop*)
  have *5*: $\vdash$ *f*; *empty* $\supset_i$ *f*; *true$_i$* **using** *3 4 MP* **by** *auto*
  hence *6*: $\vdash$ *f*; *empty* $\supset_i$ *di f* **by** (*simp add*: *di-d-def*)
  from *1 6* **show** *?thesis* **by** *auto*
qed

lemma *BiElim*:
$\vdash$ *bi f* $\supset_i$ *f*
proof −
  have *1*: $\vdash$ $\neg_i$ *f* $\supset_i$ *di* $\neg_i$ *f* **by** (*rule DiIntro*)

have *2*: $\vdash (\neg_i\ f \supset_i\ di\ \neg_i\ f) \supset_i (\neg_i\ (\ di\ \neg_i\ f) \supset_i f)$ **by** *auto*
have *3*: $\vdash \neg_i\ (di\ \neg_i\ f) \supset_i f$ **using** *1 2 MP* **by** *blast*
from *3* **show** *?thesis* **by** (*metis bi-d-def*)
qed

lemma *BiContraPosImpDist*:
$\vdash bi\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i (bi\ f) \supset_i (bi\ g)$
proof –
have *1*: $\vdash bi\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i (\ di\ \neg_i\ g) \supset_i (\ di\ \neg_i\ f)$ **by** (*rule BiImpDiImpDi*)
hence *2*: $\vdash bi\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i (\neg_i\ (\ di\ \neg_i\ f)) \supset_i (\neg_i\ (\ di\ \neg_i\ g))$ **by** *auto*
from *2* **show** *?thesis* **by** (*metis bi-d-def*)
qed

lemma *BiImpDist*:
$\vdash bi\ (f \supset_i g) \supset_i (bi\ f) \supset_i (bi\ g)$
proof –
have *1*: $\vdash (f \supset_i g) \supset_i (\neg_i\ g \supset_i \neg_i\ f)$ **by** *auto*
hence *2*: $\vdash \neg_i\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i \neg_i\ (f \supset_i g)$ **by** *auto*
hence *3*: $\vdash bi\ (\neg_i\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i \neg_i\ (f \supset_i g))$ **by** (*rule BiGen*)
have *4*: $\vdash bi\ (\neg_i\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i \neg_i\ (f \supset_i g))$
$\supset_i$
$bi\ (f \supset_i g) \supset_i bi\ (\neg_i\ g \supset_i \neg_i\ f)$ **by** (*rule BiContraPosImpDist*)
have *5*: $\vdash bi\ (f \supset_i g) \supset_i bi\ (\neg_i\ g \supset_i \neg_i\ f)$ **using** *3 4 MP* **by** *blast*
have *6*: $\vdash bi\ (\neg_i\ g \supset_i \neg_i\ f) \supset_i (bi\ f) \supset_i (bi\ g)$ **by** (*rule BiContraPosImpDist*)
from *5 6* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *IfChopEqvRule*:
assumes $\vdash f \equiv_i\ if_i\ (init\ w)\ then\ f1\ else\ f2$
shows $\vdash f; g \equiv_i\ if_i\ (init\ w)\ then\ (f1; g)\ else\ (f2; g)$
proof –
have *1*: $\vdash f \equiv_i\ if_i\ (init\ w)\ then\ f1\ else\ f2$ **using** *assms* **by** *auto*
hence *2*: $\vdash f \equiv_i ((init\ w) \wedge_i f1) \vee_i\ (\ (init\ \neg_i\ w) \wedge_i f2)$ **by** (*simp add*: *ifthenelse-d-def*)
hence *3*: $\vdash f; g \equiv_i ((init\ w) \wedge_i f1); g \vee_i\ (\ \ (init\ \neg_i\ w) \wedge_i f2); g$ **by** (*rule OrChopEqvRule*)
have *4*: $\vdash ((init\ w) \wedge_i f1); g \equiv_i (init\ w) \wedge_i (f1; g)$ **by** (*rule StateAndChop*)
have *5*: $\vdash (\ \ (init\ \neg_i\ w) \wedge_i f2); g \equiv_i\ \ (init\ \neg_i\ w) \wedge_i (f2; g)$ **by** (*rule StateAndChop*)
have *6*: $\vdash f; g \equiv_i ((init\ w) \wedge_i f1; g)\ \vee_i\ (\ \ (init\ \neg_i\ w) \wedge_i f2; g)$ **using** *3 4 5* **by** *auto*
from *6* **show** *?thesis* **by** (*simp add*: *ifthenelse-d-def*)
qed

lemma *ChopOrEqvRule*:
assumes $\vdash g \equiv_i g1 \vee_i\ g2$
shows $\vdash f; g \equiv_i (f; g1) \vee_i\ (f; g2)$
proof –
have *1*: $\vdash g \equiv_i g1 \vee_i\ g2$ **using** *assms* **by** *auto*
hence *2*: $\vdash f; g \equiv_i f; (g1 \vee_i\ g2)$ **by** (*rule RightChopEqvChop*)
have *3*: $\vdash f; (g1 \vee_i\ g2) \equiv_i f; g1 \vee_i\ f; g2$ **by** (*rule ChopOrEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrChopEqv*:
$\vdash$ (*empty* $\vee_i$ f); $g \equiv_i g \vee_i$ (f; g)
proof $-$
have *1*: $\vdash$ (*empty* $\vee_i$ f); $g \equiv_i$ (*empty* ; g) $\vee_i$ (f; g) **by** (*rule OrChopEqv*)
have *2*: $\vdash$ *empty* ; $g \equiv_i g$ **by** (*rule EmptyChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrNextChopEqv*:
$\vdash$ (*empty* $\vee_i$ $\bigcirc f$); $g \equiv_i g \vee_i$ $\bigcirc$(f; g)
proof $-$
have *1*: $\vdash$ (*empty* $\vee_i$ $\bigcirc f$); $g \equiv_i g \vee_i$ (($\bigcirc f$); g) **by** (*rule EmptyOrChopEqv*)
have *2*: $\vdash$ ($\bigcirc f$); $g \equiv_i \bigcirc$(f; g) **by** (*rule NextChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrChopImpRule*:
assumes $\vdash$ $f \supset_i$ *empty* $\vee_i$ *f1*
shows $\vdash$ f; $g \supset_i g \vee_i$ (*f1*; g)
proof $-$
have *1*: $\vdash$ $f \supset_i$ *empty* $\vee_i$ *f1* **using** *assms* **by** *auto*
hence *2*: $\vdash$ f; $g \supset_i$ (*empty* $\vee_i$ *f1*); g **by** (*rule LeftChopImpChop*)
have *3*: $\vdash$ (*empty* $\vee_i$ *f1*); $g \equiv_i g \vee_i$ (*f1*; g) **by** (*rule EmptyOrChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrChopEqvRule*:
assumes $\vdash$ $f \equiv_i$ *empty* $\vee_i$ *f1*
shows $\vdash$ f; $g \equiv_i g \vee_i$ (*f1*; g)
proof $-$
have *1*: $\vdash$ $f \equiv_i$ *empty* $\vee_i$ *f1* **using** *assms* **by** *auto*
hence *2*: $\vdash$ f; $g \equiv_i$ (*empty* $\vee_i$ *f1*); g **by** (*rule LeftChopEqvChop*)
have *3*: $\vdash$ (*empty* $\vee_i$ *f1*); $g \equiv_i g \vee_i$ (*f1*; g) **by** (*rule EmptyOrChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrNextChopImpRule*:
assumes $\vdash$ $f \supset_i$ *empty* $\vee_i$ $\bigcirc$ *f1*
shows $\vdash$ f; $g \supset_i g \vee_i$ $\bigcirc$(*f1*; g)
proof $-$
have *1*: $\vdash$ $f \supset_i$ *empty* $\vee_i$ $\bigcirc$ *f1* **using** *assms* **by** *auto*
hence *2*: $\vdash$ f; $g \supset_i$ (*empty* $\vee_i$ $\bigcirc$ *f1*); g **by** (*rule LeftChopImpChop*)
have *3*: $\vdash$ (*empty* $\vee_i$ $\bigcirc$ *f1*); $g \equiv_i g \vee_i$ $\bigcirc$(*f1*; g) **by** (*rule EmptyOrNextChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *EmptyOrNextChopEqvRule*:
assumes $\vdash$ $f \equiv_i$ *empty* $\vee_i$ $\bigcirc$ *f1*
shows $\vdash$ f; $g \equiv_i g \vee_i$ $\bigcirc$(*f1*; g)
proof $-$

have *1*: $\vdash f \equiv_i$ *empty* $\vee_i$ $\bigcirc$ *f1* **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ g \equiv_i$ (*empty* $\vee_i$ $\bigcirc$ *f1*); *g* **by** (*rule LeftChopEqvChop*)
have *3*: $\vdash$ (*empty* $\vee_i$ $\bigcirc$ *f1*); $g \equiv_i g \vee_i$ $\bigcirc$(*f1*; *g*) **by** (*rule EmptyOrNextChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *ChopEmptyOrImpRule*:
assumes $\vdash$ $g \supset_i$ *empty* $\vee_i$ *g1*
shows $\vdash f;\ g \supset_i f \vee_i$ (*f*; *g1*)
proof $-$
have *1*: $\vdash g \supset_i$ *empty* $\vee_i$ *g1* **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ g \supset_i$ (*f*; *empty*) $\vee_i$ (*f*; *g1*) **by** (*rule ChopOrImpRule*)
have *3*: $\vdash f;$ *empty* $\equiv_i f$ **by** (*rule ChopEmpty*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *StateAndEmptyImpBoxState*:
$\vdash$ (*init w*) $\wedge_i$ *empty* $\supset_i \Box$ (*init w*)
by *simp*

lemma *BoxEqvAndBox*:
$\vdash \Box f \equiv_i f \wedge_i \Box f$
by *fastforce*

lemma *NotBoxImpNotOrNotNextBox*:
$\vdash \neg_i(\Box f) \supset_i \neg_i f \vee_i \neg_i(\bigcirc (\Box f))$
proof $-$
have *1*: $\vdash f \wedge_i$ *wnext* $(\Box f) \equiv_i f \wedge_i \Box f$
by (*meson BoxEqvAndBox BoxEqvAndWnextBox prop21*)
have *2*: $\vdash \neg_i$ (*wnext* $(\Box f)) \supset_i \neg_i (\bigcirc (\Box f))$
by (*metis* (*full-types*) *NextImpNotNextNot prop27 wnext-d-def*)
then show *?thesis* **using** *1* **by** (*metis* (*no-types*) *and-d-def itl-prop*(*33*) *prop19 prop35*)
qed

lemma *BoxStateChopBoxEqvBox*:
$\vdash$ $\Box$ (*init w*); $\Box$ (*init w*) $\equiv_i \Box$ (*init w*)
proof $-$
have *1*: $\vdash \Box$ (*init w*) $\equiv_i$ (*init w*) $\wedge_i$ (*empty* $\vee_i$ $\bigcirc$($\Box$ (*init w*)))
by (*rule BoxEqvAndEmptyOrNextBox*)
hence *2*: $\vdash \Box$ (*init w*); $\Box$ (*init w*) $\equiv_i$
(*init w*) $\wedge_i$ ((*empty* $\vee_i$ $\bigcirc$($\Box$ (*init w*))); $\Box$ (*init w*))
by (*rule ChopEqvStateAndChop*)
have *3*: $\vdash$ (*empty* $\vee_i$ $\bigcirc$($\Box$ (*init w*))); $\Box$ (*init w*) $\equiv_i$
$\Box$ (*init w*) $\vee_i$ $\bigcirc$($\Box$ (*init w*); $\Box$ (*init w*))
by (*rule EmptyOrNextChopEqv*)
have *4*: $\vdash \Box$ (*init w*); $\Box$ (*init w*) $\equiv_i$
(*init w*) $\wedge_i$ ($\Box$ (*init w*) $\vee_i$ $\bigcirc$($\Box$ (*init w*); $\Box$ (*init w*)))
using *2 3* **by** *auto*
have *5*: $\vdash \neg_i$ ($\Box$ (*init w*)) $\supset_i \neg_i$ (*init w*) $\vee_i$ $\neg_i$($\bigcirc$($\Box$ (*init w*)))

by (*rule NotBoxImpNotOrNotNextBox*)
have *6*: $\vdash$ ($\Box$ (*init w*); $\Box$ (*init w*)) $\wedge_i$ $\neg_i$($\Box$ (*init w*)) $\supset_i$
$\bigcirc$($\Box$ (*init w*); $\Box$ (*init w*)) $\wedge_i$ $\neg_i$($\bigcirc$($\Box$ (*init w*)))
using *4 5* **by** *auto*
hence *7*: $\vdash$ $\Box$ (*init w*); $\Box$ (*init w*) $\supset_i$ $\Box$ (*init w*)
by (*rule NextContra*)
have *11*: $\vdash$ $\Box$ (*init w*) $\equiv_i$ (*init w*) $\wedge_i$ $\Box$ (*init w*)
by (*rule BoxEqvAndBox*)
have *12*: $\vdash$ *empty* ; $\Box$ (*init w*) $\equiv_i$ $\Box$ (*init w*)
by (*rule EmptyChop*)
have *13*: $\vdash$ ((*init w*) $\wedge_i$ *empty*); $\Box$ (*init w*) $\equiv_i$ (*init w*) $\wedge_i$ (*empty* ; $\Box$ (*init w*))
by (*rule StateAndChop*)
have *14*: $\vdash$ $\Box$ (*init w*) $\equiv_i$ ((*init w*) $\wedge_i$ *empty*); $\Box$ (*init w*)
using *11 12 13* **by** *auto*
have *15*: $\vdash$ (*init w*) $\wedge_i$ *empty* $\supset_i$ $\Box$ (*init w*)
by (*rule StateAndEmptyImpBoxState*)
hence *16*: $\vdash$ ((*init w*) $\wedge_i$ *empty*); $\Box$ (*init w*) $\supset_i$ $\Box$ (*init w*); $\Box$ (*init w*)
by (*rule LeftChopImpChop*)
have *17*: $\vdash$ $\Box$ (*init w*) $\supset_i$ $\Box$ (*init w*); $\Box$ (*init w*)
using *14 16* **by** *auto*
from *7 17* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *NotBoxStateImpBoxYieldsNotBox*:
$\vdash$ $\neg_i$($\Box$ (*init w*)) $\supset_i$ ($\Box$ (*init w*)) *yields* $\neg_i$($\Box$ (*init w*))
proof −
have *1*: $\vdash$ $\Box$ (*init w*); $\Box$ (*init w*) $\equiv_i$ $\Box$ (*init w*) **by** (*rule BoxStateChopBoxEqvBox*)
have *2*: $\vdash$ $\Box$ (*init w*) $\equiv_i$ $\neg_i$ $\neg_i$($\Box$ (*init w*)) **by** *auto*
hence *3*: $\vdash$ $\Box$ (*init w*); $\Box$ (*init w*) $\equiv_i$ $\Box$ (*init w*); $\neg_i$ $\neg_i$($\Box$ (*init w*)) **by** (*rule RightChopEqvChop*)
have *4*: $\vdash$ $\neg_i$($\Box$ (*init w*)) $\supset_i$ $\neg_i$ ($\Box$ (*init w*); $\neg_i$ $\neg_i$ ($\Box$ (*init w*))) **using** *1 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *StateEqvBi*:
$\vdash$ (*init w*) $\equiv_i$ *bi* (*init w*)
proof −
have *1*: $\vdash$ (*init w*) $\supset_i$ *bi* (*init w*) **by** (*rule StateImpBi*)
have *2*: $\vdash$ *bi* (*init w*) $\supset_i$ (*init w*) **by** (*rule BiElim*)
from *1 2* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *TrueChopEqvDiamond*:
$\vdash$ $true_i$; *f* $\equiv_i$ $\Diamond$ *f*
by *simp*

5.5 Properties of Da and Ba

lemma *DaEqvDtDi*:
$\vdash$ *da f* $\equiv_i$ $\Diamond$ (*di f*)

proof $-$
have *1*: $\vdash$ $true_i$; $(f;\ true_i) \equiv_i true_i; (f;\ true_i)$ **by** *auto*
hence *2*: $\vdash$ $true_i$; $(f;\ true_i) \equiv_i true_i;\ \ di\ \ f$ **by** (*simp add*: *di-d-def*)
have *3*: $\vdash$ $true_i;\ \ di\ \ f \equiv_i \Diamond(\ di\ \ f)$ **by** (*rule TrueChopEqvDiamond*)
have *4*: $\vdash$ $true_i$; $(f;\ true_i) \equiv_i \Diamond(\ di\ \ f)$ **using** *2 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*:*da-d-def*)
qed

lemma *DaEqvDiDt*:
$\vdash\ \ \ da\ \ f \equiv_i\ di\ (\Diamond\ f)$
proof $-$
have *1*: $\vdash$ $true_i;\ f \equiv_i \Diamond\ f$ **by** (*rule TrueChopEqvDiamond*)
hence *2*: $\vdash$ $(true_i;\ f);\ true_i \equiv_i (\Diamond\ f);\ true_i$ **by** (*rule LeftChopEqvChop*)
hence *3*: $\vdash$ $(true_i;\ f);\ true_i \equiv_i\ \ di(\ \Diamond\ f)$ **by** (*simp add*: *di-d-def*)
have *4*: $\vdash$ $true_i;\ (f;\ true_i) \equiv_i (true_i;\ f);\ true_i$ **by** (*rule ChopAssoc*)
have *5*: $\vdash$ $true_i;\ (f;\ true_i) \equiv_i\ \ di\ (\Diamond\ f)$ **using** *3 4* **by** *auto*
from *5* **show** *?thesis* **by** (*simp add*: *da-d-def*)
qed

lemma *DtDiEqvDiDt*:
$\vdash\ \ \ \Diamond\ (di\ \ f) \equiv_i\ \ di\ (\Diamond\ f)$
by (*metis ChopAssoc di-d-def sometimes-d-def*)

lemma *DiamondNotEqvNotBox*:
$\vdash \Diamond\ \neg_i\ \ f \equiv_i \neg_i\ (\Box\ f)$
by *simp*

lemma *BaEqvBiBt*:
$\vdash\ \ \ ba\ \ f \equiv_i bi(\ \Box\ f)$
proof $-$
have *1*: $\vdash\ \ da\ \neg_i\ \ f \equiv_i\ \ di(\ \Diamond\ \neg_i\ \ f)$ **by** (*rule DaEqvDiDt*)
have *2*: $\vdash \Diamond\ \neg_i\ \ f \equiv_i \neg_i(\ \Box\ f)$ **by** (*rule DiamondNotEqvNotBox*)
hence *3*: $\vdash\ \ di\ (\Diamond\neg_i\ \ f) \equiv_i\ \ di\ \neg_i\ (\Box\ f)$ **by** (*rule DiEqvDi*)
have *4*: $\vdash\ \ da\ \neg_i\ \ f \equiv_i\ \ di\ \neg_i(\ \Box\ f)$ **using** *1 3* **by** *auto*
hence *5*: $\vdash \neg_i\ \ (da\ \neg_i\ \ f) \equiv_i \neg_i\ (\ di\ \neg_i(\ \Box\ f))$ **by** *auto*
hence *6*: $\vdash \neg_i\ (\ da\ \neg_i\ \ f) \equiv_i bi(\ \Box\ f)$ **by** (*simp add*: *bi-d-def*)
from *6* **show** *?thesis* **by** (*simp add*: *ba-d-def*)
qed

lemma *DiNotEqvNotBi*:
$\vdash\ \ \ di\ \neg_i\ \ f \equiv_i \neg_i(\ bi\ \ f)$
proof $-$
have *1*: $\vdash bi\ \ f \equiv_i \neg_i\ (\ di\ \neg_i\ \ f)$ **by** (*simp add*: *bi-d-def*)
from *1* **show** *?thesis* **by** *auto*
qed

lemma *NotDiamondNotEqvBox*:
$\vdash \neg_i\ (\Diamond\neg_i\ f) \equiv_i \Box\ f$
by *simp*

lemma *BaEqvBtBi*:
$\vdash \quad ba \;\; f \equiv_i \Box \, (bi \;\; f)$
proof $-$
have *1*: $\vdash \;\; da \, \neg_i \;\; f \equiv_i \Diamond \, (di \, \neg_i \;\; f)$ **by** (*rule DaEqvDtDi*)
have *2*: $\vdash \;\; di \, \neg_i \;\; f \equiv_i \neg_i \, (bi \;\; f)$ **by** (*rule DiNotEqvNotBi*)
hence *3*: $\vdash \Diamond \, (di \, \neg_i \;\; f) \equiv_i \Diamond\neg_i \, (bi \;\; f)$ **by** (*rule DiamondEqvDiamond*)
have *4*: $\vdash \neg_i \, (\Diamond\neg_i \, (bi \;\; f)) \equiv_i \Box(bi \;\; f)$ **by** (*rule NotDiamondNotEqvBox*)
have *5*: $\vdash \neg_i \, (\, da \, \neg_i \;\; f) \equiv_i \Box(bi \;\; f)$ **using** *1 2 3* **by** *auto*
from *5* **show** *?thesis* **by** (*simp add*: *ba-d-def*)
qed

lemma *BtBiEqvBiBt*:
$\vdash \quad \Box \, (bi \;\; f) \equiv_i bi(\, \Box \, f)$
proof $-$
have *1*: $\vdash \quad ba \;\; f \equiv_i \Box \, (bi \;\; f)$ **by** (*rule BaEqvBtBi*)
have *2*: $\vdash \quad ba \;\; f \equiv_i bi(\, \Box \, f)$ **by** (*rule BaEqvBiBt*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *BoxStateEqvBaBoxState*:
$\vdash \quad \Box \, (init \; w) \equiv_i \;\; ba \, (\Box \, (init \; w))$
proof $-$
have *1*: $\vdash (init \; w) \equiv_i bi \;\; (init \; w)$ **by** (*rule StateEqvBi*)
hence *2*: $\vdash \Box \, (init \; w) \equiv_i \Box \, (bi \;\; (init \; w))$ **by** (*rule BoxEqvBox*)
have *3*: $\vdash \Box \, (bi \;\; (init \; w)) \equiv_i bi(\, \Box \, (init \; w))$ **by** (*rule BtBiEqvBiBt*)
have *4*: $\vdash \Box \, (init \; w) \equiv_i \Box(\Box \, (init \; w))$ **by** (*rule BoxEqvBoxBox*)
hence *5*: $\vdash bi(\, \Box \, (init \; w)) \equiv_i bi \, (\Box(\Box \, (init \; w)))$ **by** (*rule BiEqvBi*)
have *6*: $\vdash \;\; ba(\, \Box \, (init \; w)) \equiv_i bi(\, \Box(\Box \, (init \; w)))$ **by** (*rule BaEqvBiBt*)
from *2 3 5 6* **show** *?thesis* **by** *simp*
qed

lemma *BaImpBi*:
$\vdash \quad ba \;\; f \supset_i bi \;\; f$
proof $-$
have *1*: $\vdash \;\; ba \;\; f \equiv_i \Box(bi \;\; f)$ **by** (*rule BaEqvBtBi*)
have *2*: $\vdash \Box(bi \;\; f) \supset_i bi \;\; f$ **by** (*rule BoxElim*)
from *1 2* **show** *?thesis* **using** *MP* **using** *itl-prop*(*31*) *prop02* **by** *blast*
qed

lemma *BaImpBt*:
$\vdash \quad ba \;\; f \supset_i \Box \, f$
proof $-$
have *1*: $\vdash \;\; ba \;\; f \equiv_i bi(\, \Box \, f)$ **by** (*rule BaEqvBiBt*)
have *2*: $\vdash bi(\, \Box \, f) \supset_i \Box \, f$ **by** (*rule BiElim*)
from *1 2* **show** *?thesis* **using** *MP* **using** *itl-prop*(*31*) *prop02* **by** *blast*
qed

lemma *DiamondImpDa*:
$\vdash \quad \Diamond \, f \supset_i \;\; da \;\; f$
by (*metis DiIntro DiamondImpDiamond da-d-def di-d-def sometimes-d-def*)

lemma *DiImpDa*:
$\vdash$ *di* $f \supset_i$ *da* f
by (*metis NowImpDiamond da-d-def di-d-def sometimes-d-def*)

lemma *BoxAndChopImport*:
$\vdash$ $\Box\, h \wedge_i f; g \supset_i f; (h \wedge_i g)$
proof −
have *1*: $\vdash h \supset_i g \supset_i (h \wedge_i g)$ **by** *auto*
hence *2*: $\vdash \Box\, h \supset_i \Box(g \supset_i (h \wedge_i g))$ **by** (*rule ImpBoxRule*)
have *3*: $\vdash \Box(g \supset_i (h \wedge_i g)) \supset_i f; g \supset_i f; (h \wedge_i g)$ **by** (*rule BoxChopImpChop*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *BaAndChopImport*:
$\vdash$ *ba* $f \wedge_i (g; g1) \supset_i (f \wedge_i g); (f \wedge_i g1)$
proof −
have *1*: $\vdash$ *ba* $f \supset_i$ *bi* f **by** (*rule BaImpBi*)
have *2*: $\vdash$ *bi* $f \wedge_i (g; g1) \supset_i (f \wedge_i g); g1$ **by** (*rule BiAndChopImport*)
have *3*: $\vdash$ *ba* $f \supset_i \Box\, f$ **by** (*rule BaImpBt*)
have *4*: $\vdash \Box\, f \wedge_i (f \wedge_i g); g1 \supset_i (f \wedge_i g); (f \wedge_i g1)$ **by** (*rule BoxAndChopImport*)
from *1 2 3 4* **show** *?thesis* **by** *simp*
qed

lemma *ChopAndCommute*:
$\vdash$ $f; (g \wedge_i g1) \equiv_i f; (g1 \wedge_i g)$
proof −
have *1*: $\vdash (g \wedge_i g1) \equiv_i (g1 \wedge_i g)$ **by** *auto*
from *1* **show** *?thesis* **by** (*rule RightChopEqvChop*)
qed

lemma *ChopAndA*:
$\vdash$ $f; (g \wedge_i g1) \supset_i f; g$
proof −
have *1*: $\vdash (g \wedge_i g1) \supset_i g$ **by** *auto*
from *1* **show** *?thesis* **by** (*rule RightChopImpChop*)
qed

lemma *ChopAndB*:
$\vdash$ $f; (g \wedge_i g1) \supset_i f; g1$
proof −
have *1*: $\vdash (g \wedge_i g1) \supset_i g1$ **by** *auto*
from *1* **show** *?thesis* **by** (*rule RightChopImpChop*)
qed

lemma *BoxStateAndChopEqvChop*:
$\vdash$ $\Box\, (\textit{init } w) \wedge_i (f; g) \equiv_i (\Box\, (\textit{init } w) \wedge_i f); (\Box\, (\textit{init } w) \wedge_i g)$
proof −
have *1*: $\vdash \Box\, (\textit{init } w) \equiv_i$ *ba*$(\Box\, (\textit{init } w))$
by (*rule BoxStateEqvBaBoxState*)

have 2: $\vdash$ *ba*($\Box$ (*init w*)) $\wedge_i$ (f; g) $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ f); ($\Box$ (*init w*) $\wedge_i$ g)
by (*rule BaAndChopImport*)
have 3: $\vdash$ $\Box$ (*init w*) $\wedge_i$ (f; g) $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ f); ($\Box$ (*init w*) $\wedge_i$ g)
using *1 2 prop18* **by** *blast*
have 11: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f); ($\Box$ (*init w*) $\wedge_i$ g) $\supset_i$ ($\Box$ (*init w*)); ($\Box$ (*init w*) $\wedge_i$ g)
by (*rule AndChopA*)
have 12: $\vdash$ ($\Box$ (*init w*)); ($\Box$ (*init w*) $\wedge_i$ g) $\supset_i$ ($\Box$ (*init w*)); ($\Box$ (*init w*))
by (*rule ChopAndA*)
have 13: $\vdash$ ($\Box$ (*init w*)); ($\Box$ (*init w*)) $\equiv_i$ $\Box$ (*init w*)
by (*rule BoxStateChopBoxEqvBox*)
have 14: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f); ($\Box$ (*init w*) $\wedge_i$ g) $\supset_i$ f; ($\Box$ (*init w*) $\wedge_i$ g)
by (*rule AndChopB*)
have 15: $\vdash$ f; ($\Box$ (*init w*) $\wedge_i$ g) $\supset_i$ f; g
by (*rule ChopAndB*)
have 16: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f); ($\Box$ (*init w*) $\wedge_i$ g) $\supset_i$ $\Box$ (*init w*) $\wedge_i$ (f; g)
using *11 12 13 14 15* **using** *itl-prop*(*31*) *itl-prop*(*32*) *prop02* **by** *metis*
from *3 16* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *DiEqvNotBiNot*:
$\vdash$ *di* f $\equiv_i$ $\neg_i$(*bi* $\neg_i$ f)
proof −
have 1: $\vdash$ *bi* $\neg_i$ f $\equiv_i$ $\neg_i$ (*di* $\neg_i$ $\neg_i$ f) **by** (*simp add*: *bi-d-def*)
hence 2: $\vdash$ *di* $\neg_i$ $\neg_i$ f $\equiv_i$ $\neg_i$(*bi* $\neg_i$ f) **by** *auto*
have 3: $\vdash$ f $\equiv_i$ $\neg_i$ $\neg_i$ f **by** *auto*
hence 4: $\vdash$ *di* f $\equiv_i$ *di* $\neg_i$ $\neg_i$ f **by** (*rule DiEqvDi*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *ChopAndBoxImport*:
$\vdash$ f; g $\wedge_i$ $\Box$ h $\supset_i$ f; (g $\wedge_i$ h)
proof −
have 1: $\vdash$ $\Box$ h $\wedge_i$ f; g $\supset_i$ f; (h $\wedge_i$ g) **by** (*rule BoxAndChopImport*)
have 2: $\vdash$ f; (h $\wedge_i$ g) $\equiv_i$ f; (g $\wedge_i$ h) **by** (*rule ChopAndCommute*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *AndChopAndCommute*:
$\vdash$ (f $\wedge_i$ g); ($f1$ $\wedge_i$ $g1$) $\equiv_i$ (g $\wedge_i$ f); ($g1$ $\wedge_i$ $f1$)
proof −
have 1: $\vdash$ (f $\wedge_i$ g); ($f1$ $\wedge_i$ $g1$) $\equiv_i$ (g $\wedge_i$ f); ($f1$ $\wedge_i$ $g1$) **by** (*rule AndChopCommute*)
have 2: $\vdash$ (g $\wedge_i$ f); ($f1$ $\wedge_i$ $g1$) $\equiv_i$ (g $\wedge_i$ f); ($g1$ $\wedge_i$ $f1$) **by** (*rule ChopAndCommute*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *ChopImpChop*:
assumes $\vdash$ $f \supset_i f1$ $\vdash$ $g \supset_i g1$
shows $\vdash$ f; g $\supset_i$ $f1$; $g1$
proof −
have 1: $\vdash$ f $\supset_i$ $f1$ **using** *assms* **by** *auto*

hence *2*: $\vdash f ; g \supset_i f1 ; g$ **by** (*rule LeftChopImpChop*)
have *3*: $\vdash g \supset_i g1$ **using** *assms* **by** *auto*
hence *4*: $\vdash f1 ; g \supset_i f1 ; g1$ **by** (*rule RightChopImpChop*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *ChopEqvChop*:
assumes $\vdash f \equiv_i f1$ $\vdash g \equiv_i g1$
shows $\vdash f ; g \equiv_i f1 ; g1$
proof −
have *1*: $\vdash f \equiv_i f1$ **using** *assms* **by** *auto*
hence *2*: $\vdash f ; g \equiv_i f1 ; g$ **by** (*rule LeftChopEqvChop*)
have *3*: $\vdash g \equiv_i g1$ **using** *assms* **by** *auto*
hence *4*: $\vdash f1 ; g \equiv_i f1 ; g1$ **by** (*rule RightChopEqvChop*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *BoxImpBoxImpBox*:
$\vdash \Box\ h \supset_i \Box(g \supset_i \Box\ h \wedge_i g\ )$
proof −
have *1*: $\vdash \Box\ h \supset_i (g \supset_i \Box\ h \wedge_i g\ )$ **by** *simp*
hence *2*: $\vdash \Box(\Box\ h) \supset_i \Box(g \supset_i \Box\ h \wedge_i g\ )$ **by** (*rule ImpBoxRule*)
have *3*: $\vdash \Box\ h \equiv_i \Box(\Box h)$ **by** (*rule BoxEqvBoxBox*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *BoxChopImpChopBox*:
$\vdash \Box\ h \supset_i f ; g \supset_i f ; (\Box\ h \wedge_i g)$
proof −
have *1*: $\vdash \Box\ h \supset_i \Box(g \supset_i \Box\ h \wedge_i g\ )$ **by** (*rule BoxImpBoxImpBox*)
have *2*: $\vdash \Box(g \supset_i \Box\ h \wedge_i g\ ) \supset_i f ; g \supset_i f ; (\Box\ h \wedge_i g)$ **by** (*rule BoxChopImpChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NotChopEqvYieldsNot*:
$\vdash \neg_i (f ; g) \equiv_i f\ yields\ \neg_i\ g$
proof −
have *1*: $\vdash g \equiv_i \neg_i \neg_i\ g$ **by** *auto*
hence *2*: $\vdash f ; g \equiv_i f ; \neg_i \neg_i\ g$ **by** (*rule RightChopEqvChop*)
hence *3*: $\vdash \neg_i (f ; g) \equiv_i \neg_i (f ; \neg_i \neg_i\ g)$ **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *NotDiFalse*:
$\vdash \neg_i (\ di\ \ false_i)$
proof −
have *1*: $\vdash (init\ true_i) \supset_i bi\ (init\ true_i)$ **by** (*rule StateImpBi*)
hence *2*: $\vdash true_i \supset_i bi\ true_i$ **by** *auto*
have *3*: $\vdash true_i$ **by** *auto*
have *4*: $\vdash bi\ true_i$ **using** *2 3 MP* **by** *auto*

hence *5*: $\vdash \neg_i$ (*di* $\neg_i$ *true*$_i$) **by** *simp*
have *6*: $\vdash \neg_i$ *true*$_i$ $\equiv_i$ *false*$_i$ **by** *auto*
hence *7*: $\vdash$ *di* $\neg_i$ *true*$_i$ $\equiv_i$ *di* *false*$_i$ **by** (*rule DiEqvDi*)
from *5 7* **show** *?thesis* **by** *auto*
qed

lemma *StateAndEmptyChop*:
$\vdash$ ((*init w*) $\wedge_i$ *empty*); *f* $\equiv_i$ (*init w*) $\wedge_i$ *f*
proof −
have *1*: $\vdash$ ((*init w*) $\wedge_i$ *empty*); *f* $\equiv_i$ (*init w*) $\wedge_i$ *empty* ; *f* **by** (*rule StateAndChop*)
have *2*: $\vdash$ *empty* ; *f* $\equiv_i$ *f* **by** (*rule EmptyChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *StateAndNextChop*:
$\vdash$ ((*init w*) $\wedge_i$ $\bigcirc$ *f*); *g* $\equiv_i$ (*init w*) $\wedge_i$ $\bigcirc$(*f*; *g*)
proof −
have *1*: $\vdash$ ((*init w*) $\wedge_i$ $\bigcirc$ *f*); *g* $\equiv_i$ (*init w*) $\wedge_i$ ($\bigcirc$ *f*); *g* **by** (*rule StateAndChop*)
have *2*: $\vdash$ ($\bigcirc$ *f*); *g* $\equiv_i$ $\bigcirc$(*f*; *g*) **by** (*rule NextChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NextAndEqvNextAndNext*:
$\vdash$ $\bigcirc$ (*f* $\wedge_i$ *g*) $\equiv_i$ $\bigcirc$ *f* $\wedge_i$ $\bigcirc$ *g*
by *auto*

lemma *NextStateAndChop*:
$\vdash$ $\bigcirc$(((*init w*) $\wedge_i$ *f*); *g*) $\equiv_i$ $\bigcirc$ (*init w*) $\wedge_i$ $\bigcirc$(*f*; *g*)
proof −
have *1*: $\vdash$ ((*init w*) $\wedge_i$ *f*); *g* $\equiv_i$ (*init w*) $\wedge_i$ *f*; *g* **by** (*rule StateAndChop*)
hence *2*: $\vdash$ $\bigcirc$(((*init w*) $\wedge_i$ *f*); *g*) $\equiv_i$ $\bigcirc$((*init w*) $\wedge_i$ *f*; *g*) **by** (*rule NextEqvNext*)
have *3*: $\vdash$ $\bigcirc$((*init w*) $\wedge_i$ *f*; *g*) $\equiv_i$ $\bigcirc$ (*init w*) $\wedge_i$ $\bigcirc$(*f*; *g*) **by** (*rule NextAndEqvNextAndNext*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *StateYieldsEqv*:
$\vdash$ ((*init w*) $\supset_i$ (*f yields* *g*)) $\equiv_i$ ((*init w*) $\wedge_i$ *f*) *yields* *g*
proof −
have *1*: $\vdash$ ((*init w*) $\wedge_i$ *f*); $\neg_i$ *g* $\equiv_i$ (*init w*) $\wedge_i$ *f*; ($\neg_i$ *g*) **by** (*rule StateAndChop*)
hence *2*: $\vdash$ ((*init w*) $\supset_i$ $\neg_i$ (*f*; $\neg_i$ *g*)) $\equiv_i$ $\neg_i$ (((*init w*) $\wedge_i$ *f*); $\neg_i$ *g*) **by** *auto*
from *2* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *StateAndDi*:
$\vdash$ (*init w*) $\wedge_i$ *di* *f* $\equiv_i$ *di* ((*init w*) $\wedge_i$ *f*)
proof −
have *1*: $\vdash$ ((*init w*) $\wedge_i$ *f*); *true*$_i$$\equiv_i$ (*init w*) $\wedge_i$ *f*; *true*$_i$ **by** (*rule StateAndChop*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiNext*:
$\vdash \quad di(\bigcirc f) \equiv_i \bigcirc (di \ \ f)$
proof –
have *1*: $\vdash (\bigcirc f); \mathit{true}_i \equiv_i \bigcirc(f; \mathit{true}_i)$ **by** (*rule NextChop*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiNextState*:
$\vdash \quad di(\bigcirc (\mathit{init}\ w)) \equiv_i \bigcirc (\mathit{init}\ w)$
proof –
have *1*: $\vdash \ di(\bigcirc (\mathit{init}\ w)) \equiv_i \bigcirc(di \ (\mathit{init}\ w))$ **by** (*rule DiNext*)
have *2*: $\vdash \ di \ (\mathit{init}\ w) \equiv_i (\mathit{init}\ w)$ **by** (*rule DiState*)
hence *3*: $\vdash \bigcirc(di \ (\mathit{init}\ w)) \equiv_i \bigcirc (\mathit{init}\ w)$ **by** (*rule NextEqvNext*)
from *1 3* **show** *?thesis* **by** *auto*
qed

lemma *StateImpBiGen*:
assumes $\vdash \ (\mathit{init}\ w) \supset_i f$
shows $\vdash (\mathit{init}\ w) \supset_i bi \ \ f$
proof –
have *1*: $\vdash (\mathit{init}\ w) \supset_i f$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i \ f \supset_i \neg_i \ (\mathit{init}\ w)$ **by** *auto*
hence *3*: $\vdash \ di \neg_i \ f \supset_i \ di \neg_i \ (\mathit{init}\ w)$ **by** (*rule DiImpDi*)
hence *4*: $\vdash \ di \neg_i \ f \supset_i \ di \ (\mathit{init} \neg_i w)$ **by** *auto*
have *5*: $\vdash \ di \ (\mathit{init} \neg_i w) \equiv_i \ (\mathit{init} \neg_i w)$ **by** (*rule DiState*)
have *6*: $\vdash \ di \neg_i \ f \supset_i \neg_i \ (\mathit{init}\ w)$ **using** *4 5* **by** *auto*
hence *7*: $\vdash (\mathit{init}\ w) \supset_i \neg_i (di \neg_i \ f)$ **by** *auto*
from *7* **show** *?thesis* **by** (*simp add*: *bi-d-def*)
qed

lemma *ChopAndNotChopImp*:
$\vdash \quad f; g \wedge_i \neg_i (f; g1) \supset_i f; (g \wedge_i \neg_i \ g1)$
proof –
have *1*: $\vdash g \supset_i (g \wedge_i \neg_i \ g1) \vee_i \ g1$ **by** *auto*
hence *2*: $\vdash f; g \supset_i f; ((g \wedge_i \neg_i \ g1) \vee_i \ g1)$ **by** (*rule RightChopImpChop*)
have *3*: $\vdash f; ((g \wedge_i \neg_i \ g1) \vee_i \ g1) \supset_i (f; (g \wedge_i \neg_i \ g1)) \vee_i \ (f; g1)$ **by** (*rule ChopOrImp*)
have *4*: $\vdash f; g \supset_i f; (g \wedge_i \neg_i \ g1) \vee_i \ f; g1$ **using** *2 3 MP* **by** *auto*
from *4* **show** *?thesis* **by** *auto*
qed

lemma *ChopAndYieldsImp*:
$\vdash \quad f; g \wedge_i f \ \mathit{yields} \ \ g1 \supset_i f; (g \wedge_i g1)$
proof –
have *1*: $\vdash g \supset_i (g \wedge_i g1) \vee_i \ \neg_i \ g1$ **by** *auto*
hence *2*: $\vdash f; g \supset_i f; ((g \wedge_i \ g1) \vee_i \ \neg_i g1)$ **by** (*rule RightChopImpChop*)
have *3*: $\vdash f; ((g \wedge_i \ g1) \vee_i \ \neg_i g1) \supset_i (f; (g \wedge_i \ g1)) \vee_i \ (f; \neg_i g1)$ **by** (*rule ChopOrImp*)
have *4*: $\vdash f; g \supset_i f; (g \wedge_i \ g1) \vee_i \ f; \neg_i g1$ **using** *2 3 MP* **by** *auto*
hence *5*: $\vdash f; g \wedge_i \neg_i \ (f; \neg_i \ g1) \supset_i f; (g \wedge_i g1)$ **by** *auto*
from *5* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *ChopAndYieldsMP*:
$\vdash\ \ f;\ g \wedge_i f\ yields\ (g \supset_i g1) \supset_i f;\ g1$
proof −
have *1*: $\vdash f;\ g \wedge_i f\ yields\ (g \supset_i g1) \supset_i f;\ (g \wedge_i (g \supset_i g1))$ **by** (*rule ChopAndYieldsImp*)
have *2*: $\vdash g \wedge_i (g \supset_i g1) \supset_i g1$ **by** *auto*
hence *3*: $\vdash f;\ (g \wedge_i (g \supset_i g1)) \supset_i f;\ g1$ **by** (*rule RightChopImpChop*)
from *1 3* **show** *?thesis* **by** *auto*
qed

lemma *OrYieldsImp*:
$\vdash\ \ (f \vee_i\ f1)\ yields\ \ g \equiv_i (f\ yields\ \ g) \wedge_i (f1\ yields\ \ g)$
proof −
have *1*: $\vdash ((f \vee_i\ f1);\ \neg_i\ \ g) \equiv_i ((f;\ \neg_i\ \ g) \vee_i\ \ (f1;\ \neg_i\ \ g))$ **by** (*rule OrChopEqv*)
hence *2*: $\vdash \neg_i\ ((f \vee_i\ f1);\ \neg_i\ \ g) \equiv_i \neg_i\ (f;\ \neg_i\ \ g) \wedge_i\ \ \neg_i(f1;\ \neg_i\ \ g)$ **by** *auto*
from *2* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *LeftYieldsImpYields*:
assumes $\vdash\ \ f \supset_i f1$
shows $\vdash (f1\ yields\ \ g) \supset_i (f\ yields\ \ g)$
proof −
have *1*: $\vdash f \supset_i f1$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ \neg_i\ \ g \supset_i f1;\ \neg_i\ \ g$ **by** (*rule LeftChopImpChop*)
hence *3*: $\vdash \neg_i\ (f1;\ \neg_i\ \ g) \supset_i \neg_i\ (f;\ \neg_i\ \ g)$ **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *LeftYieldsEqvYields*:
assumes $\vdash\ \ f \equiv_i f1$
shows $\vdash (f\ yields\ \ g) \equiv_i (f1\ yields\ \ g)$
proof −
have *1*: $\vdash f \equiv_i f1$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ \neg_i\ \ g \equiv_i f1;\ \neg_i\ \ g$ **by** (*rule LeftChopEqvChop*)
hence *3*: $\vdash \neg_i\ (f;\ \neg_i\ \ g) \equiv_i \neg_i\ (f1;\ \neg_i\ \ g)$ **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

5.6 Properties of Fin

lemma *FinEqvTrueChopAndEmpty*:
$\vdash fin\ f \equiv_i true_i;(f \wedge_i empty)$
proof −
have *1*: $\vdash fin\ f \equiv_i \Box(empty \supset_i f)$ **by** (*simp add*: *fin-d-def*)
have *2*: $\vdash \Box(empty \supset_i f) \equiv_i \neg_i(\Diamond(\neg_i(empty \supset_i f\ )\ )\ )$ **by** (*simp add*: *always-d-def*)
have *3*: $\vdash (\neg_i(empty \supset_i f\ )) \equiv_i (\neg_i\ f \wedge_i empty)$ **by** *auto*
hence *4*: $\vdash \Diamond(\neg_i(empty \supset_i f\ )) \equiv_i \Diamond(\neg_i\ f \wedge_i empty)$ **using** *DiamondEqvDiamond* **by** *blast*
hence *5*: $\vdash \neg_i(\Diamond(\neg_i(empty \supset_i f\ ))) \equiv_i \neg_i(\Diamond(\neg_i\ f \wedge_i empty))$ **by** *auto*
have *6*: $\vdash \neg_i(\Diamond(\neg_i\ f \wedge_i empty)) \equiv_i true_i;(f \wedge_i empty)$ **using** *Finprop* **by** *auto*
from *1 2 5 6* **show** *?thesis* **by** *auto*

qed

lemma *DiamondFin*:
$\vdash \Diamond(\mathit{fin}\ w) \equiv_i \mathit{fin}\ w$
by (*metis DiamondDiamondEqvDiamond DiamondEqvDiamond FinEqvTrueChopAndEmpty itl-prop*(*30*) *prop03 sometimes-d-def*)

lemma *ChopFinExportA*:
$\vdash f;(g \wedge_i \mathit{fin}\ w) \supset_i \mathit{fin}\ w$
using *DiamondFin* **by** *auto*

lemma *FinImpBox*:
$\vdash \mathit{fin}\ w \supset_i \Box(\mathit{fin}\ w)$
by (*metis BoxImpBoxBox fin-d-def*)

lemma *FinAndChopImport*:
$\vdash (\mathit{fin}\ w) \wedge_i (f;g) \supset_i f;((\mathit{fin}\ w) \wedge_i g)$
proof −
have *1*: $\vdash \mathit{fin}\ w \supset_i \Box(\mathit{fin}\ w)$ **by** (*rule FinImpBox*)
hence *2*: $\vdash \mathit{fin}\ w \wedge_i f;g \supset_i \Box(\mathit{fin}\ w) \wedge_i (f;g)$ **by** *auto*
have *3*: $\vdash \Box(\mathit{fin}\ w) \wedge_i (f;g) \supset_i f;((\mathit{fin}\ w) \wedge_i g)$ **using** *BoxAndChopImport* **by** *blast*
from *2 3* **show** *?thesis* **using** *MP* **by** *auto*
qed

lemma *FinAndChop*:
$\vdash f;(g \wedge_i \mathit{fin}\ w) \equiv_i \mathit{fin}\ w \wedge_i f;g$
using *FinAndChopImport ChopFinExportA ChopAndA ChopAndCommute itl-prop*(*31*) *itl-prop*(*32*) *prop15*
by *blast*

lemma *ChopAndEmptyEqvEmptyChopEmpty*:
$\vdash ((f;g) \wedge_i \mathit{empty}) \equiv_i (f \wedge_i \mathit{empty});(g \wedge_i \mathit{empty})$
by *auto*

lemma *FinAndEmpty*:
$\vdash (\mathit{fin}\ w) \wedge_i \mathit{empty} \equiv_i w \wedge_i \mathit{empty}$
proof −
have *1*: $\vdash (\mathit{fin}\ w) \wedge_i \mathit{empty} \equiv_i \mathit{true}_i;(w \wedge_i \mathit{empty}) \wedge_i \mathit{empty}$
using *FinEqvTrueChopAndEmpty* **by** *auto*
have *2*: $\vdash \mathit{true}_i;(w \wedge_i \mathit{empty}) \wedge_i \mathit{empty} \equiv_i (\mathit{true}_i \wedge_i \mathit{empty});(w \wedge_i \mathit{empty})$
using *ChopAndEmptyEqvEmptyChopEmpty* **by** *auto*
have *3*: $\vdash (\mathit{true}_i \wedge_i \mathit{empty});(w \wedge_i \mathit{empty}) \equiv_i \mathit{empty};(w \wedge_i \mathit{empty})$
using *LeftChopEqvChop itl-prop*(*17*) **by** *blast*
have *4*: $\vdash \mathit{empty};(w \wedge_i \mathit{empty}) \equiv_i w \wedge_i \mathit{empty}$
using *EmptyChop* **by** *blast*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *AndFinEqvChopAndEmpty*:
$\vdash f \wedge_i \mathit{fin}\ g \equiv_i f; (g \wedge_i \mathit{empty})$

proof −
have *1*: $\vdash f \wedge_i \ fin\ g \equiv_i f;empty \wedge_i fin\ g$ **using** *ChopEmpty itl-prop*(*30*) *prop06* **by** *blast*
have *2*: $\vdash fin\ g \wedge_i f;empty \equiv_i f;(empty \wedge_i fin\ g)$ **using** *FinAndChop* **using** *itl-prop*(*30*) **by** *blast*
have *3*: $\vdash empty \wedge_i fin\ g \equiv_i \ fin\ g \wedge_i empty$ **by** *auto*
have *4*: $\vdash fin\ g \wedge_i empty \equiv_i g \wedge_i empty$ **using** *FinAndEmpty* **by** *auto*
have *5*: $\vdash empty \wedge_i fin\ g \equiv_i g \wedge_i empty$ **using** *3 4* **by** *auto*
hence *6*: $\vdash f;(empty \wedge_i fin\ g) \equiv_i f;(g \wedge_i empty)$ **using** *RightChopEqvChop* **by** *blast*
from *1 2 5* **show** *?thesis* **by** *auto*
qed

lemma *AndFinEqvChopStateAndEmpty*:
$\vdash \ f \wedge_i \ fin\ \ (init\ w) \equiv_i f;\ ((init\ w) \wedge_i \ empty\ )$
using *AndFinEqvChopAndEmpty* **by** *blast*

lemma *FinStateEqvStateAndEmptyOrNextFinState*:
$\vdash \ fin\ \ (init\ w) \equiv_i ((init\ w) \wedge_i \ empty\ ) \vee_i \ \bigcirc(\ fin\ \ (init\ w))$
proof −
have *1*: $\vdash fin\ (init\ w) \equiv_i \Box(\ empty \supset_i init\ w)$
by (*simp add*: *fin-d-def*)
have *2* : $\vdash \Box(empty \supset_i init\ w) \equiv_i$
$(empty \supset_i init\ w) \wedge_i wnext\ (\Box\ (empty \supset_i init\ w))$
by (*rule BoxEqvAndWnextBox*)
have *3*: $\vdash fin\ (init\ w) \equiv_i \ (empty \supset_i init\ w) \wedge_i wnext\ (fin\ (init\ w))$
using *1 2* **by** (*simp add*: *fin-d-def*)
have *4*: $\vdash wnext\ (fin\ (init\ w)) \equiv_i empty \vee_i \bigcirc\ (fin\ (init\ w))$
by (*rule WnextEqvEmptyOrNext*)
have *5*: $\vdash fin\ (init\ w) \equiv_i \ (empty \supset_i init\ w) \wedge_i (empty \vee_i \bigcirc\ (fin\ (init\ w)))$
using *3 4* **by** (*simp add*: *fin-d-def*)
have *6*: $\vdash (empty \supset_i init\ w) \wedge_i (empty \vee_i \bigcirc\ (fin\ (init\ w))) \equiv_i$
$((empty \supset_i init\ w) \wedge_i empty) \vee_i ((empty \supset_i init\ w) \wedge_i \bigcirc\ (fin\ (init\ w)))$
by *auto*
have *7*: $\vdash (empty \supset_i init\ w) \wedge_i empty \equiv_i (init\ w) \wedge_i empty$
by *auto*
have *8*: $\vdash (empty \supset_i init\ w) \wedge_i \bigcirc\ (fin\ (init\ w)) \equiv_i \ \bigcirc\ (fin\ (init\ w))$
by *auto*
have *9*: $\vdash ((empty \supset_i init\ w) \wedge_i empty) \vee_i ((empty \supset_i init\ w) \wedge_i \bigcirc\ (fin\ (init\ w))) \equiv_i$
$((init\ w) \wedge_i \ empty\ ) \vee_i \ \bigcirc(\ fin\ \ (init\ w))$
using *7 8* **by** *auto*
from *5 6 9* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *FinChopEqvOr*:
$\vdash \ \ (\ fin\ \ (init\ w));\ f \equiv_i ((init\ w) \wedge_i f) \vee_i \ \bigcirc((\ fin\ \ (init\ w));\ f)$
proof −
have *1*: $\vdash \ fin\ \ (init\ w) \equiv_i ((init\ w) \wedge_i \ empty\ ) \vee_i \ \bigcirc(\ fin\ \ (init\ w))$
by (*rule FinStateEqvStateAndEmptyOrNextFinState*)
hence *2*: $\vdash (\ fin\ \ (init\ w));\ f \equiv_i (((init\ w) \wedge_i \ empty\ )\vee_i \ \bigcirc(\ fin\ \ (init\ w)));\ f$
by (*rule LeftChopEqvChop*)
have *3*: $\vdash (((init\ w) \wedge_i \ empty\ )\vee_i \ \bigcirc\ (fin\ \ (init\ w)));\ f$
$\equiv_i ((init\ w) \wedge_i \ empty\ );\ f \vee_i \ (\bigcirc\ (fin\ \ (init\ w)));\ f$

by (*rule OrChopEqv*)
have *4*: $\vdash$ ((*init w*) $\wedge_i$ *empty*); *f* $\equiv_i$ (*init w*) $\wedge_i$ *f*
by (*rule StateAndEmptyChop*)
have *5*: $\vdash$ ($\bigcirc$ (*fin* (*init w*))); *f* $\equiv_i$ $\bigcirc$((*fin* (*init w*)); *f*)
by (*rule NextChop*)
from *2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *FinChopEqvDiamond*:
$\vdash$ (*fin* (*init w*)); *f* $\equiv_i$ $\diamond$ ((*init w*) $\wedge_i$ *f*)
proof −
have *1*: $\vdash$ (*fin* (*init w*)) $\equiv_i$ (*true*$_i$;((*init w*) $\wedge_i$ *empty*))
by (*rule FinEqvTrueChopAndEmpty*)
hence *2*: $\vdash$ (*fin* (*init w*));*f* $\equiv_i$ (*true*$_i$;((*init w*) $\wedge_i$ *empty*));*f*
by (*rule LeftChopEqvChop*)
have *3*: $\vdash$ *true*$_i$;(((*init w*) $\wedge_i$ *empty*);*f*) $\equiv_i$ (*true*$_i$;((*init w*) $\wedge_i$ *empty*));*f*
by (*rule ChopAssoc*)
have *4*: $\vdash$ *true*$_i$;(((*init w*) $\wedge_i$ *empty*);*f*)$\equiv_i$ $\diamond$ (((*init w*) $\wedge_i$ *empty*);*f*)
by (*simp add*: *sometimes-d-def*)
have *5*: $\vdash$ ((*init w*) $\wedge_i$ *empty*);*f* $\equiv_i$ (*init w*) $\wedge_i$ *f*
using *StateAndEmptyChop* **by** *blast*
hence *6*: $\vdash$ $\diamond$ (((*init w*) $\wedge_i$ *empty*);*f*) $\equiv_i$ $\diamond$ ((*init w*) $\wedge_i$ *f*)
by (*rule DiamondEqvDiamond*)
from *2 3 4 6* **show** *?thesis* **by** *simp*
qed

lemma *NotDiamondAndNot*:
$\vdash$ $\neg_i$($\diamond$ (*f* $\wedge_i$ $\neg_i$ *f*))
by *simp*

lemma *FinYields*:
$\vdash$ (*fin* (*init w*)) *yields* (*init w*)
proof −
have *1*: $\vdash$ (*fin* (*init w*)); $\neg_i$(*init w*) $\equiv_i$ $\diamond$((*init w*) $\wedge_i$ $\neg_i$(*init w*)) **by** (*rule FinChopEqvDiamond*)
have *2*: $\vdash$ $\neg_i$($\diamond$((*init w*) $\wedge_i$ $\neg_i$ (*init w*))) **by** (*rule NotDiamondAndNot*)
have *3*: $\vdash$ $\neg_i$ ((*fin* (*init w*)); $\neg_i$ (*init w*)) **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *ImpAndFinStateOrFinNotState*:
$\vdash$ *f* $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)) $\vee_i$ *fin* $\neg_i$ (*init w*)
by (*simp*)

lemma *AndFinChopEqvStateAndChop*:
$\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); *g* $\equiv_i$ *f*; ((*init w*) $\wedge_i$ *g*)
proof −
have *1*: $\vdash$ (*fin* (*init w*)) *yields* (*init w*)
by (*rule FinYields*)
have *2*: $\vdash$ *f* $\wedge_i$ *fin* (*init w*) $\supset_i$ *fin* (*init w*)
by *auto*

hence *3*: $\vdash$ (*fin* (*init w*)) *yields* (*init w*) $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)) *yields* (*init w*)
  by (*rule LeftYieldsImpYields*)
have *4*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)) *yields* (*init w*)
  using *1 3 MP* **by** *auto*
have *5*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); *g* $\wedge_i$ (*f* $\wedge_i$ *fin* (*init w*)) *yields* (*init w*)
    $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)); (*g* $\wedge_i$ (*init w*))
  by (*rule ChopAndYieldsImp*)
have *6*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); *g* $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)); (*g* $\wedge_i$ (*init w*))
  using *4 5* **by** *auto*
have *7*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); (*g* $\wedge_i$ (*init w*)) $\supset_i$ *f*; (*g* $\wedge_i$ (*init w*))
  by (*rule AndChopA*)
have *8*: $\vdash$ *g* $\wedge_i$ (*init w*) $\supset_i$ (*init w*) $\wedge_i$ *g*
  by *auto*
hence *9*: $\vdash$ *f*; (*g* $\wedge_i$ (*init w*)) $\supset_i$ *f*; ((*init w*) $\wedge_i$ *g*)
  by (*rule RightChopImpChop*)
have *10*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); *g* $\supset_i$ *f*; ((*init w*) $\wedge_i$ *g*)
  using *6 7 9* **by** *auto*
have *11*: $\vdash$ *f* $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)) $\vee_i$ *fin* $\neg_i$ (*init w*)
  by (*rule ImpAndFinStateOrFinNotState*)
hence *12*: $\vdash$ *f*; ((*init w*) $\wedge_i$ *g*) $\supset_i$
    ((*f* $\wedge_i$ *fin* (*init w*))$\vee_i$ *fin* $\neg_i$ (*init w*)); ((*init w*) $\wedge_i$ *g*)
  by (*rule LeftChopImpChop*)
have *13*: $\vdash$ ((*f* $\wedge_i$ *fin* (*init w*))$\vee_i$ *fin* $\neg_i$ (*init w*)); ((*init w*) $\wedge_i$ *g*)
    $\equiv_i$
    (*f* $\wedge_i$ *fin* (*init w*)); ((*init w*) $\wedge_i$ *g*) $\vee_i$ (*fin* $\neg_i$ (*init w*)); ((*init w*) $\wedge_i$ *g*)
  by (*rule OrChopEqv*)
have *14*: $\vdash$ (*fin* (*init* ($\neg_i$ *w*))); ((*init w*) $\wedge_i$ *g*) $\supset_i$ $\diamond$((*init* ($\neg_i$ *w*)) $\wedge_i$ ((*init w*) $\wedge_i$ *g*))
  using *FinChopEqvDiamond itl-prop*(*31*) **by** *blast*
have *141*: $\vdash$ $\neg_i$($\diamond$((*init* ($\neg_i$ *w*)) $\wedge_i$ ((*init w*) $\wedge_i$ *g*))) $\supset_i$
    $\neg_i$ ((*fin* (*init* ($\neg_i$ *w*))); ((*init w*) $\wedge_i$ *g*))
  using *14* **by** *auto*
have *15*: $\vdash$ $\neg_i$($\diamond$((*init* ($\neg_i$ *w*)) $\wedge_i$ ((*init w*) $\wedge_i$ *g*)))
  using *NotDiamondAndNot Initprop* **by** *simp*
have *151*: $\vdash$ $\neg_i$ ((*fin* (*init* ($\neg_i$ *w*))); ((*init w*) $\wedge_i$ *g*))
  using *15 141* **by** *auto*
have *152*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); ((*init w*) $\wedge_i$ *g*) $\vee_i$ (*fin* $\neg_i$ (*init w*)); ((*init w*) $\wedge_i$ *g*) $\supset_i$
    (*f* $\wedge_i$ *fin* (*init w*)); ((*init w*) $\wedge_i$ *g*)
  using *151* **by** *auto*
have *16*: $\vdash$ *f*; ((*init w*) $\wedge_i$ *g*) $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)); ((*init w*) $\wedge_i$ *g*)
  using *12 13 152* **by** *fastforce*
have *17*: $\vdash$ (*f* $\wedge_i$ *fin* (*init w*)); ((*init w*) $\wedge_i$ *g*) $\supset_i$ (*f*$\wedge_i$ *fin* (*init w*)); *g*
  by (*rule ChopAndB*)
have *18*: $\vdash$ *f*; ((*init w*) $\wedge_i$ *g*) $\supset_i$ (*f* $\wedge_i$ *fin* (*init w*)); *g*
  using *16 17* **by** *auto*
from *10 18* **show** *?thesis* **by** *auto*
qed

lemma *DiAndFinEqvChopState*:
$\vdash$ *di* (*f* $\wedge_i$ *fin* (*init w*)) $\equiv_i$ *f*; (*init w*)
proof –

have *1*: ⊢ $(f \wedge_i fin(init\ w));\ true_i \equiv_i f;((init\ w) \wedge_i true_i)$ **by** (*rule AndFinChopEqvStateAndChop*)
have *2*: ⊢ $(init\ w) \wedge_i true_i \equiv_i (init\ w)$ **by** *auto*
hence *3*: ⊢ $f;\ ((init\ w) \wedge_i true_i) \equiv_i f;\ (init\ w)$ **by** (*rule RightChopEqvChop*)
have *4*: ⊢ $(f \wedge_i fin\ (init\ w));\ true_i \equiv_i f;\ (init\ w)$ **using** *1 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *FinNotStateEqvNotFinState*:
⊢ $fin\ (init\ \neg_i\ w) \equiv_i \neg_i(fin\ (init\ w))$
by (*simp*)

lemma *BiImpFinEqvYieldsState*:
⊢ $bi\ (f \supset_i fin\ (init\ w)) \equiv_i f\ yields\ (init\ w)$
proof −
have *1*: ⊢ $di\ (f \wedge_i fin\ (init\ \neg_i\ w)) \equiv_i f;\ (init\ \neg_i\ w)$ **by** (*rule DiAndFinEqvChopState*)
have *2*: ⊢ $f \wedge_i fin(init\ \neg_i\ w) \equiv_i f \wedge_i \neg_i(fin(init\ w))$ **using** *FinNotStateEqvNotFinState* **by** *auto*
have *3*: ⊢ $f \wedge_i \neg_i\ (fin(init\ w)) \equiv_i \neg_i\ (f \supset_i fin\ (init\ w))$ **by** *auto*
have *4*: ⊢ $f \wedge_i fin(init\ \neg_i\ w) \equiv_i \neg_i\ (f \supset_i fin(init\ w))$ **using** *2 3* **by**(*simp add*: *fin-d-def*)
hence *5*: ⊢ $di\ (f \wedge_i fin\ (init\ \neg_i\ w)) \equiv_i di\ \neg_i\ (f \supset_i fin(init\ w))$ **by** (*rule DiEqvDi*)
have *6*: ⊢ $di\ \neg_i\ (f \supset_i fin\ (init\ w)) \equiv_i \neg_i(bi\ (f \supset_i fin(init\ w)))$ **by** (*rule DiNotEqvNotBi*)
have *7*: ⊢ $\neg_i\ (bi\ (f \supset_i fin\ (init\ w))) \equiv_i f;(init\ \neg_i\ w)$ **using** *1 5 6 Initprop* **by** *auto*
hence *8*: ⊢ $bi\ (f \supset_i fin\ (init\ w)) \equiv_i \neg_i\ (f;\ \neg_i\ (init\ w))$ **by** *auto*
from *8* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *StateImpYields*:
assumes ⊢ $(init\ w) \wedge_i f \supset_i fin\ (init\ w1)$
shows ⊢ $(init\ w) \supset_i (f\ yields\ (init\ w1))$
proof −
have *1*: ⊢ $(init\ w) \wedge_i f \supset_i fin\ (init\ w1)$ **using** *assms* **by** *auto*
hence *2*: ⊢ $(init\ w) \supset_i (f \supset_i fin\ (init\ w1))$ **by** *auto*
hence *3*: ⊢ $(init\ w) \supset_i bi\ (f \supset_i fin\ (init\ w1))$ **by** (*rule StateImpBiGen*)
have *4*: ⊢ $bi\ (f \supset_i fin\ (init\ w1)) \equiv_i f\ yields\ (init\ w1)$ **by** (*rule BiImpFinEqvYieldsState*)
from *3 4* **show** *?thesis* **by** *auto*
qed

lemma *StateAndYieldsImpYields*:
assumes ⊢ $(init\ w) \wedge_i f \supset_i f1$
shows ⊢ $(init\ w) \wedge_i (f1\ yields\ g) \supset_i (f\ yields\ g)$
proof −
have *1*: ⊢ $(init\ w) \wedge_i f \supset_i f1$ **using** *assms* **by** *auto*
hence *2*: ⊢ $(init\ w) \wedge_i (f;\ \neg_i\ g) \supset_i f1;\ \neg_i\ g$ **by** (*rule StateAndChopImpChopRule*)
hence *3*: ⊢ $(init\ w) \wedge_i \neg_i\ (f1;\ \neg_i\ g) \supset_i \neg_i\ (f;\ \neg_i\ g)$ **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *AndYieldsA*:
⊢ $f\ yields\ g \supset_i (f \wedge_i f1)\ yields\ g$
proof −
have *1*: ⊢ $f \wedge_i f1 \supset_i f$ **by** *auto*

from *1* **show** *?thesis* **by** (*rule LeftYieldsImpYields*)
qed

lemma *AndYieldsB*:
$\vdash$ *f1 yields* $g \supset_i (f \wedge_i f1)$ *yields* g
proof −
have *1*: $\vdash f \wedge_i f1 \supset_i f1$ **by** *auto*
from *1* **show** *?thesis* **by** (*rule LeftYieldsImpYields*)
qed

lemma *RightYieldsImpYields*:
assumes $\vdash g \supset_i g1$
shows $\vdash$ (*f yields* g) $\supset_i$ (*f yields* $g1$)
proof −
have *1*: $\vdash g \supset_i g1$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i g1 \supset_i \neg_i g$ **by** *auto*
hence *3*: $\vdash f; \neg_i g1 \supset_i f; \neg_i g$ **by** (*rule RightChopImpChop*)
hence *4*: $\vdash \neg_i (f; \neg_i g) \supset_i \neg_i (f; \neg_i g1)$ **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *RightYieldsEqvYields*:
assumes $\vdash g \equiv_i g1$
shows $\vdash$ (*f yields* g) $\equiv_i$ (*f yields* $g1$)
proof −
have *1*: $\vdash g \equiv_i g1$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i g \equiv_i \neg_i g1$ **by** *auto*
hence *3*: $\vdash f; \neg_i g \equiv_i f; \neg_i g1$ **by** (*rule RightChopEqvChop*)
hence *4*: $\vdash \neg_i (f; \neg_i g) \equiv_i \neg_i (f; \neg_i g1)$ **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *BoxImpYields*:
$\vdash \Box g \supset_i$ *f yields* g
proof −
have *1*: $\vdash f; \neg_i g \supset_i \Diamond \neg_i g$ **by** (*rule ChopImpDiamond*)
hence *2*: $\vdash \neg_i (\Diamond \neg_i g) \supset_i \neg_i (f; \neg_i g)$ **by** *auto*
from *2* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *BoxEqvTrueYields*:
$\vdash \Box f \equiv_i true_i$ *yields* f
proof −
have *1*: $\vdash true_i; \neg_i f \equiv_i \Diamond \neg_i f$ **by** (*rule TrueChopEqvDiamond*)
hence *2*: $\vdash \neg_i (true_i; \neg_i f) \equiv_i \neg_i (\Diamond \neg_i f)$ **by** *auto*
have *3*: $\vdash \Box f \equiv_i \neg_i (\Diamond \neg_i f)$ **by** (*simp add*: *always-d-def*)
have *4*: $\vdash \Box f \equiv_i \neg_i (true_i; \neg_i f)$ **using** *2 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *YieldsGen*:
assumes $\vdash\ g$
shows $\vdash f$ *yields* g
proof –
have *1*: $\vdash g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \Box\, g$ **by** (*rule BoxGen*)
have *3*: $\vdash \Box\, g \supset_i f$ *yields* g **by** (*rule BoxImpYields*)
from *2 3* **show** *?thesis* **using** *MP* **by** *auto*
qed

lemma *YieldsAndYieldsEqvYieldsAnd*:
$\vdash\ (f\ \textit{yields}\ g) \wedge_i (f\ \textit{yields}\ g1) \equiv_i f\ \textit{yields}\ (g \wedge_i g1)$
proof –
have *1*: $\vdash f; (\neg_i\ g \vee_i\ \neg_i\ g1) \equiv_i (f; \neg_i\ g) \vee_i\ (f; \neg_i\ g1)$ **by** (*rule ChopOrEqv*)
hence *2*: $\vdash (f; \neg_i\ g) \vee_i\ (f; \neg_i\ g1) \equiv_i f; (\neg_i\ g \vee_i\ \neg_i\ g1)$ **by** *auto*
have *3*: $\vdash \neg_i\ g \vee_i\ \neg_i\ g1\ \equiv_i\ \neg_i (g \wedge_i g1)$ **by** *auto*
hence *4*: $\vdash f; (\neg_i\ g \vee_i\ \neg_i\ g1)\ \equiv_i\ f; \neg_i (g \wedge_i g1)$ **by** (*rule RightChopEqvChop*)
have *5*: $\vdash (f; \neg_i\ g) \vee_i\ (f; \neg_i\ g1) \equiv_i f; \neg_i (g \wedge_i g1)$ **using** *2 4* **by** *auto*
hence *6*: $\vdash \neg_i (f; \neg_i\ g) \wedge_i \neg_i (f; \neg_i\ g1) \equiv_i \neg_i (f; \neg_i (g \wedge_i g1))$ **by** *auto*
from *6* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *YieldsAndYieldsImpAndYieldsAnd*:
$\vdash\ (f\ \textit{yields}\ g) \wedge_i (f1\ \textit{yields}\ g1) \supset_i (f \wedge_i f1)\ \textit{yields}\ (g \wedge_i g1)$
proof –
have *1*: $\vdash f\ \textit{yields}\ g \supset_i (f \wedge_i f1)\ \textit{yields}\ g$
by (*rule AndYieldsA*)
have *2*: $\vdash f1\ \textit{yields}\ g1 \supset_i (f \wedge_i f1)\ \textit{yields}\ g1$
by (*rule AndYieldsB*)
have *3*: $\vdash (f \wedge_i f1)\ \textit{yields}\ g \wedge_i (f \wedge_i f1)\ \textit{yields}\ g1 \equiv_i (f \wedge_i f1)\ \textit{yields}\ (g \wedge_i g1)$
by (*rule YieldsAndYieldsEqvYieldsAnd*)
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *YieldsYieldsEqvChopYields*:
$\vdash\ f\ \textit{yields}\ (g\ \textit{yields}\ h) \equiv_i (f; g)\ \textit{yields}\ h$
proof –
have *1*: $\vdash f; (g; \neg_i\ h) \equiv_i (f; g); \neg_i\ h$ **by** (*rule ChopAssoc*)
hence *2*: $\vdash f; (g; \neg_i\ h) \equiv_i (f; g); \neg_i\ h$ **by** *auto*
have *3*: $\vdash g; \neg_i\ h \equiv_i \neg_i \neg_i (g; \neg_i\ h)$ **by** *auto*
hence *4*: $\vdash f; (g; \neg_i\ h) \equiv_i f; \neg_i \neg_i (g; \neg_i\ h)$ **by** (*rule RightChopEqvChop*)
have *5*: $\vdash f; \neg_i \neg_i (g; \neg_i\ h) \equiv_i (f; g); \neg_i\ h$ **using** *2 4* **by** *auto*
hence *6*: $\vdash f; \neg_i (g\ \textit{yields}\ h) \equiv_i (f; g); \neg_i\ h$ **by** (*simp add*: *yields-d-def*)
hence *7*: $\vdash \neg_i (f; \neg_i (g\ \textit{yields}\ h)) \equiv_i \neg_i ((f; g); \neg_i\ h)$ **by** *auto*
from *7* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *EmptyYields*:
$\vdash\ \textit{empty}\ \ \textit{yields}\ f \equiv_i f$
proof –

have *1*: $\vdash$ *empty* ; $\neg_i$ $f \equiv_i \neg_i$ f **by** (*rule EmptyChop*)
hence *2*: $\vdash \neg_i$ (*empty* ; $\neg_i$ f) $\equiv_i f$ **by** *auto*
from *2* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *NextYields*:
$\vdash$ ($\bigcirc$ f) *yields* $g \equiv_i$ *wnext* (f *yields* g)
proof −
have *1*: $\vdash$ ($\bigcirc$ f); $\neg_i$ $g \equiv_i \bigcirc(f$; $\neg_i$ g) **by** (*rule NextChop*)
hence *2*: $\vdash \neg_i$ (($\bigcirc$ f); $\neg_i$ g) $\equiv_i \neg_i$ ($\bigcirc(f$; $\neg_i$ g)) **by** *auto*
hence *3*: $\vdash$ ($\bigcirc$ f) *yields* $g \equiv_i \neg_i$ ($\bigcirc(f$; $\neg_i$ g)) **by** (*simp add*: *yields-d-def*)
have *4*: $\vdash \neg_i$($\bigcirc(f$; $\neg_i$ g)) $\equiv_i$ *wnext* $\neg_i$ (f; $\neg_i$ g) **by** *auto*
have *5*: $\vdash$ ($\bigcirc$ f) *yields* $g \equiv_i$ *wnext* $\neg_i$ (f; $\neg_i$ g) **using** *3 4* **by** *auto*
from *5* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *SkipChopEqvNext*:
$\vdash$ *skip* ; $f \equiv_i \bigcirc$ f
by (*simp add*: *next-d-def*)

lemma *SkipYieldsEqvWeakNext*:
$\vdash$ *skip* *yields* $f \equiv_i$ *wnext* f
proof −
have *1*: $\vdash$ *skip* ; $\neg_i$ $f \equiv_i \bigcirc\neg_i$ f **by** (*rule SkipChopEqvNext*)
hence *2*: $\vdash \neg_i$ (*skip* ; $\neg_i$ f) $\equiv_i \neg_i$($\bigcirc\neg_i$ f) **by** *auto*
have *3*: $\vdash \neg_i$ ($\bigcirc\neg_i$ f) $\equiv_i$ *wnext* f **by** *auto*
have *4*: $\vdash \neg_i$ (*skip* ; $\neg_i$ f) $\equiv_i$ *wnext* f **using** *2 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *NextImpSkipYields*:
$\vdash$ $\bigcirc$ $f \supset_i$ *skip* *yields* f
proof −
have *1*: $\vdash \bigcirc$ $f \supset_i$ *wnext* f **by** *auto*
have *2*: $\vdash$ *skip* *yields* $f \equiv_i$ *wnext* f **by** (*rule SkipYieldsEqvWeakNext*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MoreEqvSkipChopTrue*:
$\vdash$ *more* $\equiv_i$ *skip* ; $true_i$
proof −
have *1*: $\vdash$ *skip* ; $true_i \equiv_i \bigcirc true_i$ **by** (*rule SkipChopEqvNext*)
hence *2*: $\vdash \bigcirc true_i \equiv_i$ *skip* ; $true_i$ **by** *auto*
from *2* **show** *?thesis* **by** (*simp add*: *more-d-def*)
qed

lemma *MoreChopImpMore*:
$\vdash$ *more* ; $f \supset_i$ *more*
proof −
have *1*: $\vdash$ ($\bigcirc true_i$); $f \equiv_i \bigcirc(true_i$; f) **by** (*rule NextChop*)

have *2*: $\vdash \bigcirc(true_i;\ f) \supset_i\ more$ **by** *auto*
have *3*: $\vdash (\bigcirc true_i;\ f) \supset_i\ more$ **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*metis more-d-def*)
qed

lemma *ChopMoreImpMore*:
$\vdash\ f;\ more \supset_i\ more$
proof −
have *1*: $\vdash f;\ more \supset_i \Diamond\ more$ **by** (*rule ChopImpDiamond*)
have *2*: $\vdash \Diamond\ more \supset_i\ more$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MoreChopEqvNextDiamond*:
$\vdash\ more\ ;\ f \equiv_i \bigcirc(\Diamond\ f)$
proof −
have *1*: $\vdash\ more\ ;\ f \equiv_i (\bigcirc\ true_i);\ f$ **by** (*simp add*: *more-d-def*)
have *2*: $\vdash (\bigcirc true_i);\ f \equiv_i \bigcirc(true_i;\ f)$ **by** (*rule NextChop*)
have *3*: $\vdash\ more\ ;\ f \equiv_i \bigcirc(true_i;\ f)$ **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *sometimes-d-def*)
qed

lemma *WeakNextBoxImpMoreYields*:
$\vdash\ more\ \ yields\ \ f \equiv_i wnext(\ \Box\ f)$
proof −
have *1*: $\vdash more\ ;\ \neg_i\ f \equiv_i \bigcirc(\Diamond\ \neg_i f)$ **by** (*rule MoreChopEqvNextDiamond*)
have *2*: $\vdash \bigcirc(\Diamond\ \neg_i f) \equiv_i \bigcirc(\neg_i(\Box f))$ **by** *auto*
have *3*: $\vdash \bigcirc(\neg_i(\Box f)) \equiv_i \neg_i\ (\ wnext(\ \Box\ f)\ )$ **by** *auto*
have *4*: $\vdash more\ ;\ \neg_i\ f \equiv_i \neg_i(more\ \ yields\ \ f)$ **by** (*metis itl-prop*(*30*) *itl-prop*(*4*) *yields-d-def*)
from *1 2 3 4* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *NotEqvYieldsMore*:
$\vdash\ \neg_i\ f \equiv_i f\ yields\ \ more$
proof −
have *1*: $\vdash f;\ empty \equiv_i f$ **by** (*rule ChopEmpty*)
hence *2*: $\vdash \neg_i\ (f;\ empty\ ) \equiv_i \neg_i\ f$ **by** *auto*
have *3*: $\vdash\ empty \equiv_i \neg_i\ more$ **by** *auto*
hence *4*: $\vdash f;\ empty \equiv_i f;\ \neg_i\ more$ **by** (*rule RightChopEqvChop*)
hence *5*: $\vdash \neg_i\ (f;\ empty\ ) \equiv_i \neg_i\ (f;\ \neg_i\ more\ )$ **by** *auto*
have *6*: $\vdash \neg_i\ f \equiv_i \neg_i\ (f;\ \neg_i\ more\ )$ **using** *2 5* **by** *auto*
from *6* **show** *?thesis* **by** (*metis yields-d-def*)
qed

lemma *LeftChopImpMoreRule*:
assumes $\vdash\ f \supset_i\ more$
shows $\vdash f;\ g \supset_i\ more$
proof −
have *1*: $\vdash f \supset_i\ more$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;\ g \supset_i\ more\ ;\ g$ **by** (*rule LeftChopImpChop*)

have *3*: $\vdash$ *more* ; $g \supset_i$ *more* **by** (*rule MoreChopImpMore*)
from *2 3* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *RightChopImpMoreRule*:
assumes $\vdash$ $g \supset_i$ *more*
shows $\vdash f; g \supset_i$ *more*
proof –
have *1*: $\vdash g \supset_i$ *more* **using** *assms* **by** *auto*
hence *2*: $\vdash f; g \supset_i f;$ *more* **by** (*rule RightChopImpChop*)
have *3*: $\vdash f;$ *more* $\supset_i$ *more* **by** (*rule ChopMoreImpMore*)
from *2 3* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *NotDiEqvBiNot*:
$\vdash$ $\neg_i$ (*di* f) $\equiv_i$ *bi* ($\neg_i$ f)
proof –
have *1*: $\vdash f \equiv_i \neg_i \neg_i f$ **by** *auto*
hence *2*: $\vdash$ *di* $f \equiv_i$ *di* $\neg_i \neg_i f$ **by** (*rule DiEqvDi*)
hence *3*: $\vdash \neg_i$ (*di* f) $\equiv_i \neg_i$ (*di* $\neg_i \neg_i f$) **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *bi-d-def*)
qed

lemma *ChopImpDi*:
$\vdash$ $f; g \supset_i$ *di* f
proof –
have *1*: $\vdash g \supset_i true_i$ **by** *auto*
hence *2*: $\vdash f; g \supset_i f; true_i$ **by** (*rule RightChopImpChop*)
from *2* **show** *?thesis* **by** (*simp add*: *bi-d-def*)
qed

lemma *TrueEqvTrueChopTrue*:
$\vdash$ $true_i \equiv_i true_i; true_i$
proof –
have *1*: $\vdash true_i; true_i \supset_i true_i$ **by** *auto*
have *2*: $\vdash true_i \supset_i$ *di* $true_i$ **by** (*rule DiIntro*)
hence *3*: $\vdash true_i \supset_i true_i; true_i$ **by** (*simp add*: *di-d-def*)
from *1 3* **show** *?thesis* **by** *auto*
qed

lemma *DiEqvDiDi*:
$\vdash$ *di* $f \equiv_i$ *di* (*di* f)
proof –
have *1*: $\vdash true_i \equiv_i true_i; true_i$ **by** (*rule TrueEqvTrueChopTrue*)
hence *2*: $\vdash f; true_i \equiv_i f; (true_i; true_i)$ **by** (*rule RightChopEqvChop*)
have *3*: $\vdash f; (true_i; true_i) \equiv_i (f; true_i); true_i$ **by** (*rule ChopAssoc*)
have *4*: $\vdash f; true_i \equiv_i (f; true_i); true_i$ **using** *2 3* **using** *prop03* **by** *blast*
from *4* **show** *?thesis* **by** (*metis di-d-def*)
qed

lemma *BiEqvBiBi*:
$\vdash$ *bi* $f \equiv_i$ *bi*(*bi* f)
proof $-$
have *1*: $\vdash$ *di* $\neg_i$ $f \equiv_i$ *di*(*di* $\neg_i$ f) **by** (*rule DiEqvDiDi*)
have *2*: $\vdash$ *di* $\neg_i$ $f \equiv_i \neg_i$ (*bi* f) **by** (*rule DiNotEqvNotBi*)
hence *3*: $\vdash$ *di* (*di* $\neg_i$ f) $\equiv_i$ *di* $\neg_i$ (*bi* f) **by** (*rule DiEqvDi*)
have *4*: $\vdash$ *di* $\neg_i$ $f \equiv_i$ *di* $\neg_i$(*bi* f) **using** *1 3* **using** *prop03* **by** *blast*
hence *5*: $\vdash \neg_i$ (*di* $\neg_i$ f) $\equiv_i \neg_i$ (*di* $\neg_i$ (*bi* f)) **using** *itl-prop*(*33*) **by** *blast*
from *5* **show** *?thesis* **by** (*metis bi-d-def*)
qed

lemma *DiOrEqv*:
$\vdash$ *di* ($f \vee_i$ g) $\equiv_i$ *di* $f \vee_i$ *di* g
proof $-$
have *1*: $\vdash$ ($f \vee_i$ g); *true*$_i$ $\equiv_i$ f; *true*$_i$ $\vee_i$ g; *true*$_i$ **by** (*rule OrChopEqv*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiAndA*:
$\vdash$ *di* ($f \wedge_i g$) $\supset_i$ *di* f
proof $-$
have *1*: $\vdash$ ($f \wedge_i g$); *true*$_i$ $\supset_i$ f; *true*$_i$ **by** (*rule AndChopA*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiAndB*:
$\vdash$ *di* ($f \wedge_i g$) $\supset_i$ *di* g
proof $-$
have *1*: $\vdash$ ($f \wedge_i g$); *true*$_i$ $\supset_i$ g; *true*$_i$ **by** (*rule AndChopB*)
from *1* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiAndImpAnd*:
$\vdash$ *di* ($f \wedge_i g$) $\supset_i$ *di* $f \wedge_i$ *di* g
proof $-$
have *1*: $\vdash$ *di* ($f \wedge_i g$) $\supset_i$ *di* f **by** (*rule DiAndA*)
have *2*: $\vdash$ *di* ($f \wedge_i g$) $\supset_i$ *di* g **by** (*rule DiAndB*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *DiSkipEqvMore*:
$\vdash$ *di skip* $\equiv_i$ *more*
proof $-$
have *1*: $\vdash$ *skip* ; *true*$_i$ $\equiv_i$ $\bigcirc$*true*$_i$ **by** (*rule SkipChopEqvNext*)
have *2*: $\vdash$ $\bigcirc$*true*$_i$ $\equiv_i$ *more* **by** *auto*
have *3*: $\vdash$ *skip* ; *true*$_i$ $\equiv_i$ *more* **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiMoreEqvMore*:

$\vdash$ *di more* $\equiv_i$ *more*
proof −
have *1*: $\vdash$ *di* ($\bigcirc$ *true*$_i$) $\equiv_i$ $\bigcirc$(*di true*$_i$) **by** (*rule DiNext*)
have *2*: $\vdash$ $\bigcirc$(*di true*$_i$) $\supset_i$ *more* **by** *auto*
have *3*: $\vdash$ *di*($\bigcirc$ *true*$_i$) $\supset_i$ *more* **using** *1 2* **by** *auto*
hence *4*: $\vdash$ *di more* $\supset_i$ *more* **by** (*simp add*: *more-d-def*)
have *5*: $\vdash$ *more* $\supset_i$ *di more* **by** (*rule ImpDi*)
from *4 5* **show** *?thesis* **by** *auto*
qed

lemma *DiIfEqvRule*:
assumes $\vdash$ *f* $\equiv_i$ *if*$_i$ (*init w*) *then g else h*
shows $\vdash$ *di f* $\equiv_i$ *if*$_i$ (*init w*) *then* (*di g*) *else* (*di h*)
proof −
have *1*: $\vdash$ *f* $\equiv_i$ *if*$_i$ (*init w*) *then g else h* **using** *assms* **by** *auto*
hence *2*: $\vdash$ *f*; *true*$_i$ $\equiv_i$ *if*$_i$ (*init w*) *then* (*g*; *true*$_i$) *else* (*h*; *true*$_i$) **by** (*rule IfChopEqvRule*)
from *2* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DiEmpty*:
$\vdash$ *di empty*
proof −
have *1*: $\vdash$ *true*$_i$ **by** *auto*
have *2*: $\vdash$ *empty* ; *true*$_i$ $\equiv_i$ *true*$_i$ **by** (*rule EmptyChop*)
have *3*: $\vdash$ *empty* ; *true*$_i$ **using** *1 2* **by** *auto*
from *3* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DaNotEqvNotBa*:
$\vdash$ *da* $\neg_i$ *f* $\equiv_i$ $\neg_i$ (*ba f*)
proof −
have *1*: $\vdash$ *ba f* $\equiv_i$ $\neg_i$ (*da* $\neg_i$ *f*) **by** (*simp add*: *ba-d-def*)
from *1* **show** *?thesis* **by** *simp*
qed

lemma *DaEqvDa*:
assumes $\vdash$ *f* $\equiv_i$ *g*
shows $\vdash$ *da f* $\equiv_i$ *da g*
using *assms* **by** *auto*

lemma *DaEqvNotBaNot*:
$\vdash$ *da f* $\equiv_i$ $\neg_i$ (*ba* $\neg_i$ *f*)
proof −
have *1*: $\vdash$ *ba* $\neg_i$ *f* $\equiv_i$ $\neg_i$ (*da* $\neg_i$ $\neg_i$ *f*) **by** (*simp add*: *ba-d-def*)
hence *2*: $\vdash$ *da* $\neg_i$ $\neg_i$ *f* $\equiv_i$ $\neg_i$(*ba* $\neg_i$ *f*) **by** *simp*
have *3*: $\vdash$ *f* $\equiv_i$ $\neg_i$ $\neg_i$ *f* **by** *simp*
hence *4*: $\vdash$ *da f* $\equiv_i$ *da* $\neg_i$ $\neg_i$ *f* **by** (*rule DaEqvDa*)
from *2 4* **show** *?thesis* **by** *simp*
qed

lemma *BaElim*:
$\vdash\ \ ba\ \ f \supset_i f$
proof −
have *1*: $\vdash\ ba\ \ f \equiv_i \Box(bi\ \ f)$ **by** (*rule BaEqvBtBi*)
have *2*: $\vdash bi\ \ f \supset_i f$ **by** (*rule BiElim*)
hence *3*: $\vdash \Box(bi\ \ f \supset_i f)$ **by** (*rule BoxGen*)
have *4*: $\vdash \Box(bi\ \ f \supset_i f) \supset_i \Box(bi\ \ f) \supset_i \Box\ f$ **by** (*rule BoxImpDist*)
have *5*: $\vdash \Box(bi\ \ f) \supset_i \Box\ f$ **using** *3 4 MP* **by** *simp*
have *6*: $\vdash \Box\ f \supset_i f$ **by** (*rule BoxElim*)
from *1 5 6* **show** *?thesis* **using** *BaImpBt prop02* **by** *blast*
qed

lemma *DaIntro*:
$\vdash\ \ f \supset_i\ da\ \ f$
proof −
have *1*: $\vdash\ ba\ \ \neg_i\ \ f \supset_i \neg_i\ \ f$ **by** (*rule BaElim*)
hence *2*: $\vdash \neg_i \neg_i\ \ f \supset_i \neg_i\ (\ \ ba\ \ \neg_i\ \ f)$ **using** *prop27* **by** *blast*
have *3*: $\vdash f \equiv_i \neg_i \neg_i\ \ f$ **by** *simp*
have *4*: $\vdash\ da\ \ f \equiv_i \neg_i\ \ (\ ba\ \ \neg_i\ \ f)$ **by** (*rule DaEqvNotBaNot*)
from *2 3 4* **show** *?thesis* **by** *simp*
qed

lemma *BaGen*:
assumes $\vdash\ \ f$
shows $\vdash\ ba\ \ f$
proof −
have *1*: $\vdash\ \ f$ **using** *assms* **by** *auto*
hence *2*: $\vdash \Box\ f$ **by** (*rule BoxGen*)
hence *3*: $\vdash bi(\ \Box\ f)$ **by** (*rule BiGen*)
have *4*: $\vdash\ ba\ \ f \equiv_i bi\ (\Box\ f)$ **by** (*rule BaEqvBiBt*)
from *3 4* **show** *?thesis* **by** *simp*
qed

lemma *BaImpDist*:
$\vdash\ ba\ (f \supset_i g) \supset_i\ \ ba\ \ f \supset_i\ \ ba\ \ g$
proof −
have *1*: $\vdash bi\ (f \supset_i g) \supset_i (bi\ \ f \supset_i bi\ \ g)$ **by** (*rule BiImpDist*)
hence *2*: $\vdash \Box(bi\ (f \supset_i g) \supset_i (bi\ \ f \supset_i bi\ \ g))$ **by** (*rule BoxGen*)
have *3*: $\vdash \Box(bi\ (f \supset_i g) \supset_i (bi\ \ f \supset_i bi\ \ g))$
$\supset_i$
$(\Box\ (bi\ (f \supset_i g)) \supset_i (\Box(bi\ \ f) \supset_i \Box(bi\ \ g)))$ **by** *simp*
have *4*: $\vdash \Box(bi\ (f \supset_i g)) \supset_i (\Box(bi\ \ f) \supset_i \Box(bi\ \ g))$ **using** *2 3 MP* **by** *simp*
have *5*: $\vdash\ ba\ (f \supset_i g) \equiv_i \Box(bi\ (f \supset_i g))$ **by** (*rule BaEqvBtBi*)
have *6*: $\vdash\ ba\ \ f \equiv_i \Box(bi\ \ f)$ **by** (*rule BaEqvBtBi*)
have *7*: $\vdash\ ba\ \ g \equiv_i \Box(bi\ \ g)$ **by** (*rule BaEqvBtBi*)
from *4 5 6 7* **show** *?thesis* **by** *simp*
qed

lemma *BaAndEqv*:

$\vdash \quad ba\ (f \wedge_i g) \equiv_i \ ba \ \ f \wedge_i \ ba \ \ g$
proof $-$
have *1*: $\vdash \quad ba\ (f \wedge_i g) \equiv_i \quad \Box(bi\ (f \wedge_i g))$ **by** (*rule BaEqvBtBi*)
have *2*: $\vdash \ bi\ (f \wedge_i g) \equiv_i bi\ f \wedge_i bi\ g$ **by** *auto*
hence *3*: $\vdash \Box(bi\ (f \wedge_i g)) \equiv_i \Box(bi\ f \wedge_i bi\ g)$ **by** *auto*
have *4*: $\vdash \Box(bi\ f \wedge_i bi\ g) \equiv_i \Box(bi\ f) \ \wedge_i \Box(bi\ g)$ **by** *auto*
have *5*: $\vdash ba \ \ f \equiv_i \Box(bi \ \ f)$ **by** (*rule BaEqvBtBi*)
have *6*: $\vdash \ ba \ \ g \equiv_i \Box(bi \ \ g)$ **by** (*rule BaEqvBtBi*)
from *1 3 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *BaImpBaEqvBa*:
$\vdash \quad ba\ (f \equiv_i g) \supset_i \ (ba \ \ f \equiv_i \ ba \ \ g)$
proof $-$
have *1*: $\vdash \ ba\ (f \supset_i g) \supset_i \ ba \ \ f \supset_i \ ba \ \ g$ **by** (*rule BaImpDist*)
have *2*: $\vdash \ ba\ (g \supset_i f) \supset_i \ ba \ \ g \supset_i \ ba \ \ f$ **by** (*rule BaImpDist*)
have *3*: $\vdash ba\ (f \equiv_i g) \equiv_i ba\ ((f \supset_i g) \wedge_i (g \supset_i f))$ **by** *auto*
have *4*: $\vdash ba\ ((f \supset_i g) \wedge_i (g \supset_i f)) \equiv_i ba((f \supset_i g)) \wedge_i ba((g \supset_i f))$ **by** (*rule BaAndEqv*)
have *5*: $\vdash (ba \ \ f \supset_i \ ba \ \ g) \wedge_i (ba \ \ g \supset_i \ ba \ \ f) \equiv_i (ba \ \ f \equiv_i \ ba \ \ g)$ **by** *auto*
from *1 2 3 4 5* **show** *?thesis* **using** *itl-prop*(*31*) *itl-prop*(*32*) *prop02* **by** *smt*
qed

lemma *BaImpBa*:
assumes $\vdash \quad f \supset_i g$
shows $\vdash ba \ \ f \supset_i \ ba \ \ g$
using *BaGen BaImpDist MP assms* **by** *blast*

lemma *BaEqvBa*:
assumes $\vdash \quad f \equiv_i g$
shows $\vdash \ ba \ \ f \equiv_i \ ba \ \ g$
using *BaGen BaImpBaEqvBa MP assms* **by** *blast*

lemma *DaImpDa*:
assumes $\vdash \quad f \supset_i g$
shows $\vdash \ da \ \ f \supset_i \ da \ \ g$
using *assms* **by** *fastforce*

lemma *DiamondEqvDiamondDiamond*:
$\vdash \Diamond\ f \equiv_i \Diamond\ (\Diamond\ f)$
proof $-$
have *1*: $\vdash \Diamond\ (\Diamond\ f) \equiv_i true_i;(true_i;f)$
by *simp*
have *2*: $\vdash true_i;(true_i;f) \ \equiv_i (true_i;true_i);f$
by (*rule ChopAssoc*)

have *3*: $\vdash$ $(true_i;true_i);f \equiv_i true_i;f$
using *LeftChopEqvChop TrueEqvTrueChopTrue itl-prop*(*30*) **by** *blast*
have *4*: $\vdash$ $true_i;f \equiv_i \diamond f$
by (*simp add*: *sometimes-d-def*)
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *DaEqvDaDa*:
$\vdash$ $da\ f \equiv_i da\ (da\ f)$
proof −
have *1*: $\vdash$ $da\ f \equiv_i \diamond(di\ f)$
by (*rule DaEqvDtDi*)
have *2*: $\vdash$ $di\ f \equiv_i (di\ (di\ f))$
by (*rule DiEqvDiDi*)
hence *3*: $\vdash \diamond (di\ f) \equiv_i \diamond (di\ (di\ f))$
by (*rule DiamondEqvDiamond*)
have *4*: $\vdash \diamond\ (di\ f) \equiv_i \diamond(\diamond\ (di\ (di\ f)))$
using *DiamondEqvDiamondDiamond DiEqvDiDi* **using** *3 prop03* **by** *blast*
have *5*: $\vdash \diamond\ (di\ (di\ f)) \equiv_i di\ (\diamond\ (di\ f))$
by (*rule DtDiEqvDiDt*)
hence *6*: $\vdash \diamond(\diamond\ (di\ (di\ f))) \equiv_i \diamond\ (di\ (\diamond\ (di\ f)))$
by (*rule DiamondEqvDiamond*)
have *7*: $\vdash$ $da\ f \equiv_i \diamond\ (di(\diamond\ (di\ f)))$
using *1 3 4 6* **using** *prop03* **by** *blast*
have *8*: $\vdash$ $da\ (\diamond\ (di\ f)) \equiv_i \diamond(di\ (\diamond\ (di\ f)))$
by (*rule DaEqvDtDi*)
have *9*: $\vdash$ $da\ (da\ f) \equiv_i da\ (\diamond\ (di\ f))$
using *1* **by** (*rule DaEqvDa*)
from *7 8 9* **show** *?thesis* **by** *auto*
qed

lemma *BaEqvBaBa*:
$\vdash$ $ba\ f \equiv_i ba\ (ba\ f)$
proof −
have *1*: $\vdash$ $da\ (\neg_i\ f) \equiv_i da\ (da\ (\neg_i\ f))$ **by** (*rule DaEqvDaDa*)
have *2*: $\vdash$ $da\ (da\ (\neg_i\ f)) \equiv_i \neg_i\ (ba\ (\neg_i\ (da\ (\neg_i\ f))))$ **by** (*rule DaEqvNotBaNot*)
have *3*: $\vdash \neg_i\ (da\ (da\ (\neg_i\ f))) \equiv_i ba\ (\neg_i\ (da\ (\neg_i\ f)))$ **by** *auto*
have *4*: $\vdash \neg_i\ (da\ (\neg_i\ f)) \equiv_i ba\ (\neg_i\ (da\ (\neg_i\ f)))$ **using** *1 2 3 prop01 prop03* **by** *blast*
from *4* **show** *?thesis* **by** (*metis ba-d-def*)
qed

lemma *BaLeftChopImpChop*:
$\vdash$ $ba\ (f \supset_i f1) \supset_i f;\ g \supset_i f1;\ g$
proof −
have *1*: $\vdash$ $ba\ (f \supset_i f1) \supset_i bi\ (f \supset_i f1)$ **by** (*rule BaImpBi*)
have *2*: $\vdash$ $bi\ (f \supset_i f1) \supset_i f;\ g \supset_i f1;\ g$ **by** (*rule BiChopImpChop*)
from *1 2* **show** *?thesis* **by** *auto*

qed

lemma *BaRightChopImpChop*:
$\vdash$ *ba* $(g \supset_i g1) \supset_i f; g \supset_i f; g1$
proof $-$
have *1*: $\vdash$ *ba* $(g \supset_i g1) \supset_i \Box(g \supset_i g1)$ **by** (*rule BaImpBt*)
have *2*: $\vdash \Box(g \supset_i g1) \supset_i f; g \supset_i f; g1$ **by** (*rule BoxChopImpChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *ChopAndBaImport*:
$\vdash$ $(f; f1) \wedge_i$ *ba* $g \supset_i (f \wedge_i g); (f1 \wedge_i g)$
proof $-$
have *1*: $\vdash$ *ba* $g \wedge_i (f; f1) \supset_i (g \wedge_i f); (g \wedge_i f1)$ **by** (*rule BaAndChopImport*)
have *2*: $\vdash (g \wedge_i f); (g \wedge_i f1) \equiv_i (f \wedge_i g); (f1 \wedge_i g)$ **by** (*rule AndChopAndCommute*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *BaImpBaImpBaAnd*:
$\vdash$ *ba* $h \supset_i$ *ba*$(g \supset_i$ *ba* $h \wedge_i g)$
proof $-$
have *1*: $\vdash$ *ba* $h \supset_i (g \supset_i$ *ba* $h \wedge_i g)$ **by** *simp*
hence *2*: $\vdash$ *ba*(*ba* h) $\supset_i$ *ba*$(g \supset_i$ *ba* $h \wedge_i g)$ **by** (*rule BaImpBa*)
have *3*: $\vdash$ *ba* $h \equiv_i$ *ba*(*ba* h) **by** (*rule BaEqvBaBa*)
from *2 3* **show** *?thesis* **using** *itl-prop*(*31*) *prop02* **by** *blast*
qed

lemma *BaChopImpChopBa*:
$\vdash$ *ba* $f \supset_i g; g1 \supset_i g;$ ((*ba* f) $\wedge_i g1$)
proof $-$
have *1*: $\vdash$ *ba* $f \supset_i$ *ba* $(g1 \supset_i$ (*ba* f) $\wedge_i g1)$ **by** (*rule BaImpBaImpBaAnd*)
have *2*: $\vdash$ *ba* $(g1 \supset_i$ *ba* $f \wedge_i g1) \supset_i g; g1 \supset_i g;$ (*ba* $f \wedge_i g1$) **by** (*rule BaRightChopImpChop*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *DiNotBaImpNotBa*:
$\vdash$ *di* $\neg_i$ (*ba* f) $\supset_i \neg_i$ (*ba* f)
proof $-$
have *1*: $\vdash$ *ba* $f \equiv_i$ *ba*(*ba* f) **by** (*rule BaEqvBaBa*)
have *2*: $\vdash$ *ba* (*ba* f) $\supset_i$ *bi* (*ba* f) **by** (*rule BaImpBi*)
have *3*: $\vdash$ *ba* $f \supset_i$ *bi* (*ba* f) **using** *1 2* **using** *itl-prop*(*31*) *prop02* **by** *blast*
hence *4*: $\vdash$ *ba* $f \supset_i \neg_i$ (*di* $\neg_i$ (*ba* f)) **by** (*simp add*: *bi-d-def*)
from *4* **show** *?thesis* **by** *fastforce*
qed

lemma *NotBaChopImpNotBa*:
$\vdash$ $(\neg_i$ (*ba* f)); $g \supset_i \neg_i$ (*ba* f)
proof $-$
have *1*: $\vdash (\neg_i$ (*ba* f)); $g \supset_i$ *di* $\neg_i$ (*ba* f) **by** (*rule ChopImpDi*)

have *2*: $\vdash$ *di* $\neg_i$ (*ba* *f*) $\supset_i$ $\neg_i$ (*ba* *f*) **by** (*rule DiNotBaImpNotBa*)
from *1 2* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *DiamondFinImpFin*:
$\vdash \Diamond$ (*fin f*) $\supset_i$ *fin f*
proof –
have *1*: $\vdash$ *fin f* $\equiv_i$ *true*$_i$;(*f* $\wedge_i$ *empty*)
by (*rule FinEqvTrueChopAndEmpty*)
hence *2*: $\vdash \Diamond$ (*fin f*) $\equiv_i$ *true*$_i$;(*true*$_i$;(*f* $\wedge_i$ *empty*))
by *fastforce*
have *3*: $\vdash$ *true*$_i$;(*true*$_i$;(*f* $\wedge_i$ *empty*)) $\equiv_i$ (*true*$_i$;*true*$_i$);(*f* $\wedge_i$ *empty*)
by (*rule ChopAssoc*)
have *4*: $\vdash$ (*true*$_i$;*true*$_i$);(*f* $\wedge_i$ *empty*) $\equiv_i$ *true*$_i$;(*f* $\wedge_i$ *empty*)
using *TrueEqvTrueChopTrue* **using** *LeftChopEqvChop itl-prop*(*30*) **by** *blast*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *ChopFinImpFin*:
$\vdash$ *f*; *fin* (*init w*) $\supset_i$ *fin* (*init w*)
proof –
have *1*: $\vdash$ *f*; *fin* (*init w*) $\supset_i$ $\Diamond$ (*fin* (*init w*)) **by** (*rule ChopImpDiamond*)
have *2*: $\vdash \Diamond$ (*fin* (*init w*)) $\supset_i$ *fin* (*init w*) **by** (*rule DiamondFinImpFin*)
from *1 2* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *FinImpYieldsFin*:
$\vdash$ *fin* (*init w*) $\supset_i$ *f yields* (*fin* (*init w*))
proof –
have *1*: $\vdash$ *f*; *fin* (*init* $\neg_i$ *w*) $\supset_i$ *fin* (*init* $\neg_i$ *w*)
by (*rule ChopFinImpFin*)
have *2*: $\vdash$ *fin* (*init* $\neg_i$ *w*) $\equiv_i$ $\neg_i$ (*fin* (*init w*))
using *FinNotStateEqvNotFinState* **by** *blast*
hence *3*: $\vdash$ *f*; *fin* (*init* $\neg_i$ *w*) $\equiv_i$ *f*; $\neg_i$ (*fin* (*init w*))
by (*rule RightChopEqvChop*)
have *4*: $\vdash$ *f*; $\neg_i$ (*fin* (*init w*)) $\supset_i$ $\neg_i$ (*fin* (*init w*))
using *1 2 3 itl-prop*(*31*) *prop02* **by** *blast*
hence *5*: $\vdash$ *fin* (*init w*) $\supset_i$ $\neg_i$ (*f*; $\neg_i$ (*fin* (*init w*)))
using *itl-prop*(*35*) **by** *auto*
from *5* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *ChopAndFin*:
$\vdash$ (*f*; *g*) $\wedge_i$ *fin* (*init w*) $\equiv_i$ *f*; (*g* $\wedge_i$ *fin* (*init w*))
proof –
have *1*: $\vdash$ *fin* (*init w*) $\supset_i$ *f yields* (*fin* (*init w*))
by (*rule FinImpYieldsFin*)
hence *2*: $\vdash$ (*f*; *g*) $\wedge_i$ *fin* (*init w*) $\supset_i$ (*f*; *g*) $\wedge_i$ *f yields* (*fin* (*init w*))

by *auto*
have *3*: $\vdash (f;\, g) \wedge_i$ *f yields* (*fin* (*init w*)) $\supset_i f;\, (g \wedge_i$ *fin* (*init w*))
by (*rule ChopAndYieldsImp*)
have *4*: $\vdash (f;\, g) \wedge_i$ *fin* (*init w*) $\supset_i f;\, (g \wedge_i$ *fin* (*init w*))
using *2 3* **by** *auto*
have *11*: $\vdash f;\, (g \wedge_i$ *fin* (*init w*)) $\supset_i f;\, g$
by (*rule ChopAndA*)
have *12*: $\vdash f;\, (g \wedge_i$ *fin* (*init w*)) $\supset_i f;$ *fin* (*init w*)
by (*rule ChopAndB*)
have *13*: $\vdash f;$ *fin* (*init w*) $\supset_i \Diamond$ (*fin* (*init w*))
by (*rule ChopImpDiamond*)
have *14*: $\vdash \Diamond$(*fin* (*init w*)) $\supset_i$ *fin* (*init w*)
by (*rule DiamondFinImpFin*)
have *15*: $\vdash f;\, (g \wedge_i$ *fin* (*init w*)) $\supset_i (f;\, g) \wedge_i$ *fin* (*init w*)
using *11 12 13 14 itl-prop*(*32*) *prop02* **by** *smt*
from *4 15* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *ChopAndNotFin*:
$\vdash$ $f;\, g \wedge_i \neg_i$ (*fin* (*init w*)) $\equiv_i f;\, (g \wedge_i \neg_i$ (*fin* (*init w*)))
proof −
have *1*: $\vdash f;\, g \wedge_i$ *fin* (*init* $\neg_i$ *w*) $\equiv_i f;\, (g \wedge_i$ *fin* (*init* $\neg_i$ *w*))
by (*rule ChopAndFin*)
have *2*: $\vdash$ *fin* (*init* $\neg_i$ *w*) $\equiv_i \neg_i$ (*fin* (*init w*))
using *FinNotStateEqvNotFinState* **by** *blast*
hence *3*: $\vdash g \wedge_i$ *fin* (*init* $\neg_i$ *w*) $\equiv_i g \wedge_i \neg_i$(*fin* (*init w*))
by *auto*
hence *4*: $\vdash f;\, (g \wedge_i$ *fin* (*init* $\neg_i$ *w*)) $\equiv_i f;\, (g \wedge_i \neg_i$ (*fin* (*init w*)))
by (*rule RightChopEqvChop*)
from *1 2 4* **show** *?thesis* **by** *auto*
qed

lemma *FinChopChain*:
$\vdash$ ((*init w*)$\supset_i$ *fin* (*init w1*)); ((*init w1*) $\supset_i$ *fin* (*init w2*))
$\supset_i$ ((*init w*) $\supset_i$ *fin* (*init w2*))
proof −
have *1*: $\vdash$ (*init w*) $\wedge_i$ ((*init w*)$\supset_i$ *fin* (*init w1*)); ((*init w1*) $\supset_i$ *fin* (*init w2*))
$\supset_i$
((*init w*) $\wedge_i$ ((*init w*)$\supset_i$ *fin* (*init w1*))); ((*init w1*)$\supset_i$ *fin* (*init w2*))
by (*rule StateAndChopImport*)
have *2*: $\vdash$ (*init w*) $\wedge_i$ ((*init w*) $\supset_i$ *fin* (*init w1*)) $\supset_i$ *fin* (*init w1*)
by *auto*
have *3*: $\vdash$ ((*init w*) $\wedge_i$ ((*init w*)$\supset_i$ *fin* (*init w1*))); ((*init w1*) $\supset_i$ *fin* (*init w2*))
$\supset_i$
(*fin* (*init w1*)); ((*init w1*) $\supset_i$ *fin* (*init w2*))
using *2* **by** (*rule LeftChopImpChop*)
have *4*: $\vdash$ (*fin* (*init w1*)); ((*init w1*) $\supset_i$ *fin* (*init w2*)) $\equiv_i$
$\Diamond$((*init w1*) $\wedge_i$ ((*init w1*) $\supset_i$ *fin* (*init w2*)))
by (*rule FinChopEqvDiamond*)
have *41*: $\vdash$ ((*init w1*) $\wedge_i$ ((*init w1*) $\supset_i$ *fin* (*init w2*))) $\supset_i$ *fin* (*init w2*)

by *auto*
have *42*: $\vdash \Diamond((init\ w1) \wedge_i ((init\ w1) \supset_i\ fin\ (init\ w2))) \supset_i \Diamond\ (\ fin\ (init\ w2))$
using *41 DiamondImpDiamond* **by** *blast*
have *5*: $\vdash \Diamond(\ fin(\ init\ w2)) \supset_i\ fin\ (init\ w2)$
using *DiamondFinImpFin* **by** *blast*
have *6*: $\vdash (init\ w) \wedge_i\ ((init\ w)\supset_i\ fin\ (init\ w1)); ((init\ w1) \supset_i\ fin\ (init\ w2))$
$\supset_i\ fin\ (init\ w2)$
using *1 3 4 5 42 itl-prop*(*30*) *prop02 prop15* **by** *smt*
from *6* **show** *?thesis* **using** *prop32* **by** *blast*
qed

lemma *ChopRule*:
assumes $\vdash\ (init\ w) \wedge_i f \supset_i\ fin\ (init\ w1)$
$\vdash\ (init\ w1)\wedge_i f1 \supset_i\ fin\ (init\ w2)$
shows $\vdash\ (init\ w) \wedge_i (f; f1) \supset_i\ fin\ (init\ w2)$
proof –
have *1*: $\vdash (init\ w) \wedge_i (f; f1) \supset_i ((init\ w) \wedge_i f); f1$ **by** (*rule StateAndChopImport*)
have *2*: $\vdash (init\ w) \wedge_i f \supset_i\ fin\ (init\ w1)$ **using** *assms* **by** *auto*
hence *3*: $\vdash ((init\ w) \wedge_i f); f1 \supset_i (\ fin\ (init\ w1)); f1$ **by** (*rule LeftChopImpChop*)
have *4*: $\vdash (\ fin\ (init\ w1)); f1 \equiv_i\ \Diamond((init\ w1) \wedge_i f1)$ **by** (*rule FinChopEqvDiamond*)
have *5*: $\vdash (init\ w1) \wedge_i f1 \supset_i\ fin\ (init\ w2)$ **using** *assms* **by** *auto*
hence *6*: $\vdash \Diamond((init\ w1) \wedge_i f1) \supset_i \Diamond\ (fin\ (init\ w2))$ **by** (*rule DiamondImpDiamond*)
have *7*: $\vdash \Diamond(\ fin\ (init\ w2)) \supset_i\ fin\ (init\ w2)$ **using** *DiamondFinImpFin* **by** *blast*
from *1 3 4 6 7* **show** *?thesis* **using** *itl-prop*(*30*) *prop02 prop15* **by** *smt*
qed

lemma *ChopRep*:
assumes $\vdash\ (init\ w) \wedge_i f \supset_i f1 \wedge_i\ fin\ (init\ w1)$
$\vdash\ (init\ w1) \wedge_i g \supset_i g1$
shows $\vdash\ (init\ w) \wedge_i (f; g) \supset_i (f1; g1)$
proof –
have *1*: $\vdash (init\ w) \wedge_i f \supset_i f1 \wedge_i\ fin\ (init\ w1)$ **using** *assms* **by** *auto*
hence *2*: $\vdash (init\ w) \wedge_i (f; g) \supset_i (f1 \wedge_i\ fin\ (init\ w1)); g$ **by** (*rule StateAndChopImpChopRule*)
have *3*: $\vdash (f1 \wedge_i\ fin\ (init\ w1)); g \equiv_i f1; ((init\ w1) \wedge_i g)$ **by** (*rule AndFinChopEqvStateAndChop*)
have *4*: $\vdash (init\ w1)\wedge_i g \supset_i g1$ **using** *assms* **by** *auto*
hence *5*: $\vdash f1; ((init\ w1) \wedge_i g) \supset_i f1; g1$ **by** (*rule RightChopImpChop*)
from *2 3 5* **show** *?thesis* **by** *simp*
qed

lemma *ChopRepAndFin*:
assumes $\vdash\ (init\ w) \wedge_i f \supset_i f1 \wedge_i\ fin\ (init\ w1)$
$\vdash\ (init\ w1) \wedge_i g \supset_i g1 \wedge_i\ fin\ (init\ w2)$
shows $\vdash\ (init\ w) \wedge_i (f; g) \supset_i (f1; g1) \wedge_i\ fin\ (init\ w2)$
proof –
have *1*: $\vdash (init\ w) \wedge_i f \supset_i f1 \wedge_i\ fin\ (init\ w1)$ **using** *assms* **by** *auto*
have *2*: $\vdash (init\ w1) \wedge_i g \supset_i g1 \wedge_i\ fin\ (init\ w2)$ **using** *assms* **by** *auto*
have *3*: $\vdash (init\ w) \wedge_i (f; g) \supset_i f1; (g1 \wedge_i\ fin\ (init\ w2))$ **using** *1 2* **by** (*rule ChopRep*)
have *4*: $\vdash f1; (g1 \wedge_i\ fin\ (init\ w2)) \supset_i f1; g1$ **by** (*rule ChopAndA*)
have *5*: $\vdash f1; (g1 \wedge_i\ fin\ (init\ w2)) \supset_i f1;\ fin\ (init\ w2)$ **by** (*rule ChopAndB*)

have *6*: $\vdash$ *f1*; *fin* (*init w2*) $\supset_i$ *fin* (*init w2*) **by** (*rule ChopFinImpFin*)
from *1 2 3 4 5 6* **show** *?thesis* **using** *ChopRep ChopRule itl-prop*(*32*) **by** *blast*
qed

lemma *TrueChopMoreEqvMore*:
$\vdash$ *true*$_i$; *more* $\equiv_i$ *more*
by *auto*

lemma *MoreChopLoop*:
assumes $\vdash$ *f* $\supset_i$ *more* ; *f*
shows $\vdash \neg_i$ *f*
proof –
have *1*: $\vdash$ *f* $\supset_i$ *more* ; *f*
using *assms* **by** *auto*
hence *11*: $\vdash \diamond$ *f* $\supset_i \diamond$ (*more*;*f*)
by (*rule DiamondImpDiamond*)
have *12*: $\vdash \diamond$ (*more*;*f*) $\equiv_i$ *true*$_i$;(*more*;*f*)
by *simp*
have *13*: $\vdash$ *true*$_i$;(*more*;*f*) $\equiv_i$ (*true*$_i$;*more*);*f*
by (*rule ChopAssoc*)
have *14*: $\vdash \diamond$ (*more*;*f*) $\equiv_i$ *more*;*f*
using *TrueChopMoreEqvMore 12 13 LeftChopChopImpChopRule NowImpDiamond*
itl-prop(*31*) *prop02* **by** *metis*
have *2*: $\vdash$ *more* ; *f* $\equiv_i \bigcirc(\diamond$ *f*)
by (*rule MoreChopEqvNextDiamond*)
have *3*: $\vdash \diamond$ *f* $\supset_i \bigcirc(\diamond$ *f*)
using *11 14 2* **by** *auto*
hence *4*: $\vdash \neg_i$ ($\diamond$ *f*)
by (*rule NextLoop*)
have *5*: $\vdash \neg_i$ ($\diamond$ *f*) $\supset_i \neg_i$ *f*
by *auto*
from *4 5* **show** *?thesis* **using** *MP* **by** *blast*
qed

lemma *MoreChopContra*:
assumes $\vdash$ *f* $\wedge_i \neg_i$ *g* $\supset_i$ (*more* ; (*f* $\wedge_i \neg_i$ *g*))
shows $\vdash$ *f* $\supset_i$ *g*
proof –
have *1*: $\vdash$ *f* $\wedge_i \neg_i$ *g* $\supset_i$ (*more* ; (*f* $\wedge_i \neg_i$ *g*)) **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i$ (*f* $\wedge_i \neg_i$ *g*) **by** (*rule MoreChopLoop*)
from *2* **show** *?thesis* **by** *auto*
qed

lemma *ChopLoop*:
assumes $\vdash$ *f* $\supset_i$ *g*;*f*
$\vdash$ *g* $\supset_i$ *more*
shows $\vdash \neg_i$ *f*
proof –

have *1*: $\vdash f \supset_i g; f$ **using** *assms* **by** *auto*
have *2*: $\vdash g \supset_i$ *more* **using** *assms* **by** *auto*
hence *3*: $\vdash g; f \supset_i$ *more* ; f **by** (*rule LeftChopImpChop*)
have *4*: $\vdash f \supset_i$ *more* ; f **using** *1 3* **by** *auto*
from *4* **show** *?thesis* **using** *MoreChopLoop* **by** *auto*
qed

lemma *ChopContra*:
assumes $\vdash f \wedge_i \neg_i g \supset_i h; f \wedge_i \neg_i (h; g)$
$\vdash h \supset_i$ *more*
shows $\vdash f \supset_i g$
proof −
have *1*: $\vdash f \wedge_i \neg_i g \supset_i h; f \wedge_i \neg_i (h; g)$ **using** *assms* **by** *auto*
have *2*: $\vdash h \supset_i$ *more* **using** *assms* **by** *auto*
have *3*: $\vdash h; f \wedge_i \neg_i (h; g) \supset_i h; (f \wedge_i \neg_i g)$ **by** (*rule ChopAndNotChopImp*)
have *4*: $\vdash h; (f \wedge_i \neg_i g) \supset_i$ *more* ; $(f \wedge_i \neg_i g)$ **using** *2* **by** (*rule LeftChopImpChop*)
have *5*: $\vdash f \wedge_i \neg_i g \supset_i$ *more* ; $(f \wedge_i \neg_i g)$ **using** *1 3 4* **by** *auto*
from *5* **show** *?thesis* **using** *MoreChopContra* **by** *auto*
qed

5.7 Properties of Chopstar and Chopplus

lemma *EmptyImpCS*:
$\vdash$ *empty* $\supset_i f^\star$
proof −
have *1*: $\vdash f^\star \equiv_i$ *empty* $\vee_i (f \wedge_i$ *more*$);f^\star$ **by** (*rule ChopstarEqv*)
have *2*: $\vdash$ *empty* $\supset_i$ *empty* $\vee_i (f \wedge_i$ *more*$);f^\star$ **by** *auto*
from *1 2* **show** *?thesis* **using** *itl-prop*(*31*) *prop02* **by** *blast*
qed

lemma *CSEqvOrChopCS*:
$\vdash f^\star \equiv_i$ *empty* $\vee_i (f; f^\star)$
proof −
have *1*: $\vdash f^\star \equiv_i$ *empty* $\vee_i (f \wedge_i$ *more*$);f^\star$ **by** (*rule ChopstarEqv*)
have *2*: $\vdash (f \wedge_i$ *more*$);f^\star \supset_i f;f^\star$ **by** (*rule AndChopA*)
have *3*: $\vdash f^\star \supset_i$ *empty* $\vee_i f; f^\star$ **using** *1 2* **using** *prop14* **by** *blast*
have *4*: $\vdash$ *empty* $\supset_i f^\star$ **by** (*rule EmptyImpCS*)
have *5*: $\vdash f \supset_i$ *empty* $\vee_i (f \wedge_i$ *more*$)$ **by** *auto*
have *6*: $\vdash f; f^\star \supset_i f^\star \vee_i (f \wedge_i$ *more* $); f^\star$ **using** *5* **by** (*rule EmptyOrChopImpRule*)
have *7*: $\vdash f^\star \supset_i$ *empty* $\vee_i (f \wedge_i$ *more*$);f^\star$ **using** *1* **using** *itl-prop*(*31*) **by** *blast*
have *8*: $\vdash f; f^\star \supset_i$ *empty* $\vee_i (f \wedge_i$ *more* $); f^\star$ **using** *6 7 prop16* **by** *blast*
hence *9*: $\vdash f; f^\star \supset_i f^\star$ **using** *1 prop15* **by** *blast*
have *10*: $\vdash$ *empty* $\vee_i f; f^\star \supset_i f^\star$ **using** *9 4 prop30* **by** *blast*
from *3 10* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *CSAndMoreEqvAndMoreChop*:
$\vdash f^\star \wedge_i$ *more* $\equiv_i (f \wedge_i$ *more* $); f^\star$
proof −
have *1*: $\vdash$ (*empty* $\vee_i (f \wedge_i$ *more* $); f^\star) \wedge_i$ *more* $\supset_i (f \wedge_i$ *more* $); f^\star$ **by** *auto*

have *2*: $\vdash f^\star \equiv_i$ *empty* $\vee_i$ $(f \wedge_i$ *more*$)$;$f^\star$ **by** (*rule ChopstarEqv*)
have *3*: $\vdash f^\star \wedge_i$ *more* $\supset_i (f \wedge_i$ *more* $)$; $f^\star$ **using** *1 2* **using** *prop18* **by** *blast*
have *4*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \supset_i f^\star$ **using** *2 prop19* **by** *blast*
have *5*: $\vdash (f \wedge_i$ *more* $) \supset_i$ *more* **by** *auto*
hence *6*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \supset_i$ *more* **by** (*rule LeftChopImpMoreRule*)
have *7*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \supset_i f^\star \wedge_i$ *more* **using** *4 6* **using** *itl-prop*(*32*) **by** *blast*
from *3 7* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *CSAndMoreImpChopCS*:
$\vdash$ $f^\star \wedge_i$ *more* $\supset_i f$; $f^\star$
proof –
have *1*: $\vdash f^\star \wedge_i$ *more* $\equiv_i (f \wedge_i$ *more* $)$; $f^\star$ **by** (*rule CSAndMoreEqvAndMoreChop*)
have *2*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \supset_i f$; $f^\star$ **by** (*rule AndChopA*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NotAndMoreEqvEmptyOr*:
$\vdash \neg_i$ $(f \wedge_i$ *more*$) \equiv_i$ (*empty* $\vee_i \neg_i f$)
by *auto*

lemma *MoreAndEmptyOrEqvMoreAnd*:
$\vdash$ *more* $\wedge_i$ (*empty* $\vee_i \neg_i f$) $\equiv_i$ *more* $\wedge_i \neg_i f$
by *auto*

lemma *CSMoreNotImpChopCSAndMore*:
$\vdash$ $f^\star \wedge_i$ *more* $\wedge_i \neg_i$ $f \supset_i (f \wedge_i$ *more* $)$; $(f^\star \wedge_i$ *more* $)$
proof –
have *1*: $\vdash f^\star \wedge_i$ *more* $\equiv_i (f \wedge_i$ *more* $)$; $f^\star$
 by (*rule CSAndMoreEqvAndMoreChop*)
have *2*: $\vdash$ *empty* $\vee_i$ *more*
 by *auto*
hence *3*: $\vdash f^\star \supset_i$ *empty* $\vee_i$ $(f^\star \wedge_i$ *more* $)$
 by *auto*
hence *4*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \supset_i (f \wedge_i$ *more* $) \vee_i$ $((f \wedge_i$ *more* $)$; $(f^\star \wedge_i$ *more* $))$
 by (*rule ChopEmptyOrImpRule*)
hence *5*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \wedge_i \neg_i(f \wedge_i$ *more*$) \supset_i$ $((f \wedge_i$ *more* $)$; $(f^\star \wedge_i$ *more* $))$
 using *prop29* **by** *blast*
have *6*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \equiv_i$ $(f \wedge_i$ *more* $)$; $f^\star \wedge_i$ *more* **using** *1*
 by *auto*
have *7*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \wedge_i \neg_i(f \wedge_i$ *more*$) \equiv_i (f \wedge_i$ *more* $)$; $f^\star \wedge_i$ *more* $\wedge_i \neg_i(f \wedge_i$ *more*$)$
 using *6* **by** *auto*
have *8*: $\vdash (f \wedge_i$ *more* $)$; $f^\star \wedge_i$ *more* $\wedge_i \neg_i$ $f \supset_i (f \wedge_i$ *more* $)$; $(f^\star \wedge_i$ *more* $)$
 using *5 7* **by** *auto*
have *9*: $\vdash f^\star \wedge_i$ *more* $\wedge_i \neg_i$ $f \equiv_i (f^\star \wedge_i$ *more*$) \wedge_i$ (*more* $\wedge_i \neg_i$ f)
 by *auto*
have *10*: $\vdash (f^\star \wedge_i$ *more*$) \wedge_i$ (*more* $\wedge_i \neg_i$ f) $\equiv_i (f \wedge_i$ *more* $)$; $f^\star \wedge_i$ (*more* $\wedge_i \neg_i$ f)
 using *1 prop06* **by** *auto*
from *1 8 9 10* **show** *?thesis* **by** *auto*
qed

lemma *CSAndMoreImpCSChop*:
$\vdash\ \ f^\star \wedge_i\ \ more\ \supset_i f^\star;\ f$
proof −
have *1*: $\vdash f^\star \wedge_i\ \ more\ \equiv_i (f \wedge_i\ \ more\ );\ f^\star$
by (*rule CSAndMoreEqvAndMoreChop*)
have *2*: $\vdash\ \ empty\ \vee_i\ \ \ more$
by *auto*
hence *3*: $\vdash f^\star \supset_i\ \ empty\ \vee_i\ \ (f^\star \wedge_i\ \ more\ )$
by *auto*
hence *4*: $\vdash (f \wedge_i\ \ more\ );\ f^\star \supset_i$
$(f \wedge_i\ \ more\ ) \vee_i\ \ ((f \wedge_i\ \ more\ );\ (f^\star \wedge_i\ more\ ))$
by (*rule ChopEmptyOrImpRule*)
have *5*: $\vdash f^\star \wedge_i\ \ more\ \wedge_i \neg_i\ \ f \supset_i (f \wedge_i\ \ more\ );\ (f^\star \wedge_i\ \ more\ )$
by (*rule CSMoreNotImpChopCSAndMore*)
have *6*: $\vdash f^\star \equiv_i\ \ empty\ \vee_i\ \ (f \wedge_i\ \ more\ );\ f^\star$
by (*rule ChopstarEqv*)
hence *7*: $\vdash f^\star;\ f \equiv_i f \vee_i\ \ ((f \wedge_i\ \ more\ );\ f^\star);\ f$
by (*rule EmptyOrChopEqvRule*)
have *8*: $\vdash (f \wedge_i\ \ more\ );\ (f^\star;\ f) \equiv_i ((f \wedge_i\ \ more\ );\ f^\star);\ f$
by (*rule ChopAssoc*)
have *9*: $\vdash (f^\star \wedge_i\ \ more\ ) \wedge_i \neg_i (f^\star;\ f) \supset_i$
$(f \wedge_i\ \ more\ );\ (f^\star \wedge_i\ \ more\ ) \wedge_i \neg_i ((f \wedge_i\ \ more\ );\ (f^\star;\ f))$
using *5 7 8* **by** *auto*
have *10*: $\vdash f \wedge_i\ \ more\ \supset_i\ \ more$
by *auto*
from *9 10* **show** *?thesis* **by** (*rule ChopContra*)
qed

lemma *NotEmptyEqvMore*:
$\vdash \neg_i\ empty \equiv_i more$
by *simp*

lemma *NotCSImpMore*:
$\vdash\ \ \neg_i\ \ (f^\star) \supset_i\ \ more$
proof −
have *1*: $\vdash empty \supset_i (f^\star)$ **using** *EmptyImpCS* **by** *blast*
hence *2*: $\vdash\ \ \neg_i\ empty \vee_i (f^\star)$ **using** *itl-prop*(*35*) **by** *metis*
from *2* **show** *?thesis* **using** *1 NotEmptyEqvMore itl-prop*(*31*) *prop02 prop27* **by** *blast*
qed

lemma *CSChopCSImpCS*:
$\vdash\ \ f^\star;\ f^\star \supset_i f^\star$
proof −
have *1*: $\vdash f^\star \equiv_i\ \ empty\ \vee_i\ \ (f \wedge_i\ \ more\ );\ f^\star$
by (*rule ChopstarEqv*)
hence *2*: $\vdash f^\star;\ f^\star \equiv_i f^\star \vee_i\ \ ((f \wedge_i\ \ more\ );\ f^\star);\ f^\star$
by (*rule EmptyOrChopEqvRule*)
have *21*: $\vdash f^\star;\ f^\star \wedge_i \neg_i (f^\star) \supset_i ((f \wedge_i\ \ more\ );\ f^\star);\ f^\star$

using *2* **by** (*simp add*: *or-d-def*)
have *22*: $\vdash \neg_i\ (f^\star) \equiv_i \neg_i empty \wedge_i \neg_i\ ((f \wedge_i\ \ more\ );\ f^\star)$
using *1 prop20* **by** *blast*
have *23*: $\vdash\ \neg_i\ (f^\star) \supset_i \neg_i\ ((f \wedge_i\ \ more\ );\ f^\star)$
using *2 22* **using** *itl-prop*(*31*) *itl-prop*(*32*) **by** *blast*
have *24*: $\vdash\ f^\star;\ f^\star \wedge_i \neg_i\ (f^\star) \supset_i \neg_i\ (f^\star)$
by *auto*
have *25*: $\vdash\ f^\star;\ f^\star \wedge_i \neg_i\ (f^\star) \supset_i \neg_i\ ((f \wedge_i\ \ more\ );\ f^\star)$
using *23 24 MP* **by** *auto*
have *3*: $\vdash f^\star;\ f^\star \wedge_i \neg_i\ (f^\star) \supset_i ((f \wedge_i\ \ more\ );\ f^\star);\ f^\star \wedge_i \neg_i\ ((f \wedge_i\ \ more\ );\ f^\star)$
using *21 25* **by** *auto*
have *4*: $\vdash (f \wedge_i\ \ more\ ); (\ f^\star;\ f^\star) \equiv_i ((f \wedge_i\ \ more\ );\ f^\star);\ f^\star$
by (*rule ChopAssoc*)
have *5*: $\vdash f^\star;\ f^\star \wedge_i \neg_i\ (f^\star) \supset_i (f \wedge_i\ \ more\ );\ (f^\star;\ f^\star) \wedge_i \neg_i\ ((f \wedge_i\ \ more\ );\ f^\star)$
using *3 4* **by** *auto*
have *6*: $\vdash f \wedge_i\ \ more \supset_i\ \ more$
by *auto*
from *5 6* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *ImpChopPlus*:
$\vdash\ \ f \supset_i f;f^\star$
proof −
have *1*: $\vdash f^\star \equiv_i empty \vee_i\ \ f;\ f^\star$ **by** (*rule CSEqvOrChopCS*)
hence *2*: $\vdash f;f^\star \equiv_i f;empty \vee_i\ \ f;\ (f;f^\star)$ **using** *ChopOrEqvRule* **by** *blast*
have *3*: $\vdash f;empty \equiv_i f$ **using** *ChopEmpty* **by** *blast*
from *2 3* **show** *?thesis* **by** *simp*
qed

lemma *ImpCS*:
$\vdash\ \ f \supset_i f^\star$
proof −
have *1*: $\vdash f \supset_i f;f^\star$ **by** (*rule ImpChopPlus*)
hence *2*: $\vdash f \supset_i\ \ empty\ \ \vee_i\ \ f;f^\star$ **by** *auto*
from *2* **show** *?thesis* **using** *CSEqvOrChopCS* **using** *prop15* **by** *blast*
qed

lemma *CSChopImpCS*:
$\vdash\ \ f^\star;\ f \supset_i f^\star$
proof −
have *1*: $\vdash f \supset_i f^\star$ **by** (*rule ImpCS*)
hence *2*: $\vdash f^\star;\ f \supset_i f^\star;\ f^\star$ **by** (*rule RightChopImpChop*)
have *3*: $\vdash f^\star;\ f^\star \supset_i f^\star$ **by** (*rule CSChopCSImpCS*)
from *2 3* **show** *?thesis* **using** *prop02* **by** *blast*
qed

lemma *ChopPlusImpCS*:
$\vdash\ \ f;f^\star\ \supset_i f^\star$
proof −

have *1*: $\vdash$ $f;f^\star \supset_i$ *empty* $\vee_i$ $f;f^\star$ **by** *auto*
from *1* **show** *?thesis* **using** *CSEqvOrChopCS* **using** *prop15* **by** *blast*
qed

lemma *CSChopEqvOrChopPlusChop*:
$\vdash$ $f^\star; g \equiv_i g \vee_i (f;f^\star) ; g$
proof −
have *1*: $\vdash f^\star \equiv_i$ *empty* $\vee_i$ $f;f^\star$ **by** (*rule CSEqvOrChopCS*)
from *1* **show** *?thesis* **using** *EmptyOrChopEqvRule* **by** *blast*
qed

lemma *CSElim*:
assumes $\vdash$ *empty* $\supset_i g$
$\vdash (f \wedge_i$ *more* $); g \supset_i g$
shows $\vdash f^\star \supset_i g$
proof −
have *1*: $\vdash f^\star \equiv_i$ *empty* $\vee_i$ $(f \wedge_i$ *more* $); f^\star$
by (*rule ChopstarEqv*)
have *2*: $\vdash$ *empty* $\supset_i g$
using *assms* **by** *blast*
have *3*: $\vdash (f \wedge_i$ *more* $); g \supset_i g$
using *assms* **by** *blast*
have *31*: $\vdash \neg_i g \supset_i$ *more*
using *2* **by** *auto*
have *32*: $\vdash \neg_i g \supset_i \neg_i ((f \wedge_i$ *more* $); g)$
using *3 prop27* **by** *blast*
have *33*: $\vdash f^\star \wedge_i$ *more* $\supset_i (f \wedge_i$ *more* $); f^\star$
using *1* **using** *CSAndMoreEqvAndMoreChop itl-prop*(*31*) **by** *blast*
have *34*: $\vdash f^\star \wedge_i \neg_i g \supset_i f^\star \wedge_i$ *more*
using *31* **by** *auto*
have *35*: $\vdash f^\star \wedge_i \neg_i g \supset_i (f \wedge_i$ *more* $); f^\star$
using *33 34* **by** *auto*
have *36*: $\vdash f^\star \wedge_i \neg_i g \supset_i \neg_i ((f \wedge_i$ *more* $); g)$
using *32* **by** *auto*
have *4*: $\vdash f^\star \wedge_i \neg_i g \supset_i (f \wedge_i$ *more* $); f^\star \wedge_i \neg_i ((f \wedge_i$ *more* $); g)$
using *35 36* **by** *auto*
have *5*: $\vdash f \wedge_i$ *more* $\supset_i$ *more*
by *auto*
from *4 5* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *CSCSImpCS*:
$\vdash (f^\star)^\star \supset_i f^\star$
proof −
have *1*: $\vdash$ *empty* $\supset_i f^\star$ **by** (*rule EmptyImpCS*)
have *2*: $\vdash (f^\star \wedge_i$ *more* $); f^\star \supset_i f^\star; f^\star$ **by** (*rule AndChopA*)
have *3*: $\vdash f^\star; f^\star \supset_i f^\star$ **by** (*rule CSChopCSImpCS*)
have *4*: $\vdash (f^\star \wedge_i$ *more* $); f^\star \supset_i f^\star$ **using** *2 3 prop02* **by** *blast*
from *1 4* **show** *?thesis* **using** *CSElim* **by** *blast*

qed

lemma *RightEmptyOrChopEqv*:
$\vdash$ g;(*empty* $\vee_i$ f) $\equiv_i$ g $\vee_i$ (g; f)
proof −
have *1*: $\vdash$ g;(*empty* $\vee_i$ f) $\equiv_i$ g;*empty* $\vee_i$ g;f **by** (*rule ChopOrEqv*)
have *2*: $\vdash$ g;*empty* $\equiv_i$ g **by** (*rule ChopEmpty*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RightEmptyOrChopEqvRule*:
assumes $\vdash$ f $\equiv_i$ *empty* $\vee_i$ *f1*
shows $\vdash$ g;f $\equiv_i$ g $\vee_i$ (g;*f1*)
proof −
have *1*: $\vdash$ f $\equiv_i$ *empty* $\vee_i$ *f1* **using** *assms* **by** *auto*
hence *2*: $\vdash$ g;f $\equiv_i$ g;(*empty* $\vee_i$ *f1*) **by** (*rule RightChopEqvChop*)
have *3*: $\vdash$ g;(*empty* $\vee_i$ *f1*) $\equiv_i$ g $\vee_i$ (g;*f1*) **by** (*rule RightEmptyOrChopEqv*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *ChopPlusEqvOrChopChopPlus*:
$\vdash$ (f;$f^\star$) $\equiv_i$ f $\vee_i$ f; (f;$f^\star$)
proof −
have *1*: $\vdash$ $f^\star$ $\equiv_i$ *empty* $\vee_i$ f;$f^\star$ **by** (*rule CSEqvOrChopCS*)
from *1* **show** *?thesis* **by** (*rule RightEmptyOrChopEqvRule*)
qed

lemma *CSAndEmptyEqvEmpty*:
$\vdash$ ($f^\star$) $\wedge_i$ *empty* $\equiv_i$ *empty*
using *EmptyImpCS* **by** *auto*

lemma *NotAndMoreChopAndEmpty*:
$\vdash$ $\neg_i$(((f $\wedge_i$ *more*);g) $\wedge_i$ *empty*)
by *auto*

lemma *NotChopAndMoreAndEmpty*:
$\vdash$ $\neg_i$((f;($g$$\wedge_i$ *more*)) $\wedge_i$ *empty*)
by *auto*

lemma *ChopCsAndEmptyEqvAndEmpty*:
$\vdash$ ((f;$f^\star$) $\wedge_i$ *empty*) $\equiv_i$ (f $\wedge_i$ *empty*)
proof −
have *1*: $\vdash$ ((f;$f^\star$) $\wedge_i$ *empty*) $\equiv_i$ ($f$$\wedge_i$ *empty*);($f^\star$ $\wedge_i$ *empty*)
using *ChopAndEmptyEqvEmptyChopEmpty* **by** *blast*
have *2*: $\vdash$ ($f$$\wedge_i$ *empty*);($f^\star$ $\wedge_i$ *empty*) $\equiv_i$ ($f$$\wedge_i$ *empty*);*empty*
using *CSAndEmptyEqvEmpty* **using** *RightChopEqvChop* **by** *blast*
have *3*: $\vdash$ ($f$$\wedge_i$ *empty*);*empty* $\equiv_i$ $f$$\wedge_i$ *empty*
by (*rule ChopEmpty*)
from *1 2 3* **show** *?thesis* **by** *auto*

qed

lemma *AndMoreChopAndMoreEqvAndMoreChop*:
$\vdash (f \wedge_i more);g \wedge_i more \equiv_i (f \wedge_i more);g$
apply *simp-all*
using *interval-prefix-length-good* **by** *auto*

lemma *ChopPlusEqv*:
$\vdash (f;f^\star) \equiv_i f \vee_i (f \wedge_i more); (f;f^\star)$
proof –
have *1*: $\vdash f^\star \equiv_i empty \vee_i (f \wedge_i more); f^\star$
by (*rule ChopstarEqv*)
have *2*: $\vdash f^\star \equiv_i empty \vee_i f;f^\star$
by (*rule CSEqvOrChopCS*)
hence *3*: $\vdash empty \vee_i f;f^\star \equiv_i empty \vee_i (f \wedge_i more);f^\star$
using *1 2 prop21* **by** *blast*
have *4*: $\vdash (f \wedge_i more);(f^\star) \equiv_i (f \wedge_i more);(empty \vee_i f;f^\star)$
using *2* **using** *RightChopEqvChop* **by** *blast*
hence *5*: $\vdash empty \vee_i f;f^\star \equiv_i empty \vee_i (f \wedge_i more);(empty \vee_i f;f^\star)$
using *3 4* **by** *auto*
have *6*: $\vdash (f \wedge_i more); (empty \vee_i f;f^\star) \equiv_i$
$(f \wedge_i more); empty \vee_i (f \wedge_i more); (f;f^\star)$
using *ChopOrEqv* **by** *blast*
have *7*: $\vdash (f \wedge_i more); empty \equiv_i f \wedge_i more$
using *ChopEmpty* **by** *blast*
have *8*: $\vdash empty \vee_i f;f^\star \equiv_i$
$empty \vee_i (f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)$
using *5 6 7* **by** *auto*
have *9*: $\vdash (empty \vee_i f;f^\star) \wedge_i more \equiv_i f;f^\star \wedge_i more$
by *auto*
have *10*: $\vdash (empty \vee_i (f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)) \wedge_i more \equiv_i$
$((f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)) \wedge_i more$
by *auto*
have *11*: $\vdash ((f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)) \wedge_i more \equiv_i$
$(f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)$
using *AndMoreChopAndMoreEqvAndMoreChop*
by (*metis 10 RightEmptyOrChopEqv itl-prop*(*31*) *itl-prop*(*32*) *prop15*)
have *12*: $\vdash f;f^\star \wedge_i more \equiv_i (f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star)$
using *8 9 10 11* **by** *auto*
have *13*: $\vdash f;f^\star \wedge_i empty \equiv_i f \wedge_i empty$
by (*rule ChopCsAndEmptyEqvAndEmpty*)
have *14*: $\vdash (f \wedge_i more) \vee_i (f \wedge_i more); (f;f^\star) \vee_i (f \wedge_i empty) \equiv_i$
$f \vee_i (f \wedge_i more);(f;f^\star)$
by *auto*
have *15*: $\vdash f;f^\star \equiv_i (f;f^\star \wedge_i empty) \vee_i (f;f^\star \wedge_i more)$
by *auto*
from *11 12 13 14 15* **show** *?thesis* **by** *auto*
qed

lemma *ChopPlusImpChopPlus*:
 assumes $\vdash\ f \supset_i g$
 shows $\vdash\ f;f^\star \supset_i g;g^\star$
proof −
 have *1*: $\vdash f \supset_i g$
  using *assms* **by** *auto*
 have *2*: $\vdash f;f^\star\ \equiv_i f \lor_i\ (f \land_i\ more); (f;f^\star)$
  by (*rule ChopPlusEqv*)
 have *3*: $\vdash g;g^\star\ \equiv_i g \lor_i\ (g \land_i\ more);(g;g^\star)$
  by (*rule ChopPlusEqv*)
 have *4*: $\vdash\ f;f^\star\ \land_i \neg_i (g;g^\star) \supset_i ((f \land_i\ more); (f;f^\star)) \land_i \neg_i ((g \land_i\ more); (g;g^\star))$
  using *1 2 3* **by** *auto*
 have *5*: $\vdash f \land_i\ more\ \supset_i g \land_i\ more$ **using** *1*
  by *auto*
 have *6*: $\vdash (f \land_i\ more); (f;f^\star)\ \supset_i (g \land_i\ more); (f;f^\star)$
  using *5* **by** (*rule LeftChopImpChop*)
 have *7*: $\vdash f;f^\star\ \land_i \neg_i (g;g^\star) \supset_i$
   $((g \land_i\ more); (f;f^\star)) \land_i \neg_i ((g \land_i\ more); (g;g^\star))$
  using *4 6* **by** *auto*
 have *8*: $\vdash g \land_i\ more\ \supset_i\ more$
  by *auto*
 from *7 8* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *ChopChopPlusImpChopPlus*:
$\vdash\ f; (f;f^\star)\ \supset_i f;f^\star$
proof −
 have *1*: $\vdash\ empty\ \lor_i\ more$ **by** *auto*
 hence *2*: $\vdash f \supset_i\ empty\ \lor_i\ (f \land_i\ more)$ **by** *auto*
 hence *3*: $\vdash f; (f;f^\star)\ \supset_i (f;f^\star)\ \lor_i\ (f \land_i\ more);(f;f^\star)$ **by** (*rule EmptyOrChopImpRule*)
 have *4*: $\vdash\ f;f^\star \equiv_i f \lor_i\ (f \land_i\ more); (f;f^\star)$ **by** (*rule ChopPlusEqv*)
 hence *5*: $\vdash (f \land_i\ more); (f;f^\star)\ \supset_i f;f^\star$ **by** *auto*
 from *3 5* **show** *?thesis* **using** *ChopPlusImpCS RightChopImpChop* **by** *blast*
qed

lemma *CSImpCS*:
 assumes $\vdash\ f \supset_i g$
 shows $\vdash f^\star \supset_i g^\star$
proof −
 have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *auto*
 hence *2*: $\vdash f;f^\star \supset_i g;g^\star$ **by** (*rule ChopPlusImpChopPlus*)
 hence *3*: $\vdash\ empty\ \lor_i\ f;f^\star\ \supset_i\ empty\ \lor_i\ g;g^\star$ **by** *auto*
 from *2 3* **show** *?thesis* **using** *CSEqvOrChopCS prop14 prop15* **by** *blast*
qed

lemma *ChopPlusIntro*:
 assumes $\vdash\ f \land_i \neg_i\ g \supset_i (g \land_i\ more); f$
 shows $\vdash f \supset_i g;g^\star$
proof −
 have *1*: $\vdash f \land_i \neg_i\ g \supset_i (g \land_i\ more); f$ **using** *assms* **by** *auto*

have *2*: $\vdash g;g^\star \ \equiv_i g \vee_i \ (g \wedge_i \ more); (g;g^\star)$ **by** (*rule ChopPlusEqv*)
have *3*: $\vdash f \wedge_i \neg_i (g;g^\star) \supset_i$
$(g \wedge_i \ more); f \wedge_i \neg_i ((g \wedge_i \ more); (g;g^\star))$ **using** *1 2* **by** *auto*
have *4*: $\vdash g \wedge_i \ more \supset_i \ more$ **by** *auto*
from *3 4* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *ChopPlusElim*:
assumes $\vdash \ f \supset_i g$
$\vdash (f \wedge_i \ more); g \supset_i g$
shows $\vdash f;f^\star \ \supset_i g$
proof –
have *1*: $\vdash f;f^\star \ \equiv_i f \vee_i \ (f \wedge_i \ more); (f;f^\star)$ **by** (*rule ChopPlusEqv*)
have *2*: $\vdash f \supset_i g$ **using** *assms* **by** *blast*
hence *21*: $\vdash \neg_i g \supset_i \neg_i f$ **by** *auto*
have *3*: $\vdash (f \wedge_i \ more); g \supset_i g$ **using** *assms* **by** *blast*
hence *31*: $\vdash \neg_i \ g \supset_i \neg_i ((f \wedge_i more); g)$ **using** *prop27* **by** *blast*
hence *32*: $\vdash f;f^\star \wedge_i \neg_i \ g \supset_i \neg_i ((f \wedge_i more); g)$ **by** *auto*
have *33*: $\vdash f;f^\star \wedge_i \neg_i \ g \supset_i (f \wedge_i \ more); (f;f^\star)$ **using** *1 21* **by** *auto*
have *4*: $\vdash \ f;f^\star \wedge_i \neg_i \ g \supset_i$
$(f \wedge_i \ more); (f;f^\star) \ \wedge_i \neg_i ((f \wedge_i more); g)$ **using** *31 33* **by** *auto*
have *5*: $\vdash f \wedge_i \ more \ \supset_i \ more$ **by** *auto*
from *4 5* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *ChopPlusElimWithoutMore*:
assumes $\vdash \ f \supset_i g$
$\vdash f; g \supset_i g$
shows $\vdash f;f^\star \ \supset_i g$
proof –
have *1*: $\vdash f \supset_i g$ **using** *assms* **by** *blast*
have *2*: $\vdash (f; g) \supset_i g$ **using** *assms* **by** *blast*
have *3*: $\vdash (f \wedge_i \ more); g \supset_i f; g$ **by** (*rule AndChopA*)
have *4*: $\vdash (f \wedge_i \ more); g \supset_i g$ **using** *2 3 prop02* **by** *blast*
from *1 4* **show** *?thesis* **using** *ChopPlusElim* **by** *blast*
qed

lemma *ChopPlusEqvChopPlus*:
assumes $\vdash \ f \equiv_i g$
shows $\vdash f;f^\star \ \equiv_i g;g^\star$
proof –
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash f \supset_i g$ **by** *auto*
hence *3*: $\vdash f;f^\star \ \supset_i g;g^\star$ **by** (*rule ChopPlusImpChopPlus*)
have *4*: $\vdash g \supset_i f$ **using** *1* **by** *auto*
hence *5*: $\vdash \ g;g^\star \supset_i f;f^\star$ **by** (*rule ChopPlusImpChopPlus*)
from *3 5* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *CSEqvCS*:

assumes $\vdash\ f \equiv_i g$
shows $\vdash f^\star \equiv_i g^\star$
proof $-$
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash f;f^\star \equiv_i g;g^\star$ **by** (*rule ChopPlusEqvChopPlus*)
hence *3*: $\vdash$ *empty* $\vee_i f;f^\star \equiv_i$ *empty* $\vee_i g;g^\star$ **by** *auto*
from *3* **show** *?thesis* **using** *CSEqvOrChopCS* **using** *assms* **by** *auto*
qed

lemma *AndCSA*:
$\vdash\ (f \wedge_i g)^\star \supset_i f^\star$
proof $-$
have *1*: $\vdash f \wedge_i g \supset_i f$ **by** *auto*
from *1* **show** *?thesis* **using** *CSImpCS* **by** *blast*
qed

lemma *AndCSB*:
$\vdash\ (f \wedge_i g)^\star \supset_i g^\star$
proof $-$
have *1*: $\vdash f \wedge_i g \supset_i g$ **by** *auto*
from *1* **show** *?thesis* **using** *CSImpCS* **by** *blast*
qed

lemma *CSIntro*:
assumes $\vdash\ f \wedge_i$ *more* $\supset_i (g \wedge_i$ *more* $); f$
shows $\vdash f \supset_i g^\star$
proof $-$
have *1*: $\vdash f \wedge_i$ *more* $\supset_i (g \wedge_i$ *more* $); f$
using *assms* **by** *auto*
have *2*: $\vdash$ *more* $\equiv_i \neg_i$ *empty*
by *auto*
have *3*: $\vdash f \wedge_i \neg_i$ *empty* $\supset_i (g \wedge_i$ *more* $); f$
using *1 2* **by** *auto*
have *4*: $\vdash g^\star \equiv_i$ *empty* $\vee_i (g \wedge_i$ *more* $); g^\star$
by (*rule ChopstarEqv*)
hence *41*: $\vdash \neg_i($*empty* $\vee_i (g \wedge_i$ *more* $); g^\star) \equiv_i \neg_i$*empty* $\wedge_i \neg_i((g \wedge_i$ *more* $); g^\star)$
using *prop20 prop21* **by** *blast*
have *411*: $\vdash \neg_i$*empty* $\wedge_i \neg_i((g \wedge_i$ *more* $); g^\star) \equiv_i$ *more* $\wedge_i \neg_i((g \wedge_i$ *more* $); g^\star)$
using *NotEmptyEqvMore* **using** *prop06* **by** *blast*
have *42*: $\vdash \neg_i(g^\star) \equiv_i$ *more* $\wedge_i \neg_i((g \wedge_i$ *more* $); g^\star)$
using *4 41 411 prop01 prop03* **by** *blast*
have *43*: $\vdash f \wedge_i \neg_i(g^\star) \supset_i f \wedge_i$ *more* $\wedge_i \neg_i ((g \wedge_i$ *more* $); g^\star)$
using *42* **using** *itl-prop*(*31*) *prop12* **by** *blast*
have *44*: $\vdash f \wedge_i$ *more* $\wedge_i \neg_i ((g \wedge_i$ *more* $); g^\star) \supset_i (g \wedge_i$ *more* $); f \wedge_i \neg_i ((g \wedge_i$ *more* $); g^\star)$
using *3 43* **by** *auto*
have *5*: $\vdash f \wedge_i \neg_i (g^\star) \supset_i$
$(g \wedge_i$ *more* $); f \wedge_i \neg_i ((g \wedge_i$ *more* $); g^\star)$

using *43 44 prop02* **by** *auto*
have *6*: $\vdash g \wedge_i$ *more* $\supset_i$ *more*
by *auto*
from *5 6* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *CSElimWithoutMore*:
assumes $\vdash$ *empty* $\supset_i g$
$\vdash f; g \supset_i g$
shows $\vdash f^\star \supset_i g$
proof −
have *1*: $\vdash$ *empty* $\supset_i g$ **using** *assms* **by** *blast*
have *2*: $\vdash f; g \supset_i g$ **using** *assms* **by** *blast*
have *3*: $\vdash (f \wedge_i$ *more* $); g \supset_i f; g$ **by** (*rule AndChopA*)
have *4*: $\vdash (f \wedge_i$ *more* $); g \supset_i g$ **using** *2 3 prop02* **by** *blast*
from *1 4* **show** *?thesis* **using** *CSElim* **by** *blast*
qed

lemma *ChopAssocB*:
$\vdash (f;g);h \equiv_i f;(g;h)$
using *ChopAssoc itl-prop*(*30*) **by** *blast*

lemma *CSChopEqvChopOrRule*:
assumes $\vdash f \equiv_i (g^\star; h)$
shows $\vdash f \equiv_i (g; f) \vee_i h$
proof −
have *1*: $\vdash f \equiv_i (g^\star; h)$ **using** *assms* **by** *auto*
have *2*: $\vdash g^\star \equiv_i$ *empty* $\vee_i (g; g^\star)$ **by** (*rule CSEqvOrChopCS*)
hence *3*: $\vdash g^\star; h \equiv_i h \vee_i ((g; g^\star); h)$ **by** (*rule EmptyOrChopEqvRule*)
have *4*: $\vdash (g; g^\star); h \equiv_i g; (g^\star; h)$ **by** (*rule ChopAssocB*)
hence *41*: $\vdash g^\star; h \equiv_i h \vee_i g; (g^\star; h)$ **using** *3* **by** *auto*
have *5*: $\vdash g; f \equiv_i g; (g^\star; h)$ **using** *1* **by** (*rule RightChopEqvChop*)
hence *6*: $\vdash (g^\star; h) \equiv_i h \vee_i g; f$ **using** *41* **by** *auto*
hence *61*: $\vdash (g^\star; h) \equiv_i (g; f) \vee_i h$ **by** *auto*
from *1 61* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *CSChopIntroRule*:
assumes $\vdash f \wedge_i \neg_i h \supset_i g; f$
$\vdash g \supset_i$ *more*
shows $\vdash f \supset_i g^\star; h$
proof −
have *1*: $\vdash f \wedge_i \neg_i h \supset_i g; f$ **using** *assms* **by** *blast*
have *2*: $\vdash g \supset_i$ *more* **using** *assms* **by** *blast*
hence *3*: $\vdash g \supset_i g \wedge_i$ *more* **by** *auto*
hence *4*: $\vdash g; f \supset_i (g \wedge_i$ *more* $); f$ **by** (*rule LeftChopImpChop*)
have *5*: $\vdash f \supset_i (g \wedge_i$ *more* $); f \vee_i h$ **using** *1 4* **by** *auto*
have *6*: $\vdash g^\star \equiv_i$ *empty* $\vee_i (g \wedge_i$ *more* $); g^\star$ **by** (*rule ChopstarEqv*)
hence *7*: $\vdash (g^\star); h \equiv_i h \vee_i ((g \wedge_i$ *more* $); g^\star); h$ **by** (*rule EmptyOrChopEqvRule*)
have *8*: $\vdash ((g \wedge_i$ *more* $); g^\star); h \equiv_i (g \wedge_i$ *more* $); (g^\star; h)$ **by** (*rule ChopAssocB*)

have 9: $\vdash (g^\star);\ h \equiv_i h \vee_i\ (g \wedge_i\ more\ );\ (g^\star;\ h)$ **using** *7 8* **by** *auto*
have 10: $\vdash f \wedge_i \neg_i (g^\star;\ h) \supset_i (g \wedge_i more);\ f \wedge_i \neg_i ((g \wedge_i more);\ (g^\star;\ h))$ **using** *5 9* **by** *auto*
have 11: $\vdash g \wedge_i\ more\ \supset_i\ more$ **by** *auto*
from *10 11* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *DiamondAndEmptyEqvAndEmpty*:
$\vdash \diamond f \wedge_i empty \equiv_i f \wedge_i empty$
proof $-$
have 1: $\vdash \diamond f \wedge_i empty \supset_i f \wedge_i empty$ **by** *auto*
have 2: $\vdash f \wedge_i empty \supset_i \diamond f \wedge_i empty$ **by** *auto*
from *1 2* **show** *?thesis* **using** *itl-prop(31)* **by** *blast*
qed

lemma *InitAndEmptyEqvAndEmpty*:
$\vdash (init\ w) \wedge_i empty \equiv_i w \wedge_i empty$
proof $-$
have 1: $\vdash (init\ w) \wedge_i empty \equiv_i (w \wedge_i empty);true_i \wedge_i empty$
by *auto*
have 2: $\vdash (w \wedge_i empty);true_i \wedge_i empty \equiv_i (w \wedge_i empty);(true_i \wedge_i empty)$
using *ChopAndEmptyEqvEmptyChopEmpty* **by** *auto*
have 3: $\vdash (w \wedge_i empty);(true_i \wedge_i empty) \equiv_i (w \wedge_i empty);empty$
using *RightChopEqvChop itl-prop(17)* **by** *blast*
have 4: $\vdash (w \wedge_i empty);empty \equiv_i w \wedge_i empty$
using *ChopEmpty* **by** *blast*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *InitAndNotBoxInitImpNotEmpty*:
$\vdash init\ w \wedge_i \neg_i(\Box\ (init\ w)) \supset_i \neg_i\ empty$
proof $-$
have 1: $\vdash (init\ w) \wedge_i empty \equiv_i w \wedge_i empty$ **by** (*rule InitAndEmptyEqvAndEmpty*)
have 2: $\vdash \neg_i(\Box\ (init\ w)) \wedge_i empty \equiv_i \diamond \neg_i (init\ w) \wedge_i empty$ **by** *auto*
have 3: $\vdash \diamond \neg_i (init\ w) \wedge_i empty \equiv_i \neg_i (init\ w) \wedge_i empty$ **by** *auto*
have 4: $\vdash \neg_i (init\ w) \equiv_i (init\ \neg_i\ w)$ **by** *auto*
have 5: $\vdash \neg_i (init\ w) \wedge_i empty \equiv_i \neg_i w \wedge_i empty$ **using** *4 InitAndEmptyEqvAndEmpty* **by** *auto*
have 6: $\vdash \neg_i(\Box\ (init\ w)) \wedge_i empty \equiv_i \neg_i w \wedge_i empty$ **using** *2 3 5 prop03* **by** *blast*
have 7: $\vdash \neg_i(init\ w \wedge_i \neg_i(\Box\ (init\ w)) \wedge_i empty)$ **using** *1 6* **by** *auto*
from *7* **show** *?thesis* **by** *auto*
qed

lemma *BoxImpTrueChopAndEmpty*:
$\vdash \Box f \supset_i true_i;(f \wedge_i empty)$
by *auto*

lemma *BoxInitAndMoreImpBoxInitAndMoreAndFinInit*:
$\vdash \Box(init\ w) \wedge_i\ more\ \supset_i (\Box\ (init\ w) \wedge_i\ more\ ) \wedge_i\ fin\ (\ init\ w)$
proof $-$

have *1*: $\vdash$ *fin* (*init w*) $\equiv_i$ *true*$_i$; (*init w* $\wedge_i$ *empty*) **using** *FinEqvTrueChopAndEmpty* **by** *blast*
have *2*: $\vdash$ $\Box$(*init w*) $\supset_i$ *true*$_i$;(*init w* $\wedge_i$ *empty*) **by** (*rule BoxImpTrueChopAndEmpty*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *CSImpBox*:
assumes $\vdash$ *f* $\supset_i$ *empty* $\vee_i$ ($\Box$ (*init w*) $\wedge_i$ *more*); *f*
shows $\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ $\Box$ (*init w*)
proof −
have *1*: $\vdash$ *f* $\supset_i$ *empty* $\vee_i$ ($\Box$(*init w*) $\wedge_i$ *more*); *f*
using *assms* **by** *auto*
have *2*: $\vdash$ *init w* $\wedge_i$ $\neg_i$($\Box$ (*init w*)) $\supset_i$ $\neg_i$ *empty*
by (*rule InitAndNotBoxInitImpNotEmpty*)
have *3*: $\vdash$ *init w* $\wedge_i$ *f* $\wedge_i$ $\neg_i$($\Box$ (*init w*)) $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ *more*); *f*
using *1 2* **by** *auto*
have *4*: $\vdash$ $\Box$ (*init w*) $\wedge_i$ *more* $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ *more*) $\wedge_i$ *fin* (*init w*)
by (*rule BoxInitAndMoreImpBoxInitAndMoreAndFinInit*)
hence *5*: $\vdash$ ($\Box$(*init w*) $\wedge_i$ *more*); *f* $\supset_i$ (($\Box$ (*init w*) $\wedge_i$ *more*) $\wedge_i$ *fin* (*init w*)); *f*
by (*rule LeftChopImpChop*)
have *6*: $\vdash$ (($\Box$ (*init w*) $\wedge_i$ *more*) $\wedge_i$ *fin* (*init w*)); *f* $\equiv_i$
($\Box$(*init w*) $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)
by (*rule AndFinChopEqvStateAndChop*)
have *7*: $\vdash$ $\neg_i$($\Box$(*init w*)) $\supset_i$ ($\Box$ (*init w*)) *yields* $\neg_i$($\Box$ (*init w*))
by (*rule NotBoxStateImpBoxYieldsNotBox*)
have *8*: $\vdash$ ($\Box$(*init w*)) *yields* $\neg_i$ ($\Box$ (*init w*)) $\supset_i$
($\Box$ (*init w*) $\wedge_i$ *more*) *yields* $\neg_i$($\Box$(*init w*))
by (*rule AndYieldsA*)
have *9*: $\vdash$ ($\Box$(*init w*) $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*) $\wedge_i$ ($\Box$(*init w*) $\wedge_i$ *more*) *yields* $\neg_i$($\Box$ (*init w*))
$\supset_i$
($\Box$ (*init w*) $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$ ($\Box$ (*init w*)))
by (*rule ChopAndYieldsImp*)
have *10*: $\vdash$ (*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*)) $\supset_i$
($\Box$(*init w*) $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*)))
using *3 5 6 7 8 9* **by** *auto*
have *11*: $\vdash$ ($\Box$(*init w*) $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*))) $\supset_i$
more ; ((*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*)))
by (*rule AndChopB*)
have *12*: $\vdash$ (*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*)) $\supset_i$
more ; ((*init w* $\wedge_i$ *f*) $\wedge_i$ $\neg_i$($\Box$ (*init w*)))
using *10 11* **by** *auto*
from *12* **show** *?thesis* **using** *MoreChopContra* **by** *blast*
qed

lemma *BoxCSEqvBox*:
$\vdash$ *init w* $\wedge_i$ ($\Box$(*init w*))* $\equiv_i$ $\Box$ (*init w*)
proof −
have *1*: $\vdash$ ($\Box$ (*init w*))* $\equiv_i$ *empty* $\vee_i$ ($\Box$ (*init w*) $\wedge_i$ *more*); ($\Box$ (*init w*))*
by (*rule ChopstarEqv*)
hence *2*: $\vdash$ ($\Box$ (*init w*))* $\supset_i$ *empty* $\vee_i$ ($\Box$ (*init w*) $\wedge_i$ *more*); ($\Box$ (*init w*))*
using *itl-prop*(*31*) **by** *blast*

hence *3*: $\vdash$ *init w* $\wedge_i$ ($\Box$ (*init w*))* $\supset_i$ $\Box$ (*init w*)
  by (*rule CSImpBox*)
have *11*: $\vdash$ $\Box$ (*init w*) $\supset_i$ (*init w*)
  by *auto*
have *12*: $\vdash$ $\Box$(*init w*) $\supset_i$ ($\Box$ (*init w*))*
  by (*rule ImpCS*)
have *13*: $\vdash$ $\Box$ (*init w*) $\supset_i$ *init w* $\wedge_i$ ($\Box$ (*init w*))*
  using *11 12* **using** *itl-prop*(*32*) **by** *blast*
from *3 13* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *BoxStateAndCSEqvCS*:
$\vdash$ $\Box$(*init w*) $\wedge_i$ $f^\star$ $\equiv_i$ *init w* $\wedge_i$ ($\Box$(*init w*) $\wedge_i$ f)*
proof −
have *1*: $\vdash$ $\Box$ (*init w*) $\supset_i$ *init w* **by** *auto*
have *2*: $\vdash$ $f^\star$ $\wedge_i$ *more* $\equiv_i$ (f $\wedge_i$ *more*); $f^\star$ **by** (*rule CSAndMoreEqvAndMoreChop*)
have *3*: $\vdash$ $\Box$(*init w*) $\wedge_i$ ((f $\wedge_i$ *more*); $f^\star$) $\equiv_i$
  ($\Box$ (*init w*) $\wedge_i$ f $\wedge_i$ *more*); ($\Box$ (*init w*) $\wedge_i$ $f^\star$) **by** (*rule BoxStateAndChopEqvChop*)
have *4*: $\vdash$ $\Box$ (*init w*) $\wedge_i$ f $\wedge_i$ *more* $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ f) $\wedge_i$ *more* **by** *auto*
hence *5*: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f $\wedge_i$ *more*); ($\Box$ (*init w*) $\wedge_i$ $f^\star$) $\supset_i$
  (($\Box$ (*init w*) $\wedge_i$ f) $\wedge_i$ *more*); ($\Box$ (*init w*) $\wedge_i$ $f^\star$) **by** (*rule LeftChopImpChop*)
have *6*: $\vdash$ ($\Box$(*init w*) $\wedge_i$ $f^\star$) $\wedge_i$ *more* $\supset_i$
  (($\Box$ (*init w*) $\wedge_i$ f) $\wedge_i$ *more*); ($\Box$ (*init w*) $\wedge_i$ $f^\star$) **using** *2 3 5* **by** *auto*
hence *7*: $\vdash$ $\Box$ (*init w*) $\wedge_i$ $f^\star$ $\supset_i$ ($\Box$ (*init w*) $\wedge_i$ f)* **by** (*rule CSIntro*)
have *71*: $\vdash$ *init w* $\wedge_i$ $\Box$ (*init w*) $\wedge_i$ $f^\star$ $\supset_i$ *init w* $\wedge_i$ ($\Box$ (*init w*) $\wedge_i$ f)* **using** *7 prop12* **by** *blast*
have *8*: $\vdash$ $\Box$(*init w*) $\wedge_i$ $f^\star$ $\supset_i$ *init w* $\wedge_i$ ($\Box$ (*init w*) $\wedge_i$ f)* **using** *1 71 prop37* **by** *blast*
have *11*: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f)* $\supset_i$ ($\Box$ (*init w*))* **by** (*rule AndCSA*)
have *12*: $\vdash$ *init w* $\wedge_i$ ($\Box$ (*init w*))* $\equiv_i$ $\Box$ (*init w*) **by** (*rule BoxCSEqvBox*)
have *13*: $\vdash$ ($\Box$ (*init w*) $\wedge_i$ f)* $\supset_i$ $f^\star$ **by** (*rule AndCSB*)
have *14*: $\vdash$ *init w* $\wedge_i$ ($\Box$ (*init w*) $\wedge_i$ f)* $\supset_i$ *init w* $\wedge_i$ ($\Box$ (*init w*))* $\wedge_i$ $f^\star$ **using** *11 13* **by** *auto*
have *15*: $\vdash$ *init w* $\wedge_i$ ($\Box$ (*init w*))* $\wedge_i$ $f^\star$ $\supset_i$ $\Box$ (*init w*) $\wedge_i$ $f^\star$ **using** *12* **by** *auto*
have *16*: $\vdash$ *init w* $\wedge_i$ ($\Box$ (*init w*) $\wedge_i$ f)* $\supset_i$ $\Box$ (*init w*) $\wedge_i$ $f^\star$ **using** *14 15 prop02* **by** *blast*
from *8 16* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *BaCSImpCS*:
$\vdash$ *ba* (f $\supset_i$ g) $\supset_i$ $f^\star$ $\supset_i$ $g^\star$
proof −
have *1*: $\vdash$ $f^\star$ $\equiv_i$ *empty* $\vee_i$ (f $\wedge_i$ *more*); $f^\star$
  by (*rule ChopstarEqv*)
have *2*: $\vdash$ $g^\star$ $\equiv_i$ *empty* $\vee_i$ (g $\wedge_i$ *more*); $g^\star$
  by (*rule ChopstarEqv*)
have *21*: $\vdash$ $\neg_i$($g^\star$) $\equiv_i$ $\neg_i$*empty* $\wedge_i$ $\neg_i$((g $\wedge_i$ *more*); $g^\star$)
  using *2 prop20* **by** *blast*
hence *22*: $\vdash$ $\neg_i$($g^\star$) $\equiv_i$ *more* $\wedge_i$ $\neg_i$((g $\wedge_i$ *more*); $g^\star$)
  by (*meson NotCSImpMore itl-prop*(*31*) *itl-prop*(*32*) *prop18 NotEmptyEqvMore*)
have *3*: $\vdash$ $f^\star$ $\wedge_i$ $\neg_i$ ($g^\star$) $\supset_i$
  (*empty* $\vee_i$ (f $\wedge_i$ *more*); $f^\star$) $\wedge_i$ *more* $\wedge_i$ $\neg_i$ ((g $\wedge_i$ *more*); $g^\star$)
  using *1 22 prop22* **by** *blast*
have *31*: $\vdash$ (*empty* $\vee_i$ (f $\wedge_i$ *more*); $f^\star$) $\wedge_i$ *more* $\equiv_i$ (f $\wedge_i$ *more*); $f^\star$ $\wedge_i$ *more*

by *auto*
have *32*: $\vdash f^\star \wedge_i \neg_i (g^\star) \supset_i (f \wedge_i \textit{ more }); f^\star \wedge_i \neg_i ((g \wedge_i \textit{ more }); g^\star)$
using *3 31* **by** *auto*
have *4*: $\vdash (f \supset_i g) \supset_i (f \wedge_i \textit{ more } \supset_i g \wedge_i \textit{ more })$
by *auto*
hence *5*: $\vdash \textit{ ba } (f \supset_i g) \supset_i \textit{ ba } (f \wedge_i \textit{ more } \supset_i g \wedge_i \textit{ more })$
by (*rule BaImpBa*)
have *6*: $\vdash \textit{ ba } (f \wedge_i \textit{ more } \supset_i g \wedge_i \textit{ more }) \supset_i$
$(f \wedge_i \textit{ more }); f^\star \supset_i (g \wedge_i \textit{ more }); f^\star$
by (*rule BaLeftChopImpChop*)
have *7*: $\vdash \textit{ ba } (f \supset_i g) \wedge_i (f \wedge_i \textit{ more }); f^\star \supset_i (g \wedge_i \textit{ more }); f^\star$
using *5 6* **by** *auto*
have *8*: $\vdash (g \wedge_i \textit{ more }); f^\star \wedge_i \neg_i ((g \wedge_i \textit{ more }); g^\star)$
$\supset_i (g \wedge_i \textit{ more }); (f^\star \wedge_i \neg_i (g^\star))$
by (*rule ChopAndNotChopImp*)
have *9*: $\vdash (g \wedge_i \textit{ more }); (f^\star \wedge_i \neg_i (g^\star)) \supset_i \textit{ more }; (f^\star \wedge_i \neg_i (g^\star))$
by (*rule AndChopB*)
have *10*: $\vdash \textit{ ba } (f \supset_i g) \supset_i \textit{ more }; (f^\star \wedge_i \neg_i (g^\star)) \supset_i$
$\textit{more }; (\textit{ ba } (f \supset_i g) \wedge_i f^\star \wedge_i \neg_i (g^\star))$
by (*rule BaChopImpChopBa*)
have *11*: $\vdash \textit{ ba } (f \supset_i g) \wedge_i f^\star \wedge_i \neg_i (g^\star) \supset_i$
$\textit{more }; (\textit{ ba } (f \supset_i g) \wedge_i f^\star \wedge_i \neg_i (g^\star))$
using *32 7 8 9 10* **by** *auto*
hence *12*: $\vdash \neg_i ((\textit{ba } (f \supset_i g)) \wedge_i (f^\star) \wedge_i (\neg_i (g^\star)))$
using *MoreChopLoop* **by** *blast*
from *12* **show** *?thesis* **using** *prop04 MP itl-prop*(*31*) **by** *blast*
qed

lemma *BaCSEqvCS*:
$\vdash \textit{ ba } (f \equiv_i g) \supset_i (f^\star \equiv_i g^\star)$
proof −
have *1*: $\vdash \textit{ba } (f \equiv_i g) \equiv_i \textit{ba } (f \supset_i g) \wedge_i \textit{ba } (g \supset_i f)$ **by** *auto*
have *2*: $\vdash \textit{ba } (f \supset_i g) \supset_i (f^\star \supset_i g^\star)$ **by** (*rule BaCSImpCS*)
have *3*: $\vdash \textit{ba } (g \supset_i f) \supset_i (g^\star \supset_i f^\star)$ **by** (*rule BaCSImpCS*)
have *4*: $\vdash \textit{ba } (f \equiv_i g) \supset_i (f^\star \supset_i g^\star) \wedge_i (g^\star \supset_i f^\star)$ **using** *1 2 3* **by** *auto*
have *5*: $\vdash (f^\star \supset_i g^\star) \wedge_i (g^\star \supset_i f^\star) \equiv_i (f^\star \equiv_i g^\star)$ **by** *auto*
from *4 5* **show** *?thesis* **by** *auto*
qed

lemma *BaAndCSImport*:
$\vdash \textit{ ba } f \wedge_i g^\star \supset_i (f \wedge_i g)^\star$
proof −
have *1*: $\vdash f \supset_i (g \supset_i f \wedge_i g)$ **by** *auto*
hence *2*: $\vdash \textit{ ba } f \supset_i \textit{ ba } (g \supset_i f \wedge_i g)$ **by** (*rule BaImpBa*)
have *3*: $\vdash \textit{ ba } (g \supset_i f \wedge_i g) \supset_i g^\star \supset_i (f \wedge_i g)^\star$ **by** (*rule BaCSImpCS*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *CSSkip*:
$\vdash \textit{skip}^\star$

by (*metis ChopPlusImpCS EmptyImpCS EmptyNextInducta next-d-def*)

5.8 Properties of While

lemma *WhileEqvIf*:
$\vdash$ *while* (*init w*) *do* $f \equiv_i$ *if* $_i$ (*init w*) *then* (f; (*while* (*init w*) *do* f)) *else* *empty*
proof $-$
have *1*: $\vdash$ *while* (*init w*) *do* $f \equiv_i$ ((*init w*) $\wedge_i f)^\star \wedge_i$ *fin* $\neg_i$ (*init w*)
by (*simp add*: *while-d-def*)
have *2*: $\vdash$ (*init w* $\wedge_i f)^\star \equiv_i$ *empty* $\vee_i$ ((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$)
by (*rule CSEqvOrChopCS*)
have *21*: $\vdash$ ((*init w*) $\wedge_i f)^\star \wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
(*empty* $\vee_i$ ((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$)) $\wedge_i$ *fin* $\neg_i$ (*init w*)
using *2 prop06* **by** *blast*
have *22*: $\vdash$ (*empty* $\vee_i$ ((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$)) $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
(*empty* $\wedge_i$ *fin* $\neg_i$(*init w*)) $\vee_i$ (((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*))
by *auto*
have *3*: $\vdash$ *empty* $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$ $\neg_i$ (*init w*) $\wedge_i$ *empty*
using *AndFinEqvChopAndEmpty EmptyChop prop03* **by** *blast*
have *4*: $\vdash$ (*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star \equiv_i$ *init w* $\wedge_i$ (f; (*init w* $\wedge_i f)^\star$)
by (*rule StateAndChop*)
have *41*: $\vdash$ ((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
init w $\wedge_i$ (f; (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*)
using *4* **by** *auto*
have *42*: $\vdash$ *init w* $\wedge_i$ (f; (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
init w $\wedge_i$ (f; (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* (*init* $\neg_i$ *w*)
by (*simp*)
have *5*: $\vdash$ (f; ((*init w* $\wedge_i f)^\star$)) $\wedge_i$ (*fin* (*init* $\neg_i$ *w*))
$\equiv_i$ (f; ((*init w* $\wedge_i f)^\star \wedge_i$ (*fin* (*init* $\neg_i$ *w*))))
by (*rule ChopAndFin*)
have *51*: $\vdash$ (f; ((*init w* $\wedge_i f)^\star \wedge_i$ (*fin* (*init* $\neg_i$ *w*)))) $\equiv_i$
(f; ((*init w* $\wedge_i f)^\star \wedge_i$ (*fin* $\neg_i$ (*init* *w*))))
by (*simp*)
have *52*: $\vdash$ *init w* $\wedge_i$ (f; (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
init w $\wedge_i$ (f; ((*init w* $\wedge_i f)^\star \wedge_i$ *fin* $\neg_i$ (*init* *w*)))
using *42 5 51* **by** *auto*
have *6*: $\vdash$ f; ((*init w* $\wedge_i f)^\star \wedge_i$ *fin* $\neg_i$ (*init w*)) $\equiv_i$ f; *while* (*init w*) *do* f
by (*simp add*: *while-d-def*)
have *61*: $\vdash$ *init w* $\wedge_i$ (f; ((*init w* $\wedge_i f)^\star \wedge_i$ *fin* $\neg_i$ (*init* *w*))) $\equiv_i$
init w $\wedge_i$ (f; *while* (*init w*) *do* f) **using** *6*
by *auto*
have *62*: $\vdash$ (*empty* $\wedge_i$ *fin* $\neg_i$ (*init w*)) $\vee_i$ (((*init w* $\wedge_i f$); (*init w* $\wedge_i f)^\star$) $\wedge_i$ *fin* $\neg_i$ (*init w*))
$\equiv_i$ ($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (f; *while* (*init w*) *do* f))
using *21 22 3 4 52 61* **by** *auto*
have *7*: $\vdash$ *while* (*init w*) *do* f
$\equiv_i$ ($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (f; *while* (*init w*) *do* f))
using *1 21 22 62 prop03* **by** *blast*
have *71*: $\vdash$ *if* $_i$ (*init w*) *then* (f; (*while* (*init w*) *do* f)) *else* *empty* $\equiv_i$
($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (f; *while* (*init w*) *do* f))
by *auto*

from *7 71* **show** *?thesis* **using** *itl-prop*(*30*) *prop03* **by** *blast*
qed

lemma *WhileChopEqvIf*:
$\vdash$ (*while* (*init w*) *do f*); $g \equiv_i$ *if*$_i$(*init w*) *then* (*f*; ((*while* (*init w*) *do f*); *g*)) *else g*
proof −
have *1*: $\vdash$ *while* (*init w*) *do f* $\equiv_i$
if$_i$ (*init w*) *then* (*f*; (*while* (*init w*) *do f*)) *else empty*
by (*rule WhileEqvIf*)
hence *2*: $\vdash$ (*while* (*init w*) *do f*); $g \equiv_i$
if$_i$ (*init w*) *then* ((*f*; *while* (*init w*) *do f*); *g*) *else* (*empty* ; *g*)
by (*rule IfChopEqvRule*)
have *3*: $\vdash$ *empty* ; $g \equiv_i g$
by (*rule EmptyChop*)
have *4*: $\vdash$ *if*$_i$ (*init w*) *then* ((*f*; *while* (*init w*) *do f*); *g*) *else* (*empty* ; *g*) $\equiv_i$
if$_i$ (*init w*) *then* ((*f*; *while* (*init w*) *do f*); *g*) *else g*
using *3* **by** (*simp add*: *ifthenelse-d-def*)
have *5*: $\vdash$ ((*f*; *while* (*init w*) *do f*); *g*) $\equiv_i$ (*f*; (*while* (*init w*) *do f*; *g*))
by (*rule ChopAssocB*)
have *6*: $\vdash$ *if*$_i$ (*init w*) *then* ((*f*; *while* (*init w*) *do f*); *g*) *else* $g \equiv_i$
if$_i$ (*init w*) *then* (*f*; ((*while* (*init w*) *do f*); *g*)) *else g*
using *5* **by** (*simp add*: *ifthenelse-d-def*)
from *1 2 4 6* **show** *?thesis* **using** *prop02 prop03* **by** *blast*
qed

lemma *WhileChopEqvIfRule*:
assumes $\vdash$ $f \equiv_i$ (*while* (*init w*) *do g*); *h*
shows $\vdash f \equiv_i$ *if*$_i$ (*init w*) *then* (*g*; *f*) *else h*
proof −
have *1*: $\vdash f \equiv_i$ (*while* (*init w*) *do g*); *h*
using *assms* **by** *auto*
have *2*: $\vdash$ (*while* (*init w*) *do g*); $h \equiv_i$
if$_i$ (*init w*) *then* (*g*; ((*while* (*init w*) *do g*); *h*)) *else h*
by (*rule WhileChopEqvIf*)
have *3*: $\vdash$ (*g*; *f*) $\equiv_i$ (*g*; ((*while* (*init w*) *do g*); *h*))
using *1* **by** (*rule RightChopEqvChop*)
have *4*: $\vdash$ (*g*; ((*while* (*init w*) *do g*); *h*)) $\equiv_i$ (*g*; *f*)
using *3* **by** *auto*
have *5*: $\vdash$ *if*$_i$ (*init w*) *then* (*g*; ((*while* (*init w*) *do g*); *h*)) *else* $h \equiv_i$
if$_i$ (*init w*) *then* (*g*; *f*) *else h*
using *4* **by** (*simp add*: *ifthenelse-d-def*)
from *1 2 5* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *WhileImpFin*:
$\vdash$ *while* (*init w*) *do f* $\supset_i$ *fin* $\neg_i$ (*init w*)
proof −
have *1*: $\vdash$ (*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*) $\supset_i$ *fin* $\neg_i$ (*init w*) **by** *auto*
from *1* **show** *?thesis* **by** (*simp add*: *while-d-def*)
qed

lemma *WhileEqvEmptyOrChopWhile*:
$\vdash$ *while* (*init w*) *do f* $\equiv_i$ ($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (*f* $\wedge_i$ *more*);*while* (*init w*) *do f*)
proof –
have *1*: $\vdash$ (*init w* $\wedge_i$ *f*)* $\equiv_i$ *empty* $\vee_i$ ((*init w* $\wedge_i$ *f*)$\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*
by (*rule ChopstarEqv*)
have *2*: $\vdash$ (*init w* $\wedge_i$ *f*) $\wedge_i$ *more* $\equiv_i$ *init w* $\wedge_i$ (*f* $\wedge_i$ *more*)
by *auto*
hence *3*: $\vdash$ ((*init w* $\wedge_i$ *f*)$\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)* $\equiv_i$ (*init w* $\wedge_i$ *f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*
by (*rule LeftChopEqvChop*)
have *4*: $\vdash$ (*init w* $\wedge_i$ *f*)* $\equiv_i$ *empty* $\vee_i$ (*init w* $\wedge_i$ *f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*
using *1 3* **by** (*meson EmptyImpCS itl-prop*(*31*) *prop30 prop19 prop02 prop14*)
have *5*: $\vdash$ (*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
(*empty* $\wedge_i$ *fin* $\neg_i$ (*init w*)) $\vee_i$
((*init w* $\wedge_i$ *f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*$\wedge_i$ *fin* $\neg_i$ (*init w*))
using *1 prop23* **using** *4* **by** *blast*
have *6*: $\vdash$ *empty* $\wedge_i$ *fin* $\neg_i$ (*init w*)$\equiv_i$ $\neg_i$ (*init w*) $\wedge_i$ *empty*
using *AndFinEqvChopAndEmpty EmptyChop prop03* **by** *blast*
have *7*: $\vdash$ (*init w* $\wedge_i$ *f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)* $\equiv_i$ *init w* $\wedge_i$ (*f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*
by (*rule StateAndChop*)
have *8*: $\vdash$ ((*f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)*) $\wedge_i$ *fin* (*init* $\neg_i$ *w*) $\equiv_i$
(*f* $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* (*init* $\neg_i$ *w*))
by (*rule ChopAndFin*)
have *81*: $\vdash$ *fin* (*init* $\neg_i$ *w*) $\equiv_i$ *fin* $\neg_i$ (*init w*)
using *FinEqvFin Initprop*(*2*) *itl-prop*(*30*) **by** *blast*
have *82*: $\vdash$ (*f* $\wedge_i$ *more*); (*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
(*f* $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*))
using *8 81* **by** *auto*
have *9*: $\vdash$ (*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*) $\equiv_i$
($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$
(*init w* $\wedge_i$ (*f* $\wedge_i$ *more*); ((*init w* $\wedge_i$ *f*)* $\wedge_i$ *fin* $\neg_i$ (*init w*)))
using *5 6 7 82 prop24* **by** *blast*
from *9* **show** *?thesis* **by** (*metis while-d-def*)
qed

lemma *WhileIntro*:
assumes $\vdash$ $\neg_i$ (*init w*) $\wedge_i$ *f* $\supset_i$ *empty*
$\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ (*g* $\wedge_i$ *more*); *f*
shows $\vdash$ *f*$\supset_i$ *while* (*init w*) *do g*
proof –
have *1*: $\vdash$ $\neg_i$ (*init w*) $\wedge_i$ *f* $\supset_i$ *empty*
using *assms* **by** *blast*
have *2*: $\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ (*g* $\wedge_i$ *more*); *f*
using *assms* **by** *blast*
have *3*: $\vdash$ *while* (*init w*) *do g* $\equiv_i$
($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (*g* $\wedge_i$ *more*); *while* (*init w*) *do g*)
by (*rule WhileEqvEmptyOrChopWhile*)
hence *31*: $\vdash$ $\neg_i$ (*while* (*init w*) *do g*) $\equiv_i$
$\neg_i$(($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (*g* $\wedge_i$ *more*); *while* (*init w*) *do g*))
using *itl-prop*(*33*) **by** *blast*

hence *32*: $\vdash f \wedge_i \neg_i$ (*while* (*init w*) *do* g) $\equiv_i$
$f \wedge_i \neg_i$(($\neg_i$(*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ ($g \wedge_i$ *more*); *while* (*init w*) *do* g))
using *prop05* **by** *blast*
have *33*: $\vdash f \wedge_i \neg_i$(($\neg_i$(*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ ($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\equiv_i$
$f \wedge_i \neg_i$($\neg_i$(*init w*) $\wedge_i$ *empty*) $\wedge_i \neg_i$(*init w* $\wedge_i$ ($g \wedge_i$ *more*); *while* (*init w*) *do* g)
by *auto*
have *34*: $\vdash f \wedge_i \neg_i$($\neg_i$(*init w*) $\wedge_i$ *empty*) $\wedge_i \neg_i$(*init w* $\wedge_i$ ($g \wedge_i$ *more*); *while* (*init w*) *do* g) $\equiv_i$
$f \wedge_i$ ((*init w*) $\vee_i$ *more*) $\wedge_i$ ($\neg_i$(*init w*) $\vee_i \neg_i$(($g \wedge_i$ *more*); *while* (*init w*) *do* g))
by *auto*
have *35*: $\vdash f \wedge_i$ ((*init w*) $\vee_i$ *more*) $\wedge_i$ ($\neg_i$(*init w*) $\vee_i \neg_i$(($g \wedge_i$ *more*);*while* (*init w*) *do* g)) $\equiv_i$
($f \wedge_i$ (*init w*) $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\vee_i$
($f \wedge_i$ (*init w*) $\wedge_i \neg_i$(*init w*)) $\vee_i$
($f \wedge_i$ *more* $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\vee_i$
($f \wedge_i$ *more* $\wedge_i \neg_i$(*init w*))
by *auto*
have *36*: $\vdash \neg_i$($f \wedge_i$ (*init w*) $\wedge_i \neg_i$(*init w*))
by *auto*
have *37*: $\vdash \neg_i$($f \wedge_i$ *more* $\wedge_i \neg_i$(*init w*))
using *1* **by** *auto*
have *38*: $\vdash$ ($f \wedge_i$ *more* $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\supset_i$
(($g \wedge_i$ *more*); $f \wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g))
using *1 2* **by** *auto*
have *39*: $\vdash$ ($f \wedge_i$ (*init w*) $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\supset_i$
(($g \wedge_i$ *more*); $f \wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g))
using *2* **by** *auto*
have *40*: $\vdash$ (($f \wedge_i$ (*init w*) $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\vee_i$
($f \wedge_i$ (*init w*) $\wedge_i \neg_i$(*init w*)) $\vee_i$
($f \wedge_i$ *more* $\wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)) $\vee_i$
($f \wedge_i$ *more* $\wedge_i \neg_i$(*init w*))) $\supset_i$
($g \wedge_i$ *more*); $f \wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)
using *39 38 37 38* **by** *auto*
have *4*: $\vdash f \wedge_i \neg_i$ (*while* (*init w*) *do* g) $\supset_i$
($g \wedge_i$ *more*); $f \wedge_i \neg_i$ (($g \wedge_i$ *more*); *while* (*init w*) *do* g)
using *32 33 34 35 40* **by** *auto*
have *5*: $\vdash g \wedge_i$ *more* $\supset_i$ *more*
by *auto*
from *4 5* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *WhileElim*:
assumes $\vdash$ $\neg_i$ (*init w*) $\wedge_i$ *empty* $\supset_i g$
$\vdash$ *init w* $\wedge_i$ ($f \wedge_i$ *more*); $g \supset_i g$
shows $\vdash$ *while* (*init w*) *do* $f \supset_i g$
proof −
have *1*: $\vdash$ *while* (*init w*) *do* $f \equiv_i$
($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ ($f \wedge_i$ *more*); *while* (*init w*) *do* f)
by (*rule WhileEqvEmptyOrChopWhile*)
hence *11*: $\vdash$ (*while* (*init w*) *do* f) $\wedge_i \neg_i g \equiv_i$
(($\neg_i$(*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ ($f \wedge_i$ *more*);*while* (*init w*) *do* f)) $\wedge_i \neg_i g$
using *prop06* **by** *blast*

have *2*: $\vdash \neg_i$ (*init w*) $\wedge_i$ *empty* $\supset_i$ *g*
using *assms* **by** *blast*
hence *21*: $\vdash \neg_i$ *g* $\supset_i$ $\neg_i(\neg_i$ (*init w*) $\wedge_i$ *empty*)
by *auto*
have *22*: $\vdash$ (($\neg_i$ (*init w*) $\wedge_i$ *empty*) $\vee_i$ (*init w* $\wedge_i$ (*f* $\wedge_i$ *more*);*while* (*init w*) *do f*)) $\wedge_i$ $\neg_i$ *g* $\supset_i$
(*init w* $\wedge_i$ (*f* $\wedge_i$ *more*); *while* (*init w*) *do f*)
using *21* **by** *auto*
have *23*: $\vdash$ (*while* (*init w*) *do f*) $\wedge_i$ $\neg_i$ *g* $\supset_i$
(*init w* $\wedge_i$ (*f* $\wedge_i$ *more*); *while* (*init w*) *do f*) $\wedge_i$ $\neg_i$ *g*
using *11 21* **by** *auto*
have *3*: $\vdash$ (*init w*) $\wedge_i$ ((*f* $\wedge_i$ *more*); *g*) $\supset_i$ *g*
using *assms* **by** *blast*
hence *31*: $\vdash \neg_i$ *g* $\supset_i$ $\neg_i$((*init w*) $\wedge_i$ ((*f* $\wedge_i$ *more*); *g*))
using *prop27* **by** *blast*
have *32*: $\vdash$ (*init w* $\wedge_i$ (*f* $\wedge_i$ *more*); *while* (*init w*) *do f*) $\wedge_i$ $\neg_i$ *g* $\supset_i$
(((*f* $\wedge_i$ *more*); (*while* (*init w*) *do f*)) $\wedge_i$ $\neg_i$ ((*f* $\wedge_i$ *more*); *g*)) $\wedge_i$ $\neg_i$*g*
using *31* **by** *auto*
have *4*: $\vdash$ (*while* (*init w*) *do f*) $\wedge_i$ $\neg_i$ *g* $\supset_i$
((*f* $\wedge_i$ *more*); (*while* (*init w*) *do f*)) $\wedge_i$ $\neg_i$ ((*f* $\wedge_i$ *more*); *g*)
using *23 32* **by** *auto*
have *5*: $\vdash$ *f* $\wedge_i$ *more* $\supset_i$ *more*
by *auto*
from *4 5* **show** *?thesis* **using** *ChopContra* **by** *blast*
qed

lemma *BaWhileImpWhile*:
$\vdash$ *ba* (*f* $\supset_i$ *g*) $\supset_i$ (*while* (*init w*) *do f*) $\supset_i$ (*while* (*init w*) *do g*)
proof −
have *1*: $\vdash$ (*f* $\supset_i$ *g*) $\supset_i$ ((*init w* $\wedge_i$ *f*) $\supset_i$ (*init w* $\wedge_i$ *g*))
by *auto*
hence *2*: $\vdash$ *ba* (*f* $\supset_i$ *g*) $\supset_i$ *ba* ((*init w* $\wedge_i$ *f*) $\supset_i$ (*init w* $\wedge_i$ *g*))
by (*rule BaImpBa*)
have *3*: $\vdash$ *ba* ((*init w* $\wedge_i$ *f*) $\supset_i$ (*init w* $\wedge_i$ *g*)) $\supset_i$ ((*init w* $\wedge_i$ *f*)* $\supset_i$ (*init w* $\wedge_i$ *g*)*)
by (*rule BaCSImpCS*)
have *4*: $\vdash$ *ba* (*f* $\supset_i$ *g*) $\supset_i$ ((*init w* $\wedge_i$ *f*)$^\star\wedge_i$ *fin* $\neg_i$ (*init w*) $\supset_i$ (*init w* $\wedge_i$ *g*)$^\star\wedge_i$ *fin* $\neg_i$ (*init w*))
using *2 3* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *while-d-def*)
qed

lemma *WhileImpWhile*:
assumes $\vdash$ *f* $\supset_i$ *g*
shows $\vdash$ (*while* (*init w*) *do f*) $\supset_i$ (*while* (*init w*) *do g*)
proof −
have *1*: $\vdash$ *f* $\supset_i$ *g*
using *assms* **by** *auto*
hence *2*: $\vdash$ *ba* (*f* $\supset_i$ *g*)
by (*rule BaGen*)
have *3*: $\vdash$ *ba* (*f* $\supset_i$ *g*) $\supset_i$ (*while* (*init w*) *do f*) $\supset_i$ (*while* (*init w*) *do g*)
by (*rule BaWhileImpWhile*)
from *2 3* **show** *?thesis* **using** *MP* **by** *blast*

qed

5.9 Properties of Halt

lemma *WnextAndMoreEqvNext*:
$\vdash$ *wnext f* $\wedge_i$ *more* $\equiv_i$ $\bigcirc$ *f*
by *auto*

lemma *BoxStateAndEmptyEqvStateAndEmpty*:
$\vdash \Box(empty \equiv_i (init\ w)) \wedge_i empty \equiv_i (init\ w) \wedge_i empty$
apply *simp-all*
by *auto*

lemma *BoxEmptyEqvIStateqvEmptyAndStateOrNotStateNext*:
$\vdash \Box(empty \equiv_i (init\ w)) \equiv_i (empty \wedge_i init\ w) \vee_i (\neg_i(init\ w) \wedge_i \bigcirc(\Box(empty \equiv_i (init\ w))))$
proof –
have *1*: $\vdash \Box(empty \equiv_i (init\ w)) \equiv_i$
$(\Box(empty \equiv_i (init\ w)) \wedge_i empty) \vee_i (\Box(empty \equiv_i (init\ w)) \wedge_i more)$
by *auto*
have *2*: $\vdash \Box(empty \equiv_i (init\ w)) \wedge_i empty \equiv_i (init\ w) \wedge_i empty$
using *BoxStateAndEmptyEqvStateAndEmpty* **by** *blast*
have *3*: $\vdash \Box(empty \equiv_i (init\ w)) \equiv_i (empty \equiv_i (init\ w)) \wedge_i wnext(\Box(empty \equiv_i (init\ w)))$
using *BoxEqvAndWnextBox* **by** *blast*
hence *4*: $\vdash \Box(empty \equiv_i (init\ w)) \wedge_i more \equiv_i$
$(empty \equiv_i (init\ w)) \wedge_i wnext(\Box(empty \equiv_i (init\ w))) \wedge_i more$
by *auto*
have *5*: $\vdash (empty \equiv_i (init\ w)) \wedge_i more \equiv_i \neg_i(init\ w) \wedge_i more$
by *auto*
have *6*: $\vdash wnext(\Box(empty \equiv_i (init\ w))) \wedge_i more \equiv_i \bigcirc(\Box(empty \equiv_i (init\ w)))$
using *WnextAndMoreEqvNext* **by** *auto*
from *1 2 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *HaltStateEqvIfStateThenEmptyElseNext*:
$\vdash halt(\ init\ w) \equiv_i if_i\ (init\ w)\ then\ empty\ else\ (\bigcirc(\ halt\ (\ init\ w)))$
proof –
have *1*: $\vdash halt(\ init\ w) \equiv_i \Box(empty \equiv_i (init\ w))$
by (*simp add*: *halt-d-def*)
have *2*: $\vdash \Box(empty \equiv_i (init\ w)) \equiv_i$
$(empty \wedge_i init\ w) \vee_i (\neg_i(init\ w) \wedge_i \bigcirc(\Box(empty \equiv_i (init\ w))))$
by (*rule BoxEmptyEqvIStateqvEmptyAndStateOrNotStateNext*)
have *21*: $\vdash (empty \wedge_i init\ w) \vee_i (\neg_i(init\ w) \wedge_i \bigcirc(\Box(empty \equiv_i (init\ w)))) \equiv_i$
$(init\ w \wedge_i empty) \vee_i (\neg_i(init\ w) \wedge_i \bigcirc(\Box(empty \equiv_i (init\ w))))$
by *auto*
have *22*: $\vdash \bigcirc(halt\ (\ init\ w)) \equiv_i \bigcirc(\Box(empty \equiv_i (init\ w)))$
using *NextEqvNext* **using** *1* **by** *blast*
have *3*: $\vdash if_i\ (init\ w)\ then\ empty\ else\ (\bigcirc(\ halt\ (\ init\ w))) \equiv_i$
$(init\ w \wedge_i empty\ ) \vee_i (\neg_i(init\ w) \wedge_i \bigcirc(\ halt\ (\ init\ w)))$
by (*simp add*: *ifthenelse-d-def*)
from *1 2 21 22 3* **show** *?thesis* **by** (*simp add*: *halt-d-def*)

qed

lemma *HaltChopEqv*:
$\vdash$ ((*halt* (*init w*)) ; *f*) $\equiv_i$ (*if* $_i$ (*init w*) *then* (*f*) *else* ($\bigcirc$((*halt* (*init w*)); *f*)))
proof –
have *1*: $\vdash$ *halt*(*init w*) $\equiv_i$
(*if* $_i$ (*init w*) *then* *empty* *else* ($\bigcirc$(*halt* (*init w*))))
by (*rule HaltStateEqvIfStateThenEmptyElseNext*)
hence *2*: $\vdash$ ((*halt*(*init w*));*f*) $\equiv_i$
(*if* $_i$ (*init w*) *then* (*empty*;*f*) *else* ($\bigcirc$(*halt* (*init w*));*f*))
by (*rule IfChopEqvRule*)
have *3*: $\vdash$ *empty* ; *f* $\equiv_i$ *f*
by (*rule EmptyChop*)
have *4*: $\vdash$ ($\bigcirc$ (*halt* (*init w*))); *f* $\equiv_i$ $\bigcirc$(*halt* (*init w*); *f*)
by (*rule NextChop*)
from *2 3 4* **show** *?thesis* **using** *prop07 prop03* **by** *blast*
qed

lemma *AndHaltChopImp*:
$\vdash$ *init w* $\wedge_i$ (*halt* (*init w*); *f*) $\supset_i$ *f*
proof –
have *1*: $\vdash$ *halt* (*init w*); *f* $\equiv_i$ *if* $_i$ (*init w*) *then* *f* *else* ($\bigcirc$(*halt* (*init w*); *f*))
by (*rule HaltChopEqv*)
have *2*: $\vdash$ *init w* $\wedge_i$ *if* $_i$ (*init w*) *then* *f* *else* ($\bigcirc$(*halt* (*init w*); *f*)) $\supset_i$ *f*
by (*simp add*: *ifthenelse-d-def and-d-def*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NotAndHaltChopImpNext*:
$\vdash$ $\neg_i$ (*init w*) $\wedge_i$ (*halt* (*init w*); *f*) $\supset_i$ $\bigcirc$(*halt* (*init w*); *f*)
proof –
have *1*: $\vdash$ *halt* (*init w*); *f* $\equiv_i$ *if* $_i$ (*init w*) *then* *f* *else* ($\bigcirc$(*halt* (*init w*); *f*))
by (*rule HaltChopEqv*)
have *2*: $\vdash$ $\neg_i$ (*init w*) $\wedge_i$ *if* $_i$ (*init w*) *then* *f* *else* ($\bigcirc$(*halt* (*init w*); *f*)) $\supset_i$
$\bigcirc$(*halt* (*init w*); *f*)
by *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NotAndHaltChopImpSkipYields*:
$\vdash$ $\neg_i$ (*init w*) $\wedge_i$ (*halt* (*init w*); *f*) $\supset_i$ *skip* *yields* (*halt* (*init w*); *f*)
proof –
have *1*: $\vdash$ $\neg_i$ (*init w*) $\wedge_i$ (*halt* (*init w*); *f*) $\supset_i$ $\bigcirc$(*halt* (*init w*); *f*)
by (*rule NotAndHaltChopImpNext*)
have *2*: $\vdash$ $\bigcirc$(*halt* (*init w*); *f*) $\supset_i$ *skip* *yields* (*halt* (*init w*); *f*)
by (*rule NextImpSkipYields*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *TrueChopAndEmptyEqvChopAndEmpty*:

$\vdash$ (*true$_i$*;(*f* $\wedge_i$ *empty*)) $\wedge_i$ *g* $\equiv_i$ (*g*;(*f*$\wedge_i$ *empty*))
by *force*

lemma *WprevEqvEmptyOrPrev*:
$\vdash$ *wprev f* $\equiv_i$ *empty* $\vee_i$ *prev f*
by *auto*

lemma *NotChopSkipEqvMoreAndNotChopSkip*:
$\vdash$ ($\neg_i$ *f*);*skip* $\equiv_i$ *more* $\wedge_i$ $\neg_i$(*f*;*skip*)
using *WprevEqvEmptyOrPrev*
by (*metis* (*full-types*) *and-d-def empty-d-def*
itl-prop(*30*) *itl-prop*(*33*) *itl-prop*(*4*) *prev-d-def prop03 prop28 wprev-d-def*)

lemma *HaltChopImpNotHaltChopNot*:
$\vdash$ *halt* (*init w*); *f* $\supset_i$ $\neg_i$ (*halt* (*init w*); $\neg_i$ *f*)
proof −
have *1*: $\vdash$ *halt* (*init w*); *f* $\equiv_i$ *if$_i$* (*init w*) *then f else* ($\bigcirc$(*halt* (*init w*); *f*))
by (*rule HaltChopEqv*)
have *2*: $\vdash$ *if$_i$* (*init w*) *then f else* ($\bigcirc$(*halt* (*init w*); *f*)) $\supset_i$
(((*init w*) $\supset_i$ *f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); *f*))))
by (*rule prop11*)
have *3*: $\vdash$ *halt* (*init w*); $\neg_i$*f* $\equiv_i$
if$_i$ (*init w*) *then* $\neg_i$*f else* ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*))
by (*rule HaltChopEqv*)
have *4*: $\vdash$ *if$_i$* (*init w*) *then* $\neg_i$*f else* ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*)) $\supset_i$
(((*init w*) $\supset_i$ $\neg_i$*f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*))))
by (*rule prop11*)
have *5*: $\vdash$ *halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f* $\supset_i$
(((*init w*) $\supset_i$ *f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); *f*)))) $\wedge_i$
(((*init w*) $\supset_i$ $\neg_i$*f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*))))
using *1 2 3 4* **by** *auto*
have *6*: $\vdash$ (((*init w*) $\supset_i$ *f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); *f*)))) $\wedge_i$
(((*init w*) $\supset_i$ $\neg_i$*f*) $\wedge_i$ ($\neg_i$(*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*)))) $\supset_i$
($\bigcirc$(*halt* (*init w*); *f*)) $\wedge_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*))
by *auto*
have *7*: $\vdash$ *halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f* $\supset_i$
($\bigcirc$(*halt* (*init w*); *f*)) $\wedge_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*))
using *5 6 prop02* **by** *blast*
have *8*: $\vdash$ ($\bigcirc$(*halt* (*init w*); *f*)) $\wedge_i$ ($\bigcirc$(*halt* (*init w*); $\neg_i$*f*)) $\equiv_i$
$\bigcirc$ (*halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f*)
by *auto*
have *9*: $\vdash$ *halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f* $\supset_i$
$\bigcirc$ (*halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f*)
using *7 8 itl-prop*(*31*) *prop02* **by** *blast*
hence *10*: $\vdash$ $\neg_i$(*halt* (*init w*); *f* $\wedge_i$ *halt* (*init w*); $\neg_i$*f*)
using *NextLoop* **by** *blast*
from *10* **show** *?thesis* **by** *auto*
qed

lemma *HaltChopImpHaltYields*:
$\vdash$ *halt* (*init w*); *f* $\supset_i$ (*halt* (*init w*)) *yields f*
proof −
have *1*: $\vdash$ *halt* (*init w*); *f* $\supset_i$ $\neg_i$ (*halt* (*init w*); $\neg_i$ *f*) **by** (*rule HaltChopImpNotHaltChopNot*)
from *1* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

lemma *HaltChopAnd*:
$\vdash$ (*halt* (*init w*)); *f* $\wedge_i$ (*halt* (*init w*)); *g* $\supset_i$ (*halt* (*init w*)); (*f* $\wedge_i$ *g*)
proof −
have *1*: $\vdash$ (*halt* (*init w*)); *g* $\supset_i$ (*halt* (*init w*)) *yields g* **by** (*rule HaltChopImpHaltYields*)
hence *2*: $\vdash$ (*halt* (*init w*)); *f* $\wedge_i$ (*halt* (*init w*)); *g* $\supset_i$
(*halt* (*init w*)); *f* $\wedge_i$ (*halt* (*init w*)) *yields g* **by** *auto*
have *3*: $\vdash$ (*halt* (*init w*)); *f* $\wedge_i$ (*halt* (*init w*)) *yields g*$\supset_i$
(*halt* (*init w*)); (*f* $\wedge_i$ *g*) **by** (*rule ChopAndYieldsImp*)
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *HaltAndChopAndHaltChopImpHaltAndChopAnd*:
$\vdash$ (*halt* (*init w*) $\wedge_i$ *f*); *f1* $\wedge_i$ (*halt* (*init w*); *g*) $\supset_i$ (*halt* (*init w*) $\wedge_i$ *f*); (*f1* $\wedge_i$ *g*)
proof −
have *1*: $\vdash$ *f1* $\supset_i$ $\neg_i$ *g* $\vee_i$ (*f1* $\wedge_i$ *g*)
by *auto*
hence *2*: $\vdash$ (*halt* (*init w*) $\wedge_i$ *f*); *f1* $\supset_i$
(*halt* (*init w*) $\wedge_i$ *f*); $\neg_i$ *g* $\vee_i$ ((*halt* (*init w*) $\wedge_i$ *f*); (*f1* $\wedge_i$ *g*))
by (*rule ChopOrImpRule*)
have *3*: $\vdash$ (*halt* (*init w*) $\wedge_i$ *f*); $\neg_i$ *g* $\supset_i$ *halt* (*init w*); $\neg_i$ *g*
by (*rule AndChopA*)
have *31*: $\vdash$ (*halt* (*init w*) $\wedge_i$ *f*); *f1* $\supset_i$
halt (*init w*); $\neg_i$ *g* $\vee_i$ ((*halt* (*init w*) $\wedge_i$ *f*); (*f1* $\wedge_i$ *g*))
using *23* **by** *auto*
have *4*: $\vdash$ *halt* (*init w*); *g* $\supset_i$ $\neg_i$ (*halt* (*init w*); $\neg_i$ *g*)
by (*rule HaltChopImpNotHaltChopNot*)
hence *41*: $\vdash$ (*halt* (*init w*); $\neg_i$ *g*) $\supset_i$ $\neg_i$(*halt* (*init w*); *g*)
by *auto*
have *42*: $\vdash$ (*halt* (*init w*) $\wedge_i$ *f*); *f1* $\supset_i$
$\neg_i$(*halt* (*init w*); *g*) $\vee_i$ ((*halt* (*init w*) $\wedge_i$ *f*); (*f1* $\wedge_i$ *g*))
using *31 41* **by** *auto*
from *42* **show** *?thesis* **by** *auto*
qed

lemma *HaltImpBoxYields*:
$\vdash$ (*halt* (*init w*)); *f* $\supset_i$ ($\Box\neg_i$ (*init w*)) *yields* ((*halt* (*init w*)); *f*)
proof −
have *1*: $\vdash$ ($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*) $\supset_i$ *di* ($\Box$ $\neg_i$ (*init w*))
by (*rule ChopImpDi*)
have *2*: $\vdash$ $\Box$ $\neg_i$ (*init w*) $\supset_i$ $\neg_i$ (*init w*)
by (*rule BoxElim*)
hence *3*: $\vdash$ *di* ($\Box$ $\neg_i$ (*init w*)) $\supset_i$ *di* $\neg_i$ (*init w*)

by (*rule DiImpDi*)
have *4*: $\vdash$ *di* (*init* $\neg_i$ *w*) $\equiv_i$ (*init* $\neg_i$*w*)
by (*rule DiState*)
have *41*: $\vdash$ (*init* $\neg_i$ *w*) $\equiv_i$ $\neg_i$ (*init w*)
by *auto*
have *42*: $\vdash$ *di* $\neg_i$ (*init w*) $\equiv_i$ $\neg_i$(*init w*)
using *4 41* **by** *auto*
have *5*: $\vdash$ (($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*)) $\supset_i$ $\neg_i$ (*init w*)
using *1 2 42* **using** *3 itl-prop*(*31*) *prop02* **by** *blast*
hence *51*: $\vdash$ (*halt* (*init w*); *f*) $\wedge_i$ (($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*)) $\supset_i$
(*halt* (*init w*); *f*) $\wedge_i$ $\neg_i$ (*init w*)
using *prop12* **by** *blast*
have *6*: $\vdash$ *halt* (*init w*); *f* $\equiv_i$ *if* $_i$ (*init w*) *then f else* ($\bigcirc$(*halt* (*init w*); *f*))
by (*rule HaltChopEqv*)
hence *61*: $\vdash$ *halt* (*init w*); *f* $\wedge_i$ $\neg_i$ (*init w*) $\equiv_i$
(*if* $_i$ (*init w*) *then f else* ($\bigcirc$(*halt* (*init w*); *f*))) $\wedge_i$ $\neg_i$ (*init w*)
using *6* **by** *auto*
have *62*: $\vdash$ (*if* $_i$ (*init w*) *then f else* ($\bigcirc$(*halt* (*init w*); *f*))) $\wedge_i$
$\neg_i$ (*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); *f*))
by *auto*
have *63*: $\vdash$ *halt* (*init w*); *f* $\wedge_i$ $\neg_i$ (*init w*) $\supset_i$ ($\bigcirc$(*halt* (*init w*); *f*))
using *61 62* **by** *auto*
have *7*: $\vdash$ (*halt* (*init w*); *f*) $\wedge_i$ ($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*) $\supset_i$
$\bigcirc$((*halt* (*init w*)); *f*)
using *51 63* **using** *prop02* **by** *blast*
have *8*: $\vdash$ $\Box$ $\neg_i$ (*init w*) $\supset_i$ *empty* $\vee_i$ $\bigcirc$($\Box\neg_i$(*init w*))
by *auto*
hence *9*: $\vdash$ (($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*)) $\supset_i$
$\neg_i$ (*halt* (*init w*); *f*) $\vee_i$ $\bigcirc$(($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
by (*rule EmptyOrNextChopImpRule*)
hence *10*: $\vdash$ ((*halt* (*init w*)); *f*) $\wedge_i$ ($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*) $\supset_i$
$\bigcirc$(($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
using *prop13* **by** *blast*
have *11*: $\vdash$ (*halt* (*init w*)); *f* $\wedge_i$ ($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*) $\supset_i$
$\bigcirc$((*halt* (*init w*)); *f*) $\wedge_i$ $\bigcirc$(($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
using *7 10* **by** *auto*
have *12*: $\vdash$ $\bigcirc$((*halt* (*init w*)); *f*) $\wedge_i$ $\bigcirc$(($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
$\supset_i$ $\bigcirc$(((*halt* (*init w*)); *f*) $\wedge_i$ (($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*)))
by *auto*
have *13*: $\vdash$ (*halt* (*init w*)); *f* $\wedge_i$ ($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*) $\supset_i$
$\bigcirc$(((*halt* (*init w*)); *f*) $\wedge_i$ (($\Box\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*)))
using *11 12* **by** *auto*
hence *14*: $\vdash$ $\neg_i$ ((*halt* (*init w*)); *f* $\wedge_i$ ($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
using *NextLoop* **by** *blast*
hence *15*: $\vdash$ (*halt* (*init w*)); *f* $\supset_i$ $\neg_i$ (($\Box$ $\neg_i$ (*init w*)); $\neg_i$ (*halt* (*init w*); *f*))
by *auto*
from *15* **show** *?thesis* **by** (*simp add*: *yields-d-def*)
qed

5.10 Properties of Groups of chops

lemma *NestedChopImpChop*:
assumes $\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ *g*; (*init w1* $\wedge_i$ *f1*)
$\vdash$ *init w1* $\wedge_i$ *f1* $\supset_i$ *g1*; (*init w2* $\wedge_i$ *f2*)
shows $\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ *g*; (*g1*; (*init w2* $\wedge_i$ *f2*))
proof −
have *1*: $\vdash$ *init w* $\wedge_i$ *f* $\supset_i$ *g*; (*init w1* $\wedge_i$ *f1*) **using** *assms*(*1*) **by** *auto*
have *2*: $\vdash$ *init w1* $\wedge_i$ *f1* $\supset_i$ *g1*; (*init w2* $\wedge_i$ *f2*) **using** *assms*(*2*) **by** *auto*
hence *3*: $\vdash$ *g*; (*init w1* $\wedge_i$ *f1*) $\supset_i$ *g*; (*g1*; (*init w2* $\wedge_i$ *f2*)) **by** (*rule RightChopImpChop*)
from *1 3* **show** *?thesis* **by** *auto*
qed

5.11 Properties of Time Reversal

lemma *RNot*:
$\vdash (\neg_i f)^r \equiv_i \neg_i f^r$
by *simp*

lemma *RRNot*:
$\vdash (\neg_i(f^r))^r \equiv_i \neg_i f$
by (*metis EqvReverseReverse not-d-def rev-d.simps*(*1*) *rev-d.simps*(*3*))

lemma *RTrue*:
$\vdash (true_i)^r \equiv_i true_i$
by (*simp add*: *not-d-def true-d-def*)

lemma *ROr*:
$\vdash (f \vee_i g)^r \equiv_i f^r \vee_i g^r$
by (*simp add*: *not-d-def or-d-def*)

lemma *RROr*:
$\vdash (f^r \vee_i g^r)^r \equiv_i f \vee_i g$
proof −
have *1*: $\vdash (f^r \vee_i g^r)^r \equiv_i (f^r)^r \vee_i (g^r)^r$ **using** *ROr* **by** *blast*
have *2*: $\vdash (f^r)^r \vee_i (g^r)^r \equiv_i f \vee_i g$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RAnd*:
$\vdash (f \wedge_i g)^r \equiv_i f^r \wedge_i g^r$
by (*simp add*: *and-d-def not-d-def or-d-def*)

lemma *RRAnd*:
$\vdash (f^r \wedge_i g^r)^r \equiv_i f \wedge_i g$
proof −
have *1*: $\vdash (f^r \wedge_i g^r)^r \equiv_i (f^r)^r \wedge_i (g^r)^r$ **using** *RAnd* **by** *blast*
have *2*: $\vdash (f^r)^r \wedge_i (g^r)^r \equiv_i f \wedge_i g$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *REqvRule*:
assumes $\vdash f \equiv_i g$
shows $\vdash f^r \equiv_i g^r$
using *assms*
by (*metis ReverseEqv itl-prop*(*31*) *rev-d.simps*(*3*))

lemma *RImpRule*:
assumes $\vdash f \supset_i g$
shows $\vdash f^r \supset_i g^r$
using *assms*
by (*metis ReverseEqv rev-d.simps*(*3*))

lemma *RNextEqvPrev*:
$\vdash (\bigcirc f)^r \equiv_i prev\ (f^r)$
by (*simp add*: *next-d-def prev-d-def*)

lemma *RRNextEqvPrev*:
$\vdash (\bigcirc (f^r))^r \equiv_i prev\ (f)$
proof −
have *1*: $\vdash (\bigcirc (f^r))^r \equiv_i prev\ ((f^r)^r)$ **using** *RNextEqvPrev* **by** *blast*
have *2*: $\vdash prev\ ((f^r)^r) \equiv_i prev\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RWNextEqvWPrev*:
$\vdash (wnext\ f)^r \equiv_i wprev(f^r)$
by (*simp add*: *next-d-def not-d-def prev-d-def wnext-d-def wprev-d-def*)

lemma *RRWNextEqvWPrev*:
$\vdash (wnext\ (f^r))^r \equiv_i wprev(f)$
proof −
have *1*: $\vdash (wnext\ (f^r))^r \equiv_i wprev\ ((f^r)^r)$ **using** *RWNextEqvWPrev* **by** *blast*
have *2*: $\vdash wprev\ ((f^r)^r) \equiv_i wprev\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RPrevEqvNext*:
$\vdash (prev\ f)^r \equiv_i \bigcirc (f^r)$
by (*simp add*: *next-d-def prev-d-def*)

lemma *RRPrevEqvNext*:
$\vdash (prev\ (f^r))^r \equiv_i \bigcirc (f)$
proof −
have *1*: $\vdash (prev\ (f^r))^r \equiv_i \bigcirc ((f^r)^r)$ **using** *RPrevEqvNext* **by** *blast*
have *2*: $\vdash \bigcirc ((f^r)^r) \equiv_i \bigcirc f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RWPrevEqvWNext*:
$\vdash (wprev\ f)^r \equiv_i wnext(f^r)$

by (*simp add*: *next-d-def not-d-def prev-d-def wnext-d-def wprev-d-def*)

lemma *RRWPrevEqvWNext*:
$\vdash (wprev\ (f^r))^r \equiv_i wnext(f)$
proof −
have *1*: $\vdash (wprev\ (f^r))^r \equiv_i wnext\ ((f^r)^r)$ **using** *RWPrevEqvWNext* **by** *blast*
have *2*: $\vdash wnext\ ((f^r)^r) \equiv_i wnext\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RDiamondEqvDi*:
$\vdash (\Diamond f)^r \equiv_i di\ (f^r)$
by (*metis RTrue RightChopEqvChop di-d-def rev-d.simps*(*5*) *sometimes-d-def*)

lemma *RRDiamondEqvDi*:
$\vdash (\Diamond(f^r))^r \equiv_i di\ (f)$
proof −
have *1*: $\vdash (\Diamond\ (f^r))^r \equiv_i di\ ((f^r)^r)$ **using** *RDiamondEqvDi* **by** *blast*
have *2*: $\vdash di\ ((f^r)^r) \equiv_i di\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RBoxEqvBi*:
$\vdash (\Box\ f)^r \equiv_i bi\ (f^r)$
using *RDiamondEqvDi*
by (*simp add*: *always-d-def not-d-def sometimes-d-def true-d-def*)

lemma *RRBoxEqvBi*:
$\vdash (\Box\ (f^r))^r \equiv_i bi\ (f)$
proof −
have *1*: $\vdash (\Box\ (f^r))^r \equiv_i bi\ ((f^r)^r)$ **using** *RBoxEqvBi* **by** *blast*
have *2*: $\vdash bi\ ((f^r)^r) \equiv_i bi\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RDiEqvDiamond*:
$\vdash (di\ f)^r \equiv_i \Diamond\ (f^r)$
by (*metis RTrue LeftChopEqvChop di-d-def rev-d.simps*(*5*) *sometimes-d-def*)

lemma *RRDiEqvDiamond*:
$\vdash (di\ (f^r))^r \equiv_i \Diamond\ (f)$
proof −
have *1*: $\vdash (di\ (f^r))^r \equiv_i \Diamond\ ((f^r)^r)$ **using** *RDiEqvDiamond* **by** *blast*
have *2*: $\vdash \Diamond\ ((f^r)^r) \equiv_i \Diamond\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RBiEqvBox*:
$\vdash (bi\ f)^r \equiv_i \Box\ (f^r)$
using *RDiEqvDiamond*

by (*simp add*: *bi-d-def di-d-def not-d-def true-d-def*)

lemma *RRBiEqvBox*:
$\vdash (bi\ (f^r))^r \equiv_i \Box\ (f)$
proof –
have *1*: $\vdash (bi\ (f^r))^r \equiv_i \Box\ ((f^r)^r)$ **using** *RBiEqvBox* **by** *blast*
have *2*: $\vdash \Box\ ((f^r)^r) \equiv_i \Box\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RDaEqvDa*:
$\vdash (da\ f)^r \equiv_i da(f^r)$
by (*metis ChopAssoc da-d-def itl-prop*(*30*) *not-d-def rev-d.simps*(*1*)
rev-d.simps(*3*) *rev-d.simps*(*5*) *true-d-def*)

lemma *RRDaEqvDa*:
$\vdash (da\ (f^r))^r \equiv_i da(f)$
proof –
have *1*: $\vdash (da\ (f^r))^r \equiv_i da\ ((f^r)^r)$ **using** *RDaEqvDa* **by** *blast*
have *2*: $\vdash da\ ((f^r)^r) \equiv_i da\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RBaEqvBa*:
$\vdash (ba\ f)^r \equiv_i ba(f^r)$
using *RDaEqvDa*
by (*metis ba-d-def itl-prop*(*33*) *not-d-def rev-d.simps*(*1*) *rev-d.simps*(*3*))

lemma *RRBaEqvBa*:
$\vdash (ba\ (f^r))^r \equiv_i ba(f)$
proof –
have *1*: $\vdash (ba\ (f^r))^r \equiv_i ba\ ((f^r)^r)$ **using** *RBaEqvBa* **by** *blast*
have *2*: $\vdash ba\ ((f^r)^r) \equiv_i ba\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *ChopCsImpCSChop*:
$\vdash f;f^\star \supset_i f^\star;f$
by (*meson CSChopEqvChopOrRule CSChopEqvOrChopPlusChop ChopAssocB*
ChopPlusElimWithoutMore EmptyYields prop19 prop21 prop28)

lemma *CSChopImpChopCS*:
$\vdash f^\star;f \supset_i f;f^\star$
proof –
have *1*: $\vdash (f^r);(f^r)^\star \supset_i (f^r)^\star;(f^r)$ **using** *ChopCsImpCSChop* **by** *blast*
hence *2*: $\vdash ((f^r);(f^r)^\star \supset_i (f^r)^\star;(f^r)\ )^r$ **using** *ReverseEqv* **by** *blast*
have *3*: $\vdash ((f^r);(f^r)^\star \supset_i (f^r)^\star;(f^r)\ )^r \equiv_i (\ ((f^r);(f^r)^\star)^r \supset_i ((f^r)^\star;(f^r))^r)$ **by** *simp*
have *4*: $\vdash ((f^r);(f^r)^\star)^r \equiv_i ((f^r)^\star\ )^r;\ (f^r)^r$ **by** *simp*
have *5*: $\vdash ((f^r)^\star\ )^r;\ (f^r)^r \equiv_i ((f^r)^r)^\star;(f^r)^r$ **by** *simp*
have *6*: $\vdash (f^r)^r \equiv_i f$ **using** *EqvReverseReverse itl-prop*(*30*) **by** *blast*

have *7*: $\vdash ((f^r)^r)^\star;(f^r)^r \equiv_i f^\star;f$ **using** *6 CSEqvCS ChopEqvChop* **by** *blast*
have *8*: $\vdash ((f^r);(f^r)^\star)^r \equiv_i f^\star;f$ **using** *7 5* **by** *auto*
have *9*: $\vdash ((f^r)^\star;(f^r))^r \equiv_i (f^r)^r;((f^r)^\star)^r$ **by** *simp*
have *10*: $\vdash (f^r)^r;((f^r)^\star)^r \equiv_i (f^r)^r;\ ((f^r)^r)^\star$ **by** *simp*
have *11*: $\vdash (f^r)^r;\ ((f^r)^r)^\star \equiv_i f;f^\star$ **using** *6 ChopPlusEqvChopPlus* **by** *blast*
have *12*: $\vdash ((f^r);(f^r)^\star)^r \equiv_i f;f^\star$ **using** *9 10 11*
by (*metis 2 ChopCsImpCSChop itl-prop*(*31*) *prop03 rev-d.simps*(*3*) *rev-d.simps*(*5*) *rev-d.simps*(*6*))
from *2 3 8 12* **show** *?thesis* **by** *auto*
qed

lemma *CSChopEqvChopCS*:
$\vdash f;f^\star \equiv_i f^\star;f$
using *ChopCsImpCSChop CSChopImpChopCS* **using** *itl-prop*(*31*) **by** *blast*

lemma *TrueChopSkipEqvSkipChopTrue*:
$\vdash$ *true$_i$;skip* $\equiv_i$ *skip;true$_i$*
proof −
have *1*: $\vdash skip;skip^\star \equiv_i skip^\star;skip$ **using** *CSChopEqvChopCS* **by** *blast*
have *2*: $\vdash skip^\star \equiv_i true_i$ **using** *CSSkip* **by** *simp*
have *3*: $\vdash skip;skip^\star \equiv_i skip;true_i$ **using** *2* **using** *RightChopEqvChop* **by** *blast*
have *4*: $\vdash skip^\star;skip \equiv_i true_i;skip$ **using** *2* **using** *LeftChopEqvChop* **by** *blast*
from *1 3 4* **show** *?thesis* **using** *itl-prop*(*30*) *prop03* **by** *blast*
qed

lemma *RMoreEqvMore*:
$\vdash more^r \equiv_i more$
by (*metis TrueChopSkipEqvSkipChopTrue more-d-def next-d-def not-d-def rev-d.simps*(*1*)
rev-d.simps(*3*) *rev-d.simps*(*4*) *rev-d.simps*(*5*) *true-d-def*)

lemma *REmptyEqvEmpty*:
$\vdash empty^r \equiv_i empty$
by (*metis RMoreEqvMore empty-d-def not-d-def prop01 rev-d.simps*(*1*) *rev-d.simps*(*3*))

lemma *RInitEqvFin*:
$\vdash (init\ f)^r \equiv_i fin(f^r)$
proof −
have *1*: $\vdash (init\ f)^r \equiv_i ((f \wedge_i empty);true_i)^r$
by (*metis AndChopCommute REqvRule init-d-def*)
have *2*: $\vdash ((f \wedge_i empty);true_i)^r \equiv_i (true_i;(f \wedge_i empty)^r)$
using *RTrue* **by** *auto*
have *3*: $\vdash true_i;(f \wedge_i empty)^r \equiv_i true_i;(f^r \wedge_i empty)$
by (*meson ChopEqvChop RAnd REmptyEqvEmpty RTrue itl-prop*(*30*) *prop05 prop21*)
have *4*: $\vdash true_i;(f^r \wedge_i empty) \equiv_i fin(f^r)$
using *FinEqvTrueChopAndEmpty itl-prop*(*30*) **by** *blast*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *RRInitEqvFin*:
$\vdash (init\ (f^r))^r \equiv_i fin(f)$
proof −

have *1*: $\vdash$ $(init\ (f^r))^r \equiv_i fin\ ((f^r)^r)$ **using** *RInitEqvFin* **by** *blast*
have *2*: $\vdash$ $fin\ ((f^r)^r) \equiv_i fin\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RFinEqvInit*:
$\vdash (fin\ f)^r \equiv_i init\ (f^r)$
proof −
have *1*: $\vdash fin\ f \equiv_i true_i;(f \wedge_i empty)$
  using *FinEqvTrueChopAndEmpty* **by** *auto*
have *2*: $\vdash (fin\ f)^r \equiv_i (true_i;(f \wedge_i empty))^r$
  using *1 REqvRule* **by** *blast*
have *3*: $\vdash (true_i;(f \wedge_i empty))^r \equiv_i (f \wedge_i empty)^r;true_i$
  using *RTrue* **by** *auto*
have *4*: $\vdash (f \wedge_i empty)^r;true_i \equiv_i (f^r \wedge_i empty);true_i$
  using *LeftChopEqvChop RAnd REmptyEqvEmpty prop03 prop05* **by** *blast*
have *5*: $\vdash (f^r \wedge_i empty);true_i \equiv_i init(f^r)$
  by *auto*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *RRFinEqvInit*:
$\vdash (fin\ (f^r))^r \equiv_i init\ (f)$
proof −
have *1*: $\vdash (fin\ (f^r))^r \equiv_i init\ ((f^r)^r)$ **using** *RFinEqvInit* **by** *blast*
have *2*: $\vdash init\ ((f^r)^r) \equiv_i init\ f$ **using** *EqvReverseReverse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *NextDiamondEqvDiamondNext*:
$\vdash \bigcirc(\Diamond\ f) \equiv_i \Diamond(\bigcirc f)$
proof −
have *1*: $\vdash true_i;skip \equiv_i skip;true_i$ **by** (*rule TrueChopSkipEqvSkipChopTrue*)
hence *2*: $\vdash (true_i;skip);f \equiv_i (skip;true_i);f$ **using** *LeftChopEqvChop* **by** *blast*
have *3*: $\vdash (true_i;skip);f \equiv_i true_i;(skip;f)$ **using** *ChopAssoc itl-prop*(*30*) **by** *blast*
have *4*: $\vdash (skip;true_i);f \equiv_i skip;(true_i;f)$ **using** *ChopAssoc itl-prop*(*30*) **by** *blast*
from *2 3 4* **show** *?thesis* **by** (*simp add*: *next-d-def*)
qed

lemma *WeakNextBoxInduct*:
assumes $\vdash wnext\ (\Box\ f) \supset_i f$
shows $\vdash f$
proof −
have *1*: $\vdash wnext\ (\Box\ f) \supset_i f$ **using** *assms* **by** *blast*
hence *2*: $\vdash \neg_i f \supset_i \neg_i (wnext\ (\Box\ f))$ **using** *prop27* **by** *blast*
hence *3*: $\vdash \neg_i f \supset_i \bigcirc (\neg_i (\Box\ f))$ **by** *simp*
have *4*: $\vdash \neg_i (\Box\ f) \equiv_i (\Diamond\ \neg_i f)$ **by** *auto*
hence *5*: $\vdash \bigcirc(\neg_i (\Box\ f)) \equiv_i \bigcirc (\Diamond\ \neg_i f)$ **by** *auto*
have *6*: $\vdash \neg_i f \supset_i \bigcirc (\Diamond\ \neg_i f)$ **using** *3 5* **by** *auto*
have *7*: $\vdash \bigcirc (\Diamond\ \neg_i f) \equiv_i \Diamond(\bigcirc \neg_i f)$ **using** *NextDiamondEqvDiamondNext* **by** *blast*

have *8*: $\vdash \neg_i\ f\ \supset_i\ \ \Diamond(\ \bigcirc \neg_i\ f)$ **using** *6 7* **by** *auto*
have *9*: $\vdash \Diamond(\neg_i\ f)\ \supset_i\ \ \Diamond(\Diamond(\ \bigcirc \neg_i\ f))$ **using** *8 DiamondImpDiamond* **by** *blast*
have *10*: $\vdash \Diamond(\Diamond(\ \bigcirc \neg_i\ f)) \equiv_i \Diamond(\ \bigcirc \neg_i\ f)$ **using** *DiamondDiamondEqvDiamond* **by** *blast*
have *11*: $\vdash \Diamond(\neg_i\ f)\ \supset_i \Diamond(\ \bigcirc \neg_i\ f)$ **using** *9 10* **by** *auto*
have *12*: $\vdash \Diamond(\neg_i\ f)\ \supset_i \bigcirc\ (\Diamond\ \neg_i\ f)$ **using** *7 11* **by** *auto*
hence *13*: $\vdash \neg_i(\ \Diamond(\neg_i\ f))$ **using** *NextLoop* **by** *blast*
hence *14*: $\vdash \Box f$ **by** (*simp add*: *always-d-def*)
have *15*: $\vdash \Box f \supset_i f$ **using** *BoxElim* **by** *blast*
from *14 15* **show** *?thesis* **using** *MP* **by** *blast*
qed

end

theory *First*
imports
Theorems
begin

6 The First Occurrence Operator in ITL

Runtime verification (RV) has gained significant interest in recent years. The behaviour of a program can be verified in real time by analysing its evolving trace. This approach has two significant benefits over static verification techniques such as model checking. Firstly, it is only necessary to verify actual execution paths rather than all possible paths. Secondly, it is possible to react at runtime should the program diverge from its specified behaviour. RV does not replace traditional verification techniques but it does provide an extra layer of security.

Linear Temporal Logic (LTL) is a popular formalism for writing specifications from which RV monitors can be derived automatically. By contrast, Interval Temporal Logic (ITL) has not been as widely represented in this field despite being more expressive and compositional. The principal issue is efficiency. ITL uses non-deterministic operators to construct sequential and iterative specifications (chop and chop-star, respectively) and these introduce combinatorial complexity. Approaches to mitigate this include using a deterministic subset of ITL or adapting the semantics to include a deterministic chop operator. This thesis proposes an alternative approach, wholly within existing ITL, and based upon a new, derived operator called “first occurrence”.

A theory of first occurrence is developed and used to derive an algebra of RV monitors.

6.1 Definitions

6.1.1 Definitions Strict Initial and Final

definition *bs-d* ((*bs* -) [*88*] *87*)
where
bs f $\equiv$ (*empty* $\vee_i$ ((*bi f*) ; *skip*))

definition *bt-d* ((*bt* -) [*88*] *87*)
where
bt f $\equiv$ (*empty* $\vee_i$ (*skip*;($\Box$ *f*)))

definition *ds-d* ((*ds* -) [*88*] *87*)
where
  $ds\ f \equiv \neg_i\ (bs\ (\neg_i\ f))$

definition *dt-d* ((*dt* -) [*88*] *87*)
where
  $dt\ f \equiv \neg_i\ (bt\ (\neg_i\ f))$

6.1.2 Definition First and Last Operators

definition *first-d* (($\rhd$ -) [*88*] *87*)
where
  $\rhd\ f \equiv\ (f \wedge_i\ (bs\ (\neg_i\ f)))$

definition *last-d* (($\lhd$ -) [*88*] *87*)
where
  $\lhd\ f \equiv\ (f \wedge_i\ (bt\ (\neg_i\ f)))$

6.2 First and Time Reversal

lemma *BsEqvRule*:
 assumes $\vdash f \equiv_i g$
 shows $\vdash bs\ f \equiv_i bs\ g$
proof −
 have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
 hence *2*: $\vdash bi(f) \equiv_i bi(g)$ **by** *simp*
 hence *3*: $\vdash bi(f);skip\ \equiv_i bi(g);skip$ **by** *auto*
 hence *4*: $\vdash empty \vee_i bi(f);skip\ \equiv_i empty \vee_i bi(g);skip$ **by** *auto*
 hence *5*: $\vdash bs(f) \equiv_i bs(g)$ **by** (*simp add*: *bs-d-def*)
 from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *BtEqvRule*:
 assumes $\vdash f \equiv_i g$
 shows $\vdash bt\ f \equiv_i bt\ g$
proof −
 have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
 hence *2*: $\vdash \Box(f) \equiv_i \Box(g)$ **by** *simp*
 hence *3*: $\vdash skip;\Box(f)\ \equiv_i skip;\Box(g)$ **using** *RightChopEqvChop* **by** *blast*
 hence *4*: $\vdash empty \vee_i skip;\Box(f)\ \equiv_i empty \vee_i skip;\Box(g)$ **by** *auto*
 hence *5*: $\vdash bt(f) \equiv_i bt(g)$ **by** (*simp add*: *bt-d-def*)
 from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *FstEqvRule*:
 assumes $\vdash f \equiv_i g$
 shows $\vdash \rhd f \equiv_i \rhd g$
proof −
 have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*

hence *2*: $\vdash \neg_i\ f \equiv_i \neg_i\ g$ **by** *auto*
hence *3*: $\vdash bs(\neg_i\ f) \equiv_i bs(\neg_i\ g)$ **by** (*simp add*: *bs-d-def*)
hence *4*: $\vdash f \wedge_i bs(\neg_i\ f) \equiv_i g \wedge_i bs(\neg_i\ g)$ **using** *1* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*:*first-d-def*)
qed

lemma *LstEqvRule*:
assumes $\vdash f \equiv_i g$
shows $\vdash \lhd f \equiv_i \lhd g$
proof −
have *1*: $\vdash f \equiv_i g$ **using** *assms* **by** *auto*
hence *2*: $\vdash \neg_i\ f \equiv_i \neg_i\ g$ **by** *auto*
hence *3*: $\vdash bt(\neg_i\ f) \equiv_i bt(\neg_i\ g)$ **by** (*simp add*: *bt-d-def*)
hence *4*: $\vdash f \wedge_i bt(\neg_i\ f) \equiv_i g \wedge_i bt(\neg_i\ g)$ **using** *1* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*:*last-d-def*)
qed

lemma *RBsEqvBt*:
$\vdash (bs\ f)^r \equiv_i (bt\ (f^r))$
proof −
have *1*: $\vdash (bs\ f)^r \equiv_i (empty \vee_i ((bi\ f)\ ;\ skip))^r$
by (*simp add*: *bs-d-def*)
have *2*: $\vdash (empty \vee_i ((bi\ f)\ ;\ skip))^r \equiv_i (empty^r \vee_i ((bi\ f)\ ;\ skip)^r)$
using *ROr* **by** *blast*
have *3*: $\vdash (empty^r \vee_i ((bi\ f)\ ;\ skip)^r) \equiv_i (empty \vee_i (skip^r;(bi\ f)^r))$
using *REmptyEqvEmpty* **by** *auto*
have *4*: $\vdash (empty \vee_i (skip^r;(bi\ f)^r)) \equiv_i (empty \vee_i (skip;\Box\ (f^r)))$
by (*metis* (*no-types*, *lifting*) *NextEqvNext RBiEqvBox REmptyEqvEmpty next-d-def or-d-def prop01 prop21 prop39 rev-d*.*simps*(*4*))
have *5*: $\vdash (empty \vee_i (skip;\Box\ (f^r))) \equiv_i (bt\ (f^r))$
by (*simp add*: *bt-d-def*)
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *RRBsEqvBt*:
$\vdash (bs\ (f^r))^r \equiv_i (bt\ (f))$
proof −
have *1*: $\vdash (bs\ (f^r))^r \equiv_i bt\ ((f^r)^r)$ **using** *RBsEqvBt* **by** *blast*
have *2*: $\vdash bt\ ((f^r)^r) \equiv_i bt\ f$ **using** *EqvReverseReverse* **using** *BtEqvRule* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RBtEqvBs*:
$\vdash (bt\ f)^r \equiv_i (bs\ (f^r))$
proof −
have *1*: $\vdash (bt\ f)^r \equiv_i (empty \vee_i (skip;\Box\ f))^r$
by (*simp add*: *bt-d-def*)
have *2*: $\vdash (empty \vee_i (skip;\Box\ f))^r \equiv_i (empty^r \vee_i (skip;\Box\ f)^r)$
using *ROr* **by** *blast*
have *3*: $\vdash (empty^r \vee_i (skip;\Box\ f)^r) \equiv_i (empty \vee_i (\Box\ f)^r;skip^r)$

using *REmptyEqvEmpty* **by** *auto*
have *4*: $\vdash$ (*empty* $\vee_i$ ($\Box$ *f*)^r;*skip*^{r}) $\equiv_i$ (*empty* $\vee_i$ (*bi* (f^r));*skip*)
by (*metis* (*no-types*, *lifting*) *LeftChopEqvChop RBoxEqvBi REmptyEqvEmpty or-d-def prop01 prop21 prop39 rev-d.simps*(*4*))
have *5*: $\vdash$ (*empty* $\vee_i$ (*bi* (f^r));*skip*) $\equiv_i$ (*bs* (f^r))
by (*simp add*: *bs-d-def*)
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *RRBtEqvBs*:
$\vdash$ (*bt* (f^r))^r $\equiv_i$ (*bs* (*f*))
proof −
have *1*: $\vdash$ (*bt* (f^r))^r $\equiv_i$ *bs* ((f^r)^r) **using** *RBtEqvBs* **by** *blast*
have *2*: $\vdash$ *bs* ((f^r)^r) $\equiv_i$ *bs f* **using** *EqvReverseReverse* **using** *BsEqvRule* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RFirstEqvLast*:
$\vdash$ ($\rhd$ *f*)^r $\equiv_i$ ($\lhd$ (f^r))
proof −
have *1*: $\vdash$ ($\rhd$ *f*)^r $\equiv_i$ (*f* $\wedge_i$ *bs*($\neg_i$ *f*))^r **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ (*f* $\wedge_i$ *bs*($\neg_i$ *f*))^r $\equiv_i$ (f^r $\wedge_i$ (*bs* ($\neg_i$ *f*))^r) **using** *RAnd* **by** *blast*
have *3*: $\vdash$ (f^r $\wedge_i$ (*bs* ($\neg_i$ *f*))^r) $\equiv_i$ (f^r $\wedge_i$ *bt* (($\neg_i$ *f*)^r)) **using** *RBsEqvBt prop05* **by** *blast*
have *4*: $\vdash$ (f^r $\wedge_i$ *bt* (($\neg_i$ *f*)^r)) $\equiv_i$ (f^r $\wedge_i$ *bt* ($\neg_i$(f^r))) **by** (*simp add*: *not-d-def*)
have *5*: $\vdash$ (f^r $\wedge_i$ *bt* ($\neg_i$(f^r))) $\equiv_i$ ($\lhd$ (f^r)) **by** (*simp add*: *last-d-def*)
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *RRFirstEqvLast*:
$\vdash$ ($\rhd$ (f^r))^r $\equiv_i$ ($\lhd$ (*f*))
proof −
have *1*: $\vdash$ ($\rhd$ (f^r))^r $\equiv_i$ $\lhd$ ((f^r)^r) **using** *RFirstEqvLast* **by** *blast*
have *2*: $\vdash$ $\lhd$ ((f^r)^r) $\equiv_i$ $\lhd$ *f* **using** *EqvReverseReverse* **using** *LstEqvRule* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *RLastEqvFirst*:
$\vdash$ ($\lhd$ *f*)^r $\equiv_i$ ($\rhd$ (f^r))
proof −
have *1*: $\vdash$ ($\lhd$ *f*)^r $\equiv_i$ (*f* $\wedge_i$ *bt*($\neg_i$ *f*))^r **by** (*simp add*: *last-d-def*)
have *2*: $\vdash$ (*f* $\wedge_i$ *bt*($\neg_i$ *f*))^r $\equiv_i$ (f^r $\wedge_i$ (*bt* ($\neg_i$*f*))^r) **using** *RAnd* **by** *blast*
have *3*: $\vdash$ (f^r $\wedge_i$ (*bt* ($\neg_i$*f*))^r) $\equiv_i$ (f^r $\wedge_i$ *bs* (($\neg_i$*f*)^r)) **using** *RBtEqvBs prop05* **by** *blast*
have *4*: $\vdash$ (f^r $\wedge_i$ *bs* (($\neg_i$*f*)^r)) $\equiv_i$ (f^r $\wedge_i$ *bs*($\neg_i$(f^r))) **by** (*simp add*: *not-d-def*)
have *5*: $\vdash$ (f^r $\wedge_i$ *bs*($\neg_i$(f^r))) $\equiv_i$ ($\rhd$ (f^r)) **by** (*simp add*: *first-d-def*)
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *RRLastEqvFirst*:
$\vdash$ ($\lhd$ (f^r))^r $\equiv_i$ ($\rhd$ (*f*))
proof −

have *1*: $\vdash (\lhd\ (f^r))^r \equiv_i \rhd\ ((f^r)^r)$ **using** *RLastEqvFirst* **by** *blast*
have *2*: $\vdash \rhd\ ((f^r)^r) \equiv_i \rhd\ f$ **using** *EqvReverseReverse* **using** *FstEqvRule* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

6.3 Semantic Theorems

6.3.1 Semantics First and Last Operators

lemma *FstAndBisem*:
$(intlen\ \sigma > 0 \wedge (\sigma \models f) \wedge (\ \sigma \models\ \ bi\ \neg_i f;skip)) =$
$(intlen\ \sigma > 0 \wedge (\sigma \models f) \wedge (\forall\ ia{<}intlen\ (\sigma).\ (prefix\ ia\ \ \sigma \models \neg_i f))\ )$
apply *simp-all*
apply (*simp add*: *interval-prefix-length interval-suffix-length*)
proof −
have *1*: $(0 < intlen\ \sigma \wedge (\sigma \models f) \wedge$
$(\exists\ i.\ (i \leq intlen\ \sigma \longrightarrow (\forall\ ia{\leq}i.\ \neg\ (prefix\ ia\ (prefix\ i\ \sigma) \models f)) \wedge$
$intlen\ \sigma - i = Suc\ 0) \wedge i \leq intlen\ \sigma)$
$) =$
$(0 < intlen\ \sigma \wedge (\sigma \models f) \wedge$
$(\exists\ i.\ (i \leq intlen\ \sigma \longrightarrow (\forall\ ia{\leq}i.\ \neg\ (prefix\ ia\ (prefix\ i\ \sigma) \models f)) \wedge$
$i = intlen\ \sigma - Suc\ 0) \wedge i \leq intlen\ \sigma)$
$)$
by *auto*
also have ... =
$(0 < intlen\ \sigma \wedge (\sigma \models f) \wedge$
$(\forall\ ia{\leq}(intlen\ \sigma - Suc\ 0).\ \neg\ (prefix\ ia\ (prefix\ (intlen\ \sigma - Suc\ 0)\ \sigma) \models f)\ )$
$)$
using *diff-le-self* **by** *blast*
also have ... =
$(intlen\ \sigma > 0 \wedge (\sigma \models f) \wedge$
$(\forall\ ia{<}intlen\ (\ \sigma).\ \neg\ (prefix\ ia\ (prefix\ (intlen\ \sigma - Suc\ 0)\ \sigma) \models f))$
$)$ **by** (*metis Suc-pred less-Suc-eq-le*)
also have ... =
$(intlen\ \sigma > 0 \wedge (\sigma \models f) \wedge$
$(\forall\ ia{<}intlen\ (\ \sigma).\ (prefix\ ia\ (prefix\ (intlen\ \sigma - Suc\ 0)\ \sigma) \models \neg_i f))$
$)$
using *not-defs* **by** *blast*
also have ... =
$(intlen\ \sigma > 0 \wedge (\sigma \models f) \wedge (\forall\ ia{<}intlen\ (\ \sigma).\ \neg\ (prefix\ ia\ \ \sigma \models f)))$
by (*simp add*: *interval-pref-pref-help*)
finally show $(0 < intlen\ \sigma \wedge (\sigma \models f) \wedge$
$(\exists\ i.\ (i \leq intlen\ \sigma \longrightarrow (\forall\ ia{\leq}i.\ \neg\ (prefix\ ia\ (prefix\ i\ \sigma) \models f)) \wedge$
$intlen\ \sigma - i = Suc\ 0) \wedge i \leq intlen\ \sigma)$
$) =$
$(0 < intlen\ \sigma \wedge (\sigma \models f) \wedge (\forall\ ia{<}intlen\ \sigma.\ \neg\ (prefix\ ia\ \sigma \models f)))$ **.**
qed

lemma *Fstsem-0*:
$(\sigma \models \rhd f) =$
(

($\sigma \models f \wedge_i$ *empty*) $\vee$ (*intlen* $\sigma > 0$ $\wedge$ ($\sigma \models f$) $\wedge$ ($\sigma \models$ *bi* $\neg_i f$;*skip*))
)
apply (*simp add*: *first-d-def bs-d-def*) **using** *neq0-conv* **by** *blast*

lemma *Emptysem*:
($\sigma \models f \wedge_i$ *empty*) = (($\sigma \models f$) $\wedge$ *intlen* σ = *0*)
by *simp*

lemma *Fstsem*:
($\sigma \models \rhd f$) =
(
(($\sigma \models f$) $\wedge$ *intlen* σ = *0*) $\vee$
(*intlen* σ>*0* $\wedge$ ($\sigma \models f$) $\wedge$ ($\forall$ *ia*<*intlen* (σ). (*prefix ia* $\sigma \models \neg_i f$)))
)
using *Fstsem-0 Emptysem FstAndBisem* **by** *blast*

lemma *Lstsem*:
($\sigma \models \lhd f$) =
((($\sigma \models f$) $\wedge$ *intlen* σ = *0*) $\vee$
(*intlen* σ >*0* $\wedge$ ($\sigma \models f$) $\wedge$ ($\forall$ *ia* < *intlen* σ. (*suffix* ((*intlen* σ) −*ia*) $\sigma \models \neg_i f$)))
)

proof −
have ($\sigma \models \lhd f$) = ($\sigma \models (\rhd (f^r))^r$)
using *RRFirstEqvLast iff-defs itl-valid* **by** *blast*
also have ... = (*intrev* $\sigma \models \rhd (f^r)$)
by (*metis TimeReverseSem interval-rev-rev-ident*)
also have ... =
(
((*intrev* $\sigma \models f^r$) $\wedge$ *intlen* (*intrev* σ) = *0*) $\vee$
(*intlen* (*intrev* σ)>*0* $\wedge$ (*intrev* $\sigma \models f^r$) $\wedge$
($\forall$ *ia*<*intlen* (*intrev* σ). (*prefix ia* (*intrev* σ) $\models \neg_i(f^r)$)))
)
using *Fstsem* **by** *blast*
also have ... =
(
(($\sigma \models f$) $\wedge$ *intlen* (σ) = *0*) $\vee$
(*intlen* (σ)>*0* $\wedge$ ($\sigma \models f$) $\wedge$
($\forall$ *ia*<*intlen* (*intrev* σ). (*prefix ia* (*intrev* σ) $\models (\neg_i(f))^r$)))
)
by (*metis TimeReverseSem interval-intrev-intlen not-d-def rev-d.simps*(*1*) *rev-d.simps*(*3*))
also have ... =
(
(($\sigma \models f$) $\wedge$ *intlen* (σ) = *0*) $\vee$
(*intlen* (σ)>*0* $\wedge$ ($\sigma \models f$) $\wedge$
($\forall$ *ia*<*intlen* (*intrev* σ). (*intrev* (*prefix ia* (*intrev* σ)) $\models (\neg_i(f))$)))
)
using *TimeReverseSem* **by** (*metis interval-rev-rev-ident*)
also have ... =
(

(($\sigma \models f$) $\wedge$ *intlen* (σ) = *0*) $\vee$
(*intlen* (σ)>*0* $\wedge$ ($\sigma \models f$) $\wedge$
(∀ *ia*<*intlen* (σ). ((*suffix* ((*intlen* σ) − *ia*) (σ)) $\models$ ($\neg_i$(f)))))
)
by (*simp add*: *interval-intrev-prefix*)
finally show
($\sigma \models \lhd f$) =
((($\sigma \models f$) $\wedge$ *intlen* σ = *0*) $\vee$
(*intlen* σ >*0* $\wedge$ ($\sigma \models f$) $\wedge$
(∀ *ia* < *intlen* σ. (*suffix* ((*intlen* σ) −*ia*) $\sigma \models \neg_i f$)))
) .
qed

6.3.2 Various Semantic Lemmas

lemma *DiLensem*:
($\sigma \models$ *di* ($f \wedge_i$ *len*(i))) =
((*prefix* i $\sigma \models f$) $\wedge$ i≤*intlen* σ)
apply *simp-all*
using *interval-prefix-length-good* **by** *auto*

lemma *PrefixFstsem*:
((*prefix* i $\sigma \models \rhd f$) $\wedge$ i≤*intlen* σ) =
(i≤*intlen* σ $\wedge$
(
((*prefix* i $\sigma \models f$) $\wedge$ i = *0*) $\vee$
(i>*0* $\wedge$ (*prefix* i $\sigma \models f$) $\wedge$ (∀ *ia*<i. (*prefix ia* $\sigma \models \neg_i f$)))
)
)
proof −
have *1*: (((*prefix* i σ) $\models \rhd f$)) =
(
(((*prefix* i σ) $\models f$) $\wedge$ *intlen* (*prefix* i σ) = *0*) $\vee$
(*intlen* (*prefix* i σ)>*0* $\wedge$ ((*prefix* i σ) $\models f$) $\wedge$
(∀ *ia*<*intlen* (*prefix* i σ). (*prefix ia* (*prefix* i σ) $\models \neg_i f$)))
)
using *Fstsem* **by** *blast*
hence *2*: (((*prefix* i σ) $\models \rhd f$) $\wedge$ i≤*intlen* σ) =
(i≤*intlen* σ $\wedge$ (
(((*prefix* i σ) $\models f$) $\wedge$ *intlen* (*prefix* i σ) = *0*) $\vee$
(*intlen* (*prefix* i σ)>*0* $\wedge$ ((*prefix* i σ) $\models f$) $\wedge$
(∀ *ia*<*intlen* (*prefix* i σ). (*prefix ia* (*prefix* i σ) $\models \neg_i f$)))
)
)
by *auto*
hence *3*: (((*prefix* i σ) $\models \rhd f$) $\wedge$ i≤*intlen* σ) =
(i≤*intlen* σ $\wedge$ (
(((*prefix* i σ) $\models f$) $\wedge$ i = *0*) $\vee$
(i>*0* $\wedge$ ((*prefix* i σ) $\models f$) $\wedge$ (∀ *ia*<i. (*prefix ia* (*prefix* i σ) $\models \neg_i f$)))
)

)
by *auto*
hence *4*: ($((prefix\ i\ \sigma) \models \triangleright f) \wedge i \le intlen\ \sigma$) =
($i \le intlen\ \sigma \wedge$ (
($((prefix\ i\ \sigma) \models f) \wedge i = 0$) $\vee$
($i>0 \wedge ((prefix\ i\ \sigma) \models f) \wedge (\forall ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
)
)
using *interval-pref-pref-3* **using** *less-imp-add-positive* **by** *fastforce*
from *4* **show** *?thesis* **by** *auto*
qed

lemma *PrefixFstAndsem*:
($(prefix\ i\ \sigma \models \triangleright f \wedge_i g) \wedge i \le intlen\ \sigma$) =
($i \le intlen\ \sigma \wedge$
(
($(prefix\ i\ \sigma \models f \wedge_i g) \wedge i = 0$) $\vee$
($i>0 \wedge (prefix\ i\ \sigma \models f \wedge_i g) \wedge (\forall ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
)
)
using *PrefixFstsem* **by** (*metis and-defs*)

lemma *DiLenFstsem*:
($\sigma \models di\ (\triangleright f \wedge_i len(i))$) =
($i \le intlen\ \sigma \wedge$
(
($(prefix\ i\ \sigma \models f) \wedge i = 0$) $\vee$
($i>0 \wedge (prefix\ i\ \sigma \models f) \wedge (\forall ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
)
)
using *DiLensem PrefixFstsem* **by** *blast*

lemma *DiLenFstAndsem*:
($\sigma \models di\ (\triangleright f \wedge_i g \wedge_i len(i))$) =
($i \le intlen\ \sigma \wedge$
(
($(prefix\ i\ \sigma \models f \wedge_i g) \wedge i = 0$) $\vee$
($i>0 \wedge (prefix\ i\ \sigma \models f \wedge_i g) \wedge (\forall ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
)
)
proof −
have *1*: ($\sigma \models di\ ((\triangleright f \wedge_i g) \wedge_i len(i))$) =
($(prefix\ i\ \sigma \models (\triangleright f \wedge_i g)) \wedge i \le intlen\ \sigma$)
using *DiLensem* **by** *blast*
have *2*: ($(prefix\ i\ \sigma \models (\triangleright f \wedge_i g)) \wedge i \le intlen\ \sigma$)=
($i \le intlen\ \sigma \wedge$
(
($(prefix\ i\ \sigma \models f \wedge_i g) \wedge i = 0$) $\vee$
($i>0 \wedge (prefix\ i\ \sigma \models f \wedge_i g) \wedge (\forall ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
)

)
using *PrefixFstAndsem* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstLenSamesem*:
$((i \leq intlen\ \sigma \wedge$
$($
$((prefix\ i\ \sigma \models f) \wedge i = 0) \vee$
$(i>0 \wedge (prefix\ i\ \sigma \models f) \wedge (\forall\, ia<i.\ (prefix\ ia\ \sigma \models \neg_i f)))$
$)$
$) \wedge$
$(j \leq intlen\ \sigma \wedge$
$($
$((prefix\ j\ \sigma \models f) \wedge j = 0) \vee$
$(j>0 \wedge (prefix\ j\ \sigma \models f) \wedge (\forall\, ia<j.\ (prefix\ ia\ \sigma \models \neg_i f)))$
$)$
$)$
$) \longrightarrow (i=j)$

using *linorder-neqE-nat* **by** (*meson not-defs*)

6.4 Theorems

6.4.1 Fixed length intervals

lemma *LenZeroEqvEmpty*:
$\vdash len(0) \equiv_i empty$
by *simp*

lemma *LenOneEqvSkip*:
$\vdash len(1) \equiv_i skip$
by (*metis ChopEmpty One-nat-def* $len_{-}d.simps(1)$ $len_{-}d.simps(2)$)

lemma *LenNPlusOneA*:
$\vdash len(n+1) \equiv_i skip;len(n)$
by *simp*

lemma *LenEqvLenChopLen*:
$\vdash len(i+j) \equiv_i len(i);len(j)$
proof
(*induct i*)
case *0*
then show *?case* **using** *add.left-neutral* **by** *auto*
next
case (*Suc i*)
then show *?case*
by (*metis ChopAssoc NextEqvNext add-Suc* $len_{-}d.simps(2)$ *next-d-def prop03*)
qed

lemma *LenNPlusOneB*:

$\vdash$ *len*$(n+1) \equiv_i$ *len*(n);*skip*
proof −
have *1*: $\vdash$ *len*$(n+1) \equiv_i$ *len*(n);*len*(1) **by** (*rule LenEqvLenChopLen*)
have *2*: $\vdash$ *len*$(1) \equiv_i$ *skip* **by** (*rule LenOneEqvSkip*)
hence *3*: $\vdash$ *len*(n);*len*$(1) \equiv_i$ *len*(n);*skip* **using** *RightChopEqvChop* **by** *blast*
from *1 3* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *RLenEqvLen*:
$\vdash$ $(len\ k)^r \equiv_i (len\ k)$
apply *simp-all* **using** *TimeReverseSem*
by (*metis interval-intrev-intlen interval-rev-rev-ident len-defs*)

lemma *ExistsLen*:
$(\forall \sigma.\ \exists k.\ (\sigma \models len(k)))$
using *len-defs* **by** *simp*

lemma *AndExistsLen*:
$(\forall \sigma.\ (\sigma \models f) = (\ (\sigma \models f) \wedge (\exists k.\ (\sigma \models len(k)))))$
using *ExistsLen* **by** *simp*

lemma *AndExistsLenChop*:
$(\forall \sigma.\ (\sigma \models f;g) = (\exists k.\ \ (\sigma \models (f \wedge_i len(k));g\ )))$
using *AndExistsLen* **by** *auto*

lemma *AndExistsLenChopR*:
$(\forall \sigma.\ (\sigma \models f;g) = (\exists k.\ \ (\sigma \models f;(g \wedge_i len(k)))))$
using *AndExistsLen* **by** *auto*

lemma *LFixedAndDistr*:
$\vdash (f0 \wedge_i len(k));g0 \wedge_i (f1 \wedge_i len(k));g1 \equiv_i (f0 \wedge_i f1 \wedge_i len(k));(g0 \wedge_i g1)$
apply *simp-all*
by (*metis interval-prefix-length-good*)

lemma *RFixedAndDistr*:
$\vdash f0;(g0 \wedge_i len(k)) \wedge_i f1;(g1 \wedge_i len(k)) \equiv_i (f0 \wedge_i f1);(g0 \wedge_i g1 \wedge_i len(k))$
apply *simp-all*
apply (*simp add*: *interval-prefix-length interval-suffix-length*)
by (*metis diff-diff-cancel*)

lemma *LFixedAndDistrA*:
$\vdash (f0 \wedge_i len(k));g0 \wedge_i (f1 \wedge_i len(k));g0 \equiv_i (f0 \wedge_i f1 \wedge_i len(k));g0$
proof −
have *1*: $\vdash (f0 \wedge_i len(k));g0 \wedge_i (f1 \wedge_i len(k));g0 \equiv_i (f0 \wedge_i f1 \wedge_i len(k));(g0 \wedge_i g0)$
by (*rule LFixedAndDistr*)
have *2*: $\vdash (f0 \wedge_i f1 \wedge_i len(k));(g0 \wedge_i g0) \equiv_i (f0 \wedge_i f1 \wedge_i len(k));g0$
by *auto*
from *1 2* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *LFixedAndDistrB*:
$\vdash$ $(f0 \wedge_i len(k));g0 \wedge_i (f0 \wedge_i len(k));g1 \equiv_i (f0 \wedge_i len(k));(g0 \wedge_i g1)$
proof –
have *1*: $\vdash$ $(f0 \wedge_i len(k));g0 \wedge_i (f0 \wedge_i len(k));g1 \equiv_i (f0 \wedge_i f0 \wedge_i len(k));(g0 \wedge_i g1)$
by (*rule LFixedAndDistr*)
have *2*: $\vdash$ $(f0 \wedge_i f0 \wedge_i len(k));(g0 \wedge_i g1) \equiv_i (f0 \wedge_i len(k));(g0 \wedge_i g1)$
by *auto*
from *1 2* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *RFixedAndDistrA*:
$\vdash$ $f0;(g0 \wedge_i len(k)) \wedge_i f0;(g1 \wedge_i len(k)) \equiv_i f0;(g0 \wedge_i g1 \wedge_i len(k))$
proof –
have *1*: $\vdash$ $f0;(g0 \wedge_i len(k)) \wedge_i f0;(g1 \wedge_i len(k)) \equiv_i (f0 \wedge_i f0);(g0 \wedge_i g1 \wedge_i len(k))$
by (*rule RFixedAndDistr*)
have *2*: $\vdash$ $(f0 \wedge_i f0);(g0 \wedge_i g1 \wedge_i len(k)) \equiv_i f0;(g0 \wedge_i g1 \wedge_i len(k))$
by *auto*
from *1 2* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *RFixedAndDistrB*:
$\vdash$ $f0;(g0 \wedge_i len(k)) \wedge_i f1;(g0 \wedge_i len(k)) \equiv_i (f0 \wedge_i f1);(g0 \wedge_i len(k))$
proof –
have *1*: $\vdash$ $f0;(g0 \wedge_i len(k)) \wedge_i f1;(g0 \wedge_i len(k)) \equiv_i (f0 \wedge_i f1);(g0 \wedge_i g0 \wedge_i len(k))$
by (*rule RFixedAndDistr*)
have *2*: $\vdash$ $(f0 \wedge_i f1);(g0 \wedge_i g0 \wedge_i len(k)) \equiv_i (f0 \wedge_i f1);(g0 \wedge_i len(k))$
by *auto*
from *1 2* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *ChopSkipAndChopSkip*:
$\vdash$ $f0;skip \wedge_i f1;skip \equiv_i (f0 \wedge_i f1);skip$
proof –
have *1*: $\vdash$ $f0;(true_i \wedge_i len(1)) \wedge_i f1;(true_i \wedge_i len(1)) \equiv_i (f0 \wedge_i f1);(true_i \wedge_i len(1))$
by (*rule RFixedAndDistrB*)
have *2*: $\vdash$ $(true_i \wedge_i len(1)) \equiv_i skip$
using *LenOneEqvSkip itl-prop(17) prop03* **by** *blast*
hence *3*: $\vdash$ $f0;(true_i \wedge_i len(1)) \equiv_i f0;skip$
using *RightChopEqvChop* **by** *blast*
have *4*: $\vdash$ $f1;(true_i \wedge_i len(1)) \equiv_i f1;skip$
using *2 RightChopEqvChop* **by** *blast*
have *5*: $\vdash$ $f0;(true_i \wedge_i len(1)) \wedge_i f1;(true_i \wedge_i len(1)) \equiv_i f0;skip \wedge_i f1;skip$
using *3 4* **by** *auto*
have *6*: $\vdash$ $(f0 \wedge_i f1);(true_i \wedge_i len(1)) \equiv_i (f0 \wedge_i f1);skip$
using *2 RightChopEqvChop* **by** *blast*
from *1 5 6* **show** *?thesis* **by** *auto*
qed

lemma *BiAndChopSkipEqv*:
$\vdash$ $(bi\ (f \wedge_i g));skip \equiv_i (bi\ f);skip \wedge_i (bi\ g);skip$

proof −
 have *1*: $\vdash$ *bi* $(f \wedge_i g) \equiv_i$ (*bi f*) $\wedge_i$ (*bi g*)
  by *auto*
 hence *2*: $\vdash$ (*bi* $(f \wedge_i g)$);*skip* $\equiv_i$ (*bi f* $\wedge_i$ *bi g*);*skip*
  by (*rule LeftChopEqvChop*)
 have *3*: $\vdash$ (*bi f* $\wedge_i$ *bi g*);*skip* $\equiv_i$ (*bi f*);*skip* $\wedge_i$ (*bi g*);*skip*
  using *ChopSkipAndChopSkip itl-prop*(*30*) **by** *blast*
 from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *DiAndChopSkipEqv*:
$\vdash$ (*di* $(f \wedge_i g)$);*skip* $\supset_i$ (*di f*);*skip* $\wedge_i$ (*di g*);*skip*
proof −
 have *1*: $\vdash$ *di* $(f \wedge_i g) \supset_i$ (*di f*) $\wedge_i$ (*di g*)
  by *auto*
 hence *2*: $\vdash$ (*di* $(f \wedge_i g)$);*skip* $\supset_i$ (*di f* $\wedge_i$ *di g*);*skip*
  by (*rule LeftChopImpChop*)
 have *3*: $\vdash$ (*di f* $\wedge_i$ *di g*);*skip* $\equiv_i$ (*di f*);*skip* $\wedge_i$ (*di g*);*skip*
  using *ChopSkipAndChopSkip itl-prop*(*30*) **by** *blast*
 from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *NotNotChopSkip*:
$\vdash \neg_i(\neg_i f$;*skip*) $\equiv_i$ *empty* $\vee_i$ (f;*skip*)
by (*metis WprevEqvEmptyOrPrev prev-d-def wprev-d-def*)

lemma *NotChopFixed*:
$\vdash \neg_i(f;(g \wedge_i len(k))) \equiv_i \neg_i(\Diamond(g \wedge_i len(k))) \vee_i (\neg_i f;(g \wedge_i len(k)))$
apply *simp* **by** (*smt diff-diff-cancel interval-suffix-length-good*)

lemma *NotFixedChop*:
$\vdash \neg_i((g \wedge_i len(k));f) \equiv_i \neg_i(di(g \wedge_i len(k))) \vee_i ((g \wedge_i len(k));\neg_i f)$
apply *simp* **by** *auto*

6.4.2 Strict initial intervals

lemma *DsMoreDi*:
$\vdash$ *ds f* $\equiv_i$ *more* $\wedge_i$ (*di f*);*skip*
proof −
 have *1*: $\vdash$ *ds f* $\equiv_i \neg_i$(*bs* $\neg_i f$)
  by (*simp add*: *ds-d-def*)
 have *2*: $\vdash \neg_i$(*bs* $\neg_i f$) $\equiv_i \neg_i$(*empty* $\vee_i$ (*bi* $\neg_i f$);*skip*)
  by (*simp add*: *bs-d-def*)
 have *3*: $\vdash \neg_i$(*empty* $\vee_i$ (*bi* $\neg_i f$);*skip*) $\equiv_i \neg_i$*empty* $\wedge_i \neg_i$((*bi* $\neg_i f$);*skip*)
  by *auto*
 have *4*: $\vdash \neg_i$*empty* $\wedge_i \neg_i$((*bi* $\neg_i f$);*skip*) $\equiv_i$ *more* $\wedge_i \neg_i$((*bi* $\neg_i f$);*skip*)
  by *auto*
 have *5*: $\vdash$ *more* $\wedge_i \neg_i$((*bi* $\neg_i f$);*skip*) $\equiv_i$ *more* $\wedge_i \neg_i(\neg_i$(*di f*);*skip*)
  by *auto*

have *6*: $\vdash$ *more* $\wedge_i$ $\neg_i(\neg_i($*di f*$)$;*skip*$)$ $\equiv_i$ *more* $\wedge_i$ $($*empty* $\vee_i$ $($*di f*$)$;*skip*$)$
using *NotNotChopSkip* **using** *prop05* **by** *blast*
have *7*: $\vdash$ *more* $\wedge_i$ $($*empty* $\vee_i$ $($*di f*$)$;*skip*$)$ $\equiv_i$ *more* $\wedge_i$ $($*di f*$)$;*skip*
by *auto*
from *1 2 3 4 5 6 7* **show** *?thesis* **by** *auto*
qed

lemma *DsDi*:
$\vdash$ *ds f* $\equiv_i$ $($*di f*$)$;*skip*
proof $-$
have *1*: $\vdash$ *ds f* $\equiv_i$ *more* $\wedge_i$ $($*di f*$)$;*skip* **by** $($*rule DsMoreDi*$)$
have *2*: $\vdash$ $($*di f*$)$;*skip* $\supset_i$ *more* **by** *auto*
hence *3*: $\vdash$ *more* $\wedge_i$ $($*di f*$)$;*skip* $\equiv_i$ $($*di f*$)$;*skip* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *BsEqvNotDsNot*:
$\vdash$ *bs f* $\equiv_i$ $\neg_i($*ds* $\neg_i$ *f*$)$
proof $-$
have *1*: $\vdash$ *ds* $\neg_i$ *f* $\equiv_i$ *more* $\wedge_i$ $($*di* $\neg_i$ *f*$)$;*skip* **by** $($*rule DsMoreDi*$)$
hence *2*: $\vdash$ $\neg_i($*ds* $\neg_i$ *f*$)$ $\equiv_i$ $\neg_i($*more* $\wedge_i$ $($*di* $\neg_i$ *f*$)$;*skip*$)$ **by** *auto*
have *3*: $\vdash$ $\neg_i($*more* $\wedge_i$ $($*di* $\neg_i$ *f*$)$;*skip*$)$ $\equiv_i$ *empty* $\vee_i$ $\neg_i(($*di* $\neg_i$ *f*$)$;*skip*$)$ **by** *auto*
have *4*: $\vdash$ *empty* $\vee_i$ $\neg_i(($*di* $\neg_i$ *f*$)$;*skip*$)$ $\equiv_i$ *empty* $\vee_i$ $\neg_i(\neg_i($*bi f*$)$;*skip*$)$ **by** *auto*
have *5*: $\vdash$ $\neg_i(\neg_i($*bi f*$)$;*skip*$)$ $\equiv_i$ *empty* $\vee_i$ $($*bi f*$)$;*skip* **by** $($*rule NotNotChopSkip*$)$
hence *6*: $\vdash$ *empty* $\vee_i$ $\neg_i(\neg_i($*bi f*$)$;*skip*$)$ $\equiv_i$ *empty* $\vee_i$ $($*bi f*$)$;*skip* **by** *auto*
from *2 3 4 6* **show** *?thesis* **by** $($*simp add*: *bs-d-def*$)$
qed

lemma *NotBsEqvDsNot*:
$\vdash$ $\neg_i($*bs f*$)$ $\equiv_i$ *ds* $\neg_i$ *f*
proof $-$
have *1*: $\vdash$ *bs f* $\equiv_i$ $\neg_i($*ds* $\neg_i$ *f*$)$ **by** $($*rule BsEqvNotDsNot*$)$
hence *2*: $\vdash$ $\neg_i($ *bs f*$)$ $\equiv_i$ $\neg_i\neg_i($*ds* $\neg_i$ *f*$)$ **by** *auto*
from *2* **show** *?thesis* **by** *auto*
qed

lemma *NotDsEqvBsNot*:
$\vdash$ $\neg_i$ $($*ds f*$)$ $\equiv_i$ *bs* $\neg_i$ *f*
proof $-$
have *1*: $\vdash$ $\neg_i($*ds f*$)$ $\equiv_i$ $\neg_i\neg_i($*bs* $\neg_i$*f*$)$ **by** $($*simp add*: *ds-d-def*$)$
from *1* **show** *?thesis* **by** *auto*
qed

lemma *NotDsAndEmpty*:
$\vdash$ $\neg_i($*ds f* $\wedge_i$ *empty*$)$
proof $-$
have *1*: $\vdash$ *ds f* $\equiv_i$ *more* $\wedge_i$ $($*di f*$)$;*skip* **by** $($*rule DsMoreDi*$)$
have *2*: $\vdash$ *more* $\wedge_i$ $($*di f*$)$;*skip* $\wedge_i$ *empty* $\supset_i$ *false*$_i$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *BsMoreEqvEmpty*:
$\vdash$ *bs more* $\equiv_i$ *empty*
proof $-$
have *1*: $\vdash$ *bs more* $\equiv_i$ *empty* $\vee_i$ (*bi more*);*skip* **by** (*simp add*: *bs-d-def*)
have *2*: $\vdash$ *bi more* $\supset_i$ *false*$_i$ **using** *DiEmpty* **by** *auto*
hence *3*: $\vdash$ (*bi more*);*skip* $\supset_i$ *false*$_i$ **by** *auto*
hence *4*: $\vdash$ *empty* $\vee_i$ ((*bi more*);*skip*) $\equiv_i$ *empty* **using** *prop25* **by** *blast*
from *1 4* **show** *?thesis* **by** *auto*
qed

lemma *BsAndEqv*:
$\vdash$ *bs f* $\wedge_i$ *bs g* $\equiv_i$ *bs*(*f* $\wedge_i$ *g*)
proof $-$
have *1*: $\vdash$ *bs f* $\equiv_i$ *empty* $\vee_i$ (*bi f*) ;*skip*
 by (*simp add*: *bs-d-def*)
have *2*: $\vdash$ *bs g* $\equiv_i$ *empty* $\vee_i$ (*bi g*) ;*skip*
 by (*simp add*: *bs-d-def*)
have *3*: $\vdash$ *bs f* $\wedge_i$ *bs g* $\equiv_i$ (*empty* $\vee_i$ (*bi f*) ;*skip*) $\wedge_i$ (*empty* $\vee_i$ (*bi g*) ;*skip*)
 using *1 2* **by** *auto*
have *4*: $\vdash$ (*empty* $\vee_i$ (*bi f*) ;*skip*) $\wedge_i$ (*empty* $\vee_i$ (*bi g*) ;*skip*) $\equiv_i$
  empty $\vee_i$ ((*bi f*) ;*skip* $\wedge_i$ (*bi g*) ;*skip*)
 by *auto*
have *5*: $\vdash$ ((*bi f*) ;*skip* $\wedge_i$ (*bi g*) ;*skip*) $\equiv_i$ *bi*(*f* $\wedge_i$ *g*);*skip*
 using *BiAndChopSkipEqv itl-prop*(*30*) **by** *blast*
hence *6*: $\vdash$ *empty* $\vee_i$ ((*bi f*) ;*skip* $\wedge_i$ (*bi g*) ;*skip*) $\equiv_i$ *empty* $\vee_i$ *bi*(*f* $\wedge_i$ *g*);*skip*
 by *auto*
from *3 4 6* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *DsEqvRule*:
assumes $\vdash$ *f* $\equiv_i$ *g*
shows $\vdash$ *ds f* $\equiv_i$ *ds g*
by (*meson DiEqvDi DsDi LeftChopEqvChop assms itl-prop*(*30*) *prop03*)

lemma *DsOrEqv*:
$\vdash$ *ds f* $\vee_i$ *ds g* $\equiv_i$ *ds* (*f* $\vee_i$ *g*)
proof $-$
have *1*: $\vdash$ *ds f* $\equiv_i$ $\neg_i$(*bs* $\neg_i$ *f*) **by** (*simp add*: *ds-d-def*)
have *2*: $\vdash$ *ds g* $\equiv_i$ $\neg_i$(*bs* $\neg_i$ *g*) **by** (*simp add*: *ds-d-def*)
have *3*: $\vdash$ *ds f* $\vee_i$ *ds g* $\equiv_i$ $\neg_i$(*bs* $\neg_i$ *f*) $\vee_i$ $\neg_i$(*bs* $\neg_i$ *g*) **using** *1 2* **by** *auto*
have *4*: $\vdash$ $\neg_i$(*bs* $\neg_i$ *f*) $\vee_i$ $\neg_i$(*bs* $\neg_i$ *g*) $\equiv_i$ $\neg_i$(*bs* $\neg_i$ *f* $\wedge_i$ *bs* $\neg_i$ *g*) **by** *auto*
have *5*: $\vdash$ *bs* $\neg_i$ *f* $\wedge_i$ *bs* $\neg_i$ *g* $\equiv_i$ *bs*($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*) **by** (*rule BsAndEqv*)
hence *6*: $\vdash$ $\neg_i$(*bs* $\neg_i$ *f* $\wedge_i$ *bs* $\neg_i$ *g*) $\equiv_i$ $\neg_i$(*bs*($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*)) **by** *auto*
have *7*: $\vdash$ $\neg_i$(*bs*($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*)) $\equiv_i$ *ds* ($\neg_i$($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*)) **by** (*rule NotBsEqvDsNot*)
have *8*: $\vdash$ $\neg_i$($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*) $\equiv_i$ (*f* $\vee_i$ *g*) **by** *auto*
hence *9*: $\vdash$ *ds*($\neg_i$($\neg_i$ *f* $\wedge_i$ $\neg_i$*g*)) $\equiv_i$ *ds* (*f* $\vee_i$ *g*) **by** (*rule DsEqvRule*)
from *3 4 6 7 9* **show** *?thesis* **by** *auto*
qed

lemma *BiOrBiImpBiOr*:
$\vdash$ *bi f* $\vee_i$ *bi g* $\supset_i$ *bi*$(f \vee_i g)$
proof $-$
have *1*: $\vdash f \supset_i f \vee_i g$ **by** *auto*
hence *2*: $\vdash$ *bi f* $\supset_i$ *bi*$(f \vee_i g)$ **by** (*rule BiImpBiRule*)
have *3*: $\vdash g \supset_i f \vee_i g$ **by** *auto*
hence *4*: $\vdash$ *bi g* $\supset_i$ *bi*$(f \vee_i g)$ **by** (*rule BiImpBiRule*)
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *BsOrImp*:
$\vdash$ *bs f* $\vee_i$ *bs g* $\supset_i$ *bs*$(f \vee_i g)$
proof $-$
have *1*: $\vdash$ *bi f* $\vee_i$ *bi g* $\supset_i$ *bi*$(f \vee_i g)$
by (*rule BiOrBiImpBiOr*)
hence *2*: $\vdash$ (*bi f* $\vee_i$ *bi g*);*skip* $\supset_i$ (*bi*$(f \vee_i g)$);*skip*
by (*rule LeftChopImpChop*)
have *3*: $\vdash$ (*bi f*);*skip* $\vee_i$ (*bi g*);*skip* $\supset_i$ (*bi*$(f \vee_i g)$);*skip*
using *1 OrChopEqv 2 itl-prop*(*31*) *prop02* **by** *blast*
hence *4*: $\vdash$ *empty* $\vee_i$ (*bi f*);*skip* $\vee_i$ (*bi g*);*skip* $\supset_i$ *empty* $\vee_i$ (*bi*$(f \vee_i g)$);*skip*
by *auto*
hence *5*: $\vdash$ *empty* $\vee_i$ (*bi f*);*skip* $\vee_i$ *empty* $\vee_i$ (*bi g*);*skip* $\supset_i$ *empty* $\vee_i$ (*bi*$(f \vee_i g)$);*skip*
by *auto*
from *5* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *DsAndImp*:
$\vdash$ *ds* $(f \wedge_i g) \supset_i$ *ds f* $\wedge_i$ *ds g*
proof $-$
have *1*: $\vdash$ *bs* $\neg_i f \vee_i$ *bs* $\neg_i g \supset_i$ *bs*$(\neg_i f \vee_i \neg_i g)$ **by** (*rule BsOrImp*)
have *2*: $\vdash \neg_i f \vee_i \neg_i g \equiv_i \neg_i(f \wedge_i g)$ **by** *auto*
hence *3*: $\vdash$ *bs*$(\neg_i f \vee_i \neg_i g) \equiv_i$ *bs* $\neg_i(f \wedge_i g)$ **by** (*rule BsEqvRule*)
have *4*: $\vdash$ *bs* $\neg_i f \vee_i$ *bs* $\neg_i g \supset_i$ *bs* $\neg_i(f \wedge_i g)$ **using** *1 3* **by** *auto*
have *5*: $\vdash$ *bs* $\neg_i f \equiv_i \neg_i$(*ds f*) **using** *NotDsEqvBsNot* **by** *auto*
have *6*: $\vdash$ *bs* $\neg_i g \equiv_i \neg_i$(*ds g*) **using** *NotDsEqvBsNot* **by** *auto*
have *7*: $\vdash$ *bs* $\neg_i(f \wedge_i g) \equiv_i \neg_i$(*ds* $(f \wedge_i g)$) **using** *NotDsEqvBsNot* **by** *auto*
have *8*: $\vdash \neg_i$(*ds f*) $\vee_i \neg_i$(*ds g*) $\supset_i \neg_i$(*ds* $(f \wedge_i g)$) **using** *4 5 6 7* **by** *auto*
hence *9*: $\vdash \neg_i$(*ds f* $\wedge_i$ *ds g*) $\supset_i \neg_i$(*ds* $(f \wedge_i g)$) **by** *auto*
from *9* **show** *?thesis* **by** *auto*
qed

lemma *DsAndImpElimL*:
$\vdash$ *ds* $(f \wedge_i g) \supset_i$ *ds f*
using *DsAndImp* **by** *auto*

lemma *DsAndImpElimR*:
$\vdash$ *ds* $(f \wedge_i g) \supset_i$ *ds g*
using *DsAndImp* **by** *auto*

lemma *MoreAndBiImpBiChopSkip*:

$\vdash more \wedge_i bi\ f \supset_i (bi\ f);skip$
proof −
 have *1*: $\vdash\ (bi\ f);skip \equiv_i \neg_i(di \neg_i f);skip$ **by** *auto*
 have *2*: $\vdash \neg_i(\neg_i(di \neg_i f);skip) \equiv_i empty \vee_i (di \neg_i f);skip$ **by** (*rule NotNotChopSkip*)
 have *3*: $\vdash empty \supset_i empty \vee_i di \neg_i f$ **by** *auto*
 have *4*: $\vdash (di \neg_i f);skip \supset_i di \neg_i f$ **using** *ChopImpDi DiEqvDiDi itl-prop*(*31*) *prop02* **by** *blast*
 hence *5*: $\vdash (di \neg_i f);skip \supset_i empty \vee_i di \neg_i f$ **by** (*rule prop26*)
 have *6*: $\vdash \neg_i(\neg_i(di \neg_i f);skip) \supset_i empty \vee_i di \neg_i f$ **using** *2 3 5* **by** *auto*
 hence *7*: $\vdash \neg_i(empty \vee_i di \neg_i f) \supset_i \neg_i(\neg_i(\neg_i(di \neg_i f);skip))$ **by** (*rule prop27*)
 have *8*: $\vdash \neg_i(\neg_i(\neg_i(di \neg_i f);skip)) \equiv_i \neg_i(di \neg_i f);skip$ **by** *auto*
 have *9*: $\vdash \neg_i(empty \vee_i di \neg_i f) \equiv_i more \wedge_i \neg_i(di \neg_i f)$ **by** *auto*
 have *10*: $\vdash more \wedge_i \neg_i(di \neg_i f) \equiv_i more \wedge_i bi\ f$ **by** *auto*
 from *1 6 7 8 9 10* **show** *?thesis* **by** *auto*
qed

lemma *BiImpBs*:
$\vdash bi\ f \supset_i bs\ f$
proof −
 have *1*: $\vdash empty \supset_i empty \vee_i (bi\ f);skip$ **by** *auto*
 hence *2*: $\vdash empty \wedge_i bi\ f \supset_i empty \vee_i (bi\ f);skip$ **by** *auto*
 have *2*: $\vdash more \wedge_i bi\ f \supset_i (bi\ f);skip$ **by** (*rule MoreAndBiImpBiChopSkip*)
 hence *3*: $\vdash more \wedge_i bi\ f \supset_i empty \vee_i (bi\ f);skip$ **by** *auto*
 have *4*: $\vdash bi\ f \equiv_i (bi\ f \wedge_i empty) \vee_i (bi\ f \wedge_i more)$ **by** *auto*
 have *5*: $\vdash empty \vee_i (bi\ f);skip \equiv_i bs\ f$ **by** (*simp add*: *bs-d-def*)
 from *2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *BsImpBsBs*:
$\vdash bs\ f \supset_i bs\ (bs\ f)$
proof −
 have *1*: $\vdash bi\ f \supset_i bs\ f$ **by** (*rule BiImpBs*)
 hence *2*: $\vdash bi\ (bi\ f) \supset_i bi(bs\ f)$ **by** (*rule BiImpBiRule*)
 hence *3*: $\vdash (bi\ f) \supset_i bi(bs\ f)$ **using** *BiEqvBiBi itl-prop*(*31*) *prop02* **by** *blast*
 hence *4*: $\vdash (bi\ f);skip \supset_i (bi(bs\ f));skip$ **by** (*rule LeftChopImpChop*)
 hence *5*: $\vdash empty \vee_i (bi\ f);skip \supset_i empty \vee_i (bi(bs\ f));skip$ **by** *auto*
 from *5* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *DsImpDi*:
$\vdash ds\ f \supset_i di\ f$
proof −
 have *1*: $\vdash bi \neg_i f \supset_i bs \neg_i f$ **by** (*rule BiImpBs*)
 hence *2*: $\vdash \neg_i(bs \neg_i f) \supset_i \neg_i(bi \neg_i f)$ **by** (*rule prop27*)
 from *2* **show** *?thesis* **using** *NotBsEqvDsNot DiEqvNotBiNot* **by** (*simp add*: *ds-d-def*)
qed

lemma *BsImpBsRule*:
 assumes $\vdash f \supset_i g$
 shows $\vdash bs\ f \supset_i bs\ g$
proof −

have *1*: $\vdash$ *f* $\supset_i$ *g* **using** *assms* **by** *auto*
hence *2*: $\vdash$ *bi f* $\supset_i$ *bi g* **by** (*rule BiImpBiRule*)
hence *3*: $\vdash$ (*bi f*);*skip* $\supset_i$ (*bi g*);*skip* **by** (*rule LeftChopImpChop*)
hence *4*: $\vdash$ *empty* $\vee_i$ (*bi f*);*skip* $\supset_i$ *empty* $\vee_i$ (*bi g*);*skip* **by** *auto*
from *4* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *DiChopImpDiB*:
$\vdash$ *di*(*f*;*g*) $\supset_i$ *di f*
proof $-$
have *1*: $\vdash$ *f* ; (*g*;*true*$_i$) $\supset_i$ *di f* **by** (*rule ChopImpDi*)
have *2*: $\vdash$ *f* ; (*g*;*true*$_i$) $\equiv_i$ (*f*;*g*);*true*$_i$ **by** (*rule ChopAssoc*)
from *1 2* **show** *?thesis* **by** (*simp add*: *di-d-def*)
qed

lemma *DsChopImpDsB*:
$\vdash$ *ds* (*f*;*g*) $\supset_i$ *ds f*
proof $-$
have *1*: $\vdash$ *di*(*f*;*g*) $\supset_i$ *di f* **by** (*rule DiChopImpDiB*)
hence *2*: $\vdash$ (*di*(*f*;*g*));*skip* $\supset_i$ (*di f*);*skip* **by** (*rule LeftChopImpChop*)
from *2* **show** *?thesis* **using** *DsDi* **by** (*metis itl-prop*(*31*) *prop02*)
qed

lemma *NotBsImpBsNotChop*:
$\vdash$ *bs* $\neg_i$ *f* $\supset_i$ *bs* ($\neg_i$(*f*;*g*))
proof $-$
have *1*: $\vdash$ *ds* (*f*;*g*) $\supset_i$ *ds f* **by** (*rule DsChopImpDsB*)
hence *2*: $\vdash$ $\neg_i$(*ds f*) $\supset_i$ $\neg_i$(*ds* (*f*;*g*)) **by** (*rule prop27*)
from *2* **show** *?thesis* **using** *NotDsEqvBsNot* **by** *auto*
qed

lemma *BiBiOrImpBi*:
$\vdash$ *bi* (*bi f* $\vee_i$ *bi g*) $\supset_i$ *bi f* $\vee_i$ *bi g*
using *BiElim* **by** *auto*

lemma *BiImpBiBiOr*:
$\vdash$ *bi f* $\supset_i$ *bi* (*bi f* $\vee_i$ *bi g*)
proof $-$
have *1*: $\vdash$ *bi f* $\supset_i$ *bi f* $\vee_i$ *bi g* **by** *auto*
hence *2*: $\vdash$ *bi* (*bi f*) $\supset_i$ *bi*(*bi f* $\vee_i$ *bi g*) **using** *BiImpBiRule* **by** *blast*
have *3*: $\vdash$ *bi* (*bi f*) $\equiv_i$ *bi f* **using** *BiEqvBiBi itl-prop*(*30*) **by** *blast*
from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *BiImpBiBiOrB*:
$\vdash$ *bi g* $\supset_i$ *bi* (*bi f* $\vee_i$ *bi g*)
proof $-$
have *1*: $\vdash$ *bi g* $\supset_i$ *bi f* $\vee_i$ *bi g* **by** *auto*
hence *2*: $\vdash$ *bi* (*bi g*) $\supset_i$ *bi*(*bi f* $\vee_i$ *bi g*) **using** *BiImpBiRule* **by** *blast*
have *3*: $\vdash$ *bi* (*bi g*) $\equiv_i$ *bi g* **using** *BiEqvBiBi itl-prop*(*30*) **by** *blast*

from *2 3* **show** *?thesis* **by** *auto*
qed

lemma *BiBiOrEqvBi*:
$\vdash bi\ (\ bi\ f \vee_i bi\ g) \equiv_i bi\ f \vee_i bi\ g$
proof −
have *1*: $\vdash bi\ (\ bi\ f \vee_i bi\ g) \supset_i bi\ f \vee_i bi\ g$ **by** (*rule BiBiOrImpBi*)
have *2*: $\vdash bi\ f \supset_i bi\ (\ bi\ f \vee_i bi\ g)$ **by** (*rule BiImpBiBiOr*)
have *3*: $\vdash bi\ g \supset_i bi\ (\ bi\ f \vee_i bi\ g)$ **by** (*rule BiImpBiBiOrB*)
have *4*: $\vdash bi\ f \vee_i bi\ g \supset_i bi\ (\ bi\ f \vee_i bi\ g)$ **using** *2 3* **by** *auto*
from *1 4* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *BsOrBsEqvBsBiOrBi*:
$\vdash bs\ f \vee_i bs\ g \equiv_i bs(bi\ f \vee_i bi\ g)$
proof −
have *1*: $\vdash bs\ f \vee_i bs\ g \equiv_i empty \vee_i (bi\ f);skip \vee_i empty \vee_i (bi\ g);skip$
by (*simp add*: *bs-d-def*)
have *2*: $\vdash empty \vee_i (bi\ f);skip \vee_i empty \vee_i (bi\ g);skip \equiv_i empty \vee_i (bi\ f);skip \vee_i (bi\ g);skip$
by *auto*
have *3*: $\vdash (bi\ f);skip \vee_i (bi\ g);skip \equiv_i (bi\ f \vee_i bi\ g);skip$
using *OrChopEqv* **using** *itl-prop*(*30*) **by** *blast*
hence *4*: $\vdash empty \vee_i (bi\ f);skip \vee_i (bi\ g);skip \equiv_i empty \vee_i (bi\ f \vee_i bi\ g);skip$
by *auto*
have *5*: $\vdash bi\ (\ bi\ f \vee_i bi\ g) \equiv_i bi\ f \vee_i bi\ g$
by (*rule BiBiOrEqvBi*)
hence *6*: $\vdash bi\ (\ bi\ f \vee_i bi\ g);skip \equiv_i (bi\ f \vee_i bi\ g);skip$
using *LeftChopEqvChop* **by** *blast*
hence *7*: $\vdash empty \vee_i bi\ (\ bi\ f \vee_i bi\ g);skip \equiv_i empty \vee_i (bi\ f \vee_i bi\ g);skip$
by *auto*
from *1 2 4 7* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *DiOrDsEqvDi*:
$\vdash di\ f \vee_i ds\ f \equiv_i di\ f$
proof −
have *1*: $\vdash di\ f \supset_i di\ f \vee_i ds\ f$ **by** *auto*
have *2*: $\vdash di\ f \supset_i di\ f$ **by** *auto*
have *3*: $\vdash ds\ f \supset_i di\ f$ **by** (*rule DsImpDi*)
have *4*: $\vdash di\ f \vee_i ds\ f \supset_i di\ f$ **using** *2 3* **by** *auto*
from *1 4* **show** *?thesis* **by** *auto*
qed

lemma *DiAndDsEqvDs*:
$\vdash di\ f \wedge_i ds\ f \equiv_i ds\ f$
proof −
have *1*: $\vdash di\ f \wedge_i ds\ f \supset_i ds\ f$ **by** *auto*
have *2*: $\vdash ds\ f \supset_i ds\ f$ **by** *auto*
have *3*: $\vdash ds\ f \supset_i di\ f$ **by** (*rule DsImpDi*)

have *4*: ⊢ *ds f* $\supset_i$ *di f* $\wedge_i$ *ds f* **using** *2 3* **by** *auto*
from *1 4* **show** *?thesis* **by** *auto*
qed

lemma *DiEqvOrDiChopSkipA*:
⊢ *di f* $\equiv_i$ *f* $\vee_i$ *di*(*f*;*skip*)
proof −
have *1*: ⊢ *di f* $\equiv_i$ *f* ;*true*$_i$ **by** (*simp add*: *di-d-def*)
hence *2*: ⊢ *di f* $\equiv_i$ *f*; (*empty* $\vee_i$ *more*) **by** *auto*
hence *3*: ⊢*f*; (*empty* $\vee_i$ *more*) $\equiv_i$ *f*;*empty* $\vee_i$ *f*;*more* **using** *ChopOrEqv* **by** *blast*
have *4*: ⊢ *f*;*empty* $\equiv_i$ *f* **by** (*rule ChopEmpty*)
have *5*: ⊢ *more* $\equiv_i$ *skip*;*true*$_i$ **using** *MoreEqvSkipChopTrue* **by** *blast*
hence *6*: ⊢ *f*;*more* $\equiv_i$ *f*;(*skip*;*true*$_i$) **using** *RightChopEqvChop* **by** *blast*
have *7*: ⊢ *f*;(*skip*;*true*$_i$) $\equiv_i$ (*f*;*skip*);*true*$_i$ **by** (*rule ChopAssoc*)
from *2 3 4 6 7* **show** *?thesis* **by** *auto*
qed

lemma *SkipTrueEqvTrueSkip*:
⊢ *skip*;*true*$_i$ $\equiv_i$ *true*$_i$;*skip*
using *TrueChopSkipEqvSkipChopTrue itl-prop*(*30*) **by** *blast*

lemma *DiEqvOrDiChopSkipB*:
⊢ *di f* $\equiv_i$ *f* $\vee_i$ (*di f*);*skip*
proof −
have *1*: ⊢ (*di f*) $\equiv_i$ *f* $\vee_i$ *di*(*f*;*skip*) **by** (*rule DiEqvOrDiChopSkipA*)
have *2*: ⊢ *di*(*f*;*skip*) $\equiv_i$ (*f*;*skip*);*true*$_i$ **by** (*simp add*: *di-d-def*)
have *3*: ⊢ (*f*;*skip*);*true*$_i$ $\equiv_i$ *f*;(*skip*;*true*$_i$) **by** (*rule ChopAssocB*)
have *4*: ⊢ *di*(*f*;*skip*) $\equiv_i$ *f*;(*skip*;*true*$_i$) **using** *2 3* **by** *auto*
have *5*: ⊢ *skip*;*true*$_i$ $\equiv_i$ *true*$_i$;*skip* **by** (*rule SkipTrueEqvTrueSkip*)
hence *6*: ⊢ *f*;(*skip*;*true*$_i$) $\equiv_i$ *f*;(*true*$_i$;*skip*) **using** *RightChopEqvChop* **by** *blast*
have *7*: ⊢ *di*(*f*;*skip*) $\equiv_i$ *f*;(*true*$_i$;*skip*) **using** *4 6* **by** *auto*
have *8*: ⊢ *f*;(*true*$_i$;*skip*) $\equiv_i$ (*f*;*true*$_i$);*skip* **by** (*rule ChopAssoc*)
have *9*: ⊢ (*f*;*true*$_i$);*skip* $\equiv_i$ (*di f*);*skip* **by** (*simp add*: *di-d-def*)
have *10* : ⊢ *di*(*f*;*skip*) $\equiv_i$(*di f*);*skip* **using** *7 8 9* **by** *auto*
hence *11*: ⊢ *f* $\vee_i$ *di*(*f*;*skip*) $\equiv_i$ *f* $\vee_i$ (*di f*);*skip* **by** *auto*
from *1 11* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *OrDsEqvDi*:
⊢ *f* $\vee_i$ *ds f* $\equiv_i$ *di f*
proof −
have *1*: ⊢ *ds f* $\equiv_i$ (*di f*);*skip* **by** (*rule DsDi*)
hence *2*: ⊢ *f* $\vee_i$ *ds f* $\equiv_i$ *f* $\vee_i$ (*di f*);*skip* **by** *auto*
from *2* **show** *?thesis* **using** *DiEqvOrDiChopSkipB itl-prop*(*30*) *prop03* **by** *blast*
qed

lemma *BiEqvAndEmptyOrBiChopSkip*:
⊢ *bi f* $\equiv_i$ *f* $\wedge_i$ (*empty* $\vee_i$ (*bi f*);*skip*)
proof −
have *1*: ⊢ *di* $\neg_i$ *f* $\equiv_i$ $\neg_i$ *f* $\vee_i$ (*di* $\neg_i$ *f*;*skip*) **by** (*rule DiEqvOrDiChopSkipB*)

have *2*: $\vdash di\ \neg_i\ f \equiv_i \neg_i(\ bi\ f\ )$ **by** (*rule DiNotEqvNotBi*)
have *3*: $\vdash \neg_i(\ bi\ f\ ) \equiv_i\ \neg_i\ f \vee_i (di\ \neg_i\ f;skip)$ **using** *1 2* **using** *itl-prop*(*30*) *prop03* **by** *blast*
hence *4*: $\vdash bi\ f \equiv_i \neg_i(\neg_i\ f \vee_i (di\ \neg_i\ f;skip))$ **by** (*metis 1 bi-d-def prop01*)
have *5*: $\vdash \neg_i(\neg_i\ f \vee_i (di\ \neg_i\ f;skip)) \equiv_i f \wedge_i \neg_i(di\ \neg_i\ f;skip)$ **by** *auto*
have *6*: $\vdash di\ \neg_i\ f;skip \equiv_i \neg_i(bi\ f);skip$ **by** *auto*
hence *7*: $\vdash \neg_i(di\ \neg_i\ f;skip) \equiv_i \neg_i(\neg_i(bi\ f);skip)$ **by** *auto*
have *8*: $\vdash \neg_i(\neg_i(bi\ f);skip) \equiv_i (empty \vee_i (bi\ f);skip)$ **using** *NotNotChopSkip* **by** *blast*
hence *9*: $\vdash f \wedge_i \neg_i(di\ \neg_i\ f;skip) \equiv_i f \wedge_i (empty \vee_i (bi\ f);skip)$ **using** *7 8* **by** *auto*
from *4 5 9* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *AndBsEqvBi*:
$\vdash f \wedge_i bs\ f \equiv_i bi\ f$
proof –
have *1*: $\vdash f \wedge_i bs\ f \equiv_i f \wedge_i (empty \vee_i (bi\ f);skip)$ **by** (*simp add*: *bs-d-def*)
from *1* **show** *?thesis* **using** *BiEqvAndEmptyOrBiChopSkip* **by** (*metis bs-d-def itl-prop*(*30*))
qed

lemma *BsEqvBsBi*:
$\vdash bs\ f \equiv_i bs\ (bi\ f)$
proof –
have *1*: $\vdash bs\ f \equiv_i empty \vee_i (bi\ f);skip$ **by** (*simp add*: *bs-d-def*)
have *2*: $\vdash bi\ f \equiv_i bi\ (\ bi\ f)$ **by** (*rule BiEqvBiBi*)
hence *3*: $\vdash (bi\ f);skip \equiv_i bi\ (bi\ f);skip$ **using** *LeftChopEqvChop* **by** *blast*
hence *4*: $\vdash\ empty \vee_i (bi\ f);skip \equiv_i\ empty \vee_i bi\ (bi\ f);skip$ **by** *auto*
from *1 4* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *StateImpBs*:
$\vdash init\ w \supset_i bs\ (init\ w)$
proof –
have *1*: $\vdash init\ w \equiv_i bi\ (init\ w)$ **by** (*rule StateEqvBi*)
have *2*: $\vdash bi\ (init\ w) \supset_i bs\ (init\ w)$ **by** (*rule BiImpBs*)
from *1 2* **show** *?thesis* **using** *StateImpBi prop02* **by** *blast*
qed

lemma *DiDiAndEqvDi*:
$\vdash di\ (\ di\ f \wedge_i di\ g) \equiv_i di\ f \wedge_i di\ g$
proof –
have *1*: $\vdash bi\ (\ bi\ \neg_i\ f \vee_i bi\ \neg_i\ g) \equiv_i bi\ \neg_i f \vee_i bi\ \neg_i\ g$
by (*rule BiBiOrEqvBi*)
have *2*: $\vdash bi\ \neg_i f \equiv_i \neg_i\ (di\ f)$
by *auto*
have *3*: $\vdash bi\ \neg_i g \equiv_i \neg_i\ (di\ g)$
by *auto*
have *4*: $\vdash bi\ \neg_i f \vee_i bi\ \neg_i\ g \equiv_i \neg_i\ (di\ f) \vee_i \neg_i\ (di\ g)$
using *2 3* **by** *auto*
have *5*: $\vdash \neg_i\ (di\ f) \vee_i \neg_i\ (di\ g) \equiv_i \neg_i(di\ f \wedge_i di\ g)$
by *auto*
have *6*: $\vdash bi\ (\ bi\ \neg_i\ f \vee_i bi\ \neg_i\ g) \equiv_i \neg_i(di\ f \wedge_i di\ g)$

using *1 5* **by** *auto*

hence *7*: $\vdash \neg_i(bi\ (\ bi\ \neg_i\ f \vee_i\ bi\ \neg_i\ g)) \equiv_i (di\ f \wedge_i\ di\ g)$

by *simp*

have *8* : $\vdash \neg_i(bi\ (\ bi\ \neg_i\ f \vee_i\ bi\ \neg_i\ g)) \equiv_i di\ (\ \neg_i(bi\ \neg_i\ f \vee_i\ bi\ \neg_i\ g))$

using *DiNotEqvNotBi itl-prop*(30) **by** *blast*

have *9* : $\vdash \neg_i(bi\ \neg_i\ f \vee_i\ bi\ \neg_i\ g) \equiv_i di\ f \wedge_i\ di\ g$

by *auto*

hence *10*: $\vdash di\ (\ \neg_i(bi\ \neg_i\ f \vee_i\ bi\ \neg_i\ g)) \equiv_i di\ (\ di\ f \wedge_i\ di\ g)$

using *DiEqvDi* **by** *blast*

from *7 8 10* **show** *?thesis* **using** *itl-prop*(30) *prop03* **by** *blast*

qed

lemma *BiInduct*:

$\vdash bi(f \supset_i wprev\ f) \wedge_i f \supset_i bi\ f$

proof $-$

have *1*: $\vdash \Box((f^r) \supset_i wnext(f^r)) \wedge_i f^r \supset_i \Box(f^r)$ **using** *BoxInduct* **by** *blast*

hence *2*: $\vdash (\Box((f^r) \supset_i wnext(f^r)) \wedge_i f^r \supset_i \Box(f^r))^r$ **using** *ReverseEqv* **by** *blast*

have *3*: $\vdash ((f^r)^r\ ) \equiv_i f$ **using** *EqvReverseReverse itl-prop*(30) **by** *blast*

have *4*: $\vdash (\Box\ (f^r))^r \equiv_i bi\ (f)$ **using** *RRBoxEqvBi* **by** *blast*

have *5*: $\vdash ((f^r) \supset_i wnext(f^r))^r \equiv_i (\ (f^r)^r \supset_i (wnext(f^r))^r\ )$ **by** *simp*

have *6*: $\vdash (wnext(f^r))^r \equiv_i wprev(f)$ **using** *RRWNextEqvWPrev* **by** *blast*

have *7*: $\vdash (\ (f^r)^r \supset_i (wnext(f^r))^r\ ) \equiv_i (\ f \supset_i wprev(f)\ )$ **using** *6 3 prop39* **by** *auto*

have *8*: $\vdash bi(\ (f^r)^r \supset_i (wnext(f^r))^r\ ) \equiv_i bi(\ f \supset_i wprev(f)\ )$ **using** *7 3 BiEqvBi* **by** *blast*

have *9*: $\vdash (\Box((f^r) \supset_i wnext(f^r)))^r \equiv_i bi\ (\ ((f^r) \supset_i wnext(f^r))^r\ )$ **using** *RBoxEqvBi* **by** *blast*

have *10*: $\vdash (\Box((f^r) \supset_i wnext(f^r)))^r \equiv_i bi(\ f \supset_i wprev(f)\ )$ **using** *8 9* **by** *auto*

have *11*: $\vdash (\Box((f^r) \supset_i wnext(f^r)) \wedge_i f^r \supset_i \Box(f^r))^r \equiv_i$
$((\Box((f^r) \supset_i wnext(f^r)))^r \wedge_i (f^r)^r \supset_i (\Box(f^r))^r)$ **using** *RAnd* **by** *auto*

have *12*: $\vdash ((\Box((f^r) \supset_i wnext(f^r)))^r \wedge_i (f^r)^r \supset_i (\Box(f^r))^r) \equiv_i$
$(bi(\ f \supset_i wprev(f)\ ) \wedge_i f \supset_i bi\ f\ )$ **using** *8 3 4 10* **by** *simp*

from *2 11 12* **show** *?thesis* **using** *MP itl-prop*(31) **by** *blast*

qed

lemma *PrevLoop*:

assumes $\vdash f \supset_i prev\ f$

shows $\vdash \neg_i\ f$

proof $-$

have *1*: $\vdash f \supset_i prev\ f$ **using** *assms* **by** *auto*

hence *2*: $\vdash f \supset_i (\ more \wedge_i wprev\ f)$ **by** *auto*

hence *3*: $\vdash f \supset_i wprev\ f$ **by** *auto*

hence *4*: $\vdash bi(f \supset_i wprev\ f)$ **by** (*rule BiGen*)

have *5*: $\vdash bi(f \supset_i wprev\ f) \wedge_i f \supset_i bi\ f$ **by** (*rule BiInduct*)

hence *6*: $\vdash bi(f \supset_i wprev\ f) \supset_i (f \supset_i bi\ f)$ **using** *prop36* **by** *blast*

have *7*: $\vdash (f \supset_i bi\ f)$ **using** *4 6 MP* **by** *blast*

have *8*: $\vdash bi\ f \supset_i f$ **by** (*rule BiElim*)

have *9*: $\vdash f \equiv_i bi\ f$ **using** *7 8 itl-prop*(31) **by** *blast*

have *10*: $\vdash f \supset_i more$ **using** *2* **by** *auto*

hence *11*: $\vdash bi\ f \supset_i bi\ more$ **using** *BiImpBiRule* **by** *blast*

have *12*: $\vdash \neg_i(bi\ more)$ **using** *DiEmpty* **by** *auto*

from *7 9 11 12* **show** *?thesis* **using** *MP prop27* **by** *blast*

qed

lemma *PrevImpNotPrevNot*:
$\vdash$ *prev* $f \supset_i \neg_i($*prev* $\neg_i\ f)$
by *auto*

lemma *BiEqvAndWprevBi*:
$\vdash$ *bi* $f \equiv_i f \wedge_i$ *wprev*$($*bi* $f)$
proof $-$
have *1*: $\vdash \Box\ (f^r) \equiv_i f^r \wedge_i$ *wnext*$(\Box\ (f^r))$
    using *BoxEqvAndWnextBox* **by** *blast*
hence *2*: $\vdash (\Box\ (f^r) \equiv_i f^r \wedge_i$ *wnext*$(\Box\ (f^r)))^r$
    using *ReverseEqv* **by** *blast*
have *3*: $\vdash (\Box\ (f^r))^r \equiv_i$ *bi* (f)
    using *RRBoxEqvBi* **by** *blast*
have *4*: $\vdash (f^r)^r \equiv_i f$
    using *EqvReverseReverse itl-prop(30)* **by** *blast*
have *5*: $\vdash ($*wnext*$(\Box\ (f^r)))^r \equiv_i$ *wprev*$(\ (\Box(f^r))^r\ )$
    using *RWNextEqvWPrev* **by** *blast*
have *6*: $\vdash$ *wprev*$(\ (\Box(\ ((f^r))))^r) \equiv_i$ *wprev* $($*bi*$(f))$
    using *3 5* **by** *auto*
have *7*: $\vdash ($*wnext*$(\Box\ (f^r)))^r \equiv_i$ *wprev* $($*bi* $(f))$
    using *5 6* **by** *auto*
have *8*: $\vdash (\Box\ (f^r) \equiv_i f^r \wedge_i$ *wnext*$(\Box\ (f^r)))^r \equiv_i$
        $(\ (\Box\ (f^r))^r \equiv_i ((f^r)^r) \wedge_i ($*wnext*$(\Box\ (f^r)))^r)$
    by (*meson 1 2 RAnd REqvRule iff-defs prop03 valid-def*)
have *9*: $\vdash (\ (\Box\ (f^r))^r \equiv_i ((f^r)^r) \wedge_i ($*wnext*$(\Box\ (f^r)))^r) \equiv_i$
        $(\ ($*bi* $f) \equiv_i f \wedge_i$ *wprev*$($*bi* $f))$
    using *7 3 4 prop40* **by** *auto*
from *9 8 2* **show** *?thesis* **by** *auto*
qed

lemma *DiIntroLoop*:
assumes $\vdash (f \wedge_i \neg_i\ g) \supset_i$ *prev* f
shows $\vdash f \supset_i$ *di* g
proof $-$
have *1*: $\vdash f \wedge_i \neg_i\ g \supset_i$ *prev* f
    using *assms* **by** *auto*
hence *2*: $\vdash f \wedge_i \neg_i\ g \wedge_i ($*bi* $\neg_i\ g) \supset_i ($*prev* $f) \wedge_i ($*bi* $\neg_i\ g)$
    by *auto*
have *3*: $\vdash ($*bi* $\neg_i\ g) \supset_i \neg_i\ g$
    by (*rule BiElim*)
hence *4*: $\vdash$ *bi* $\neg_i\ g \equiv_i ($*bi* $\neg_i\ g) \wedge_i \neg_i\ g$
    using *prop38* **by** *blast*
have *5*: $\vdash f \wedge_i ($*bi* $\neg_i\ g) \supset_i$ *prev* $f \wedge_i$ *bi* $\neg_i\ g$
    using *2 4* **by** *auto*
have *6*: $\vdash$ *bi* $\neg_i\ g \equiv_i (\neg_i\ g) \wedge_i$ *wprev*$($*bi* $\neg_i\ g)$
    by (*rule BiEqvAndWprevBi*)
have *7*: $\vdash$ *prev* $f \wedge_i$ *bi* $\neg_i\ g \supset_i$ *prev* $f \wedge_i$ *wprev*$($*bi* $\neg_i\ g)$
    using *6* **using** *itl-prop(31) itl-prop(32) prop12* **by** *blast*
have *8*: $\vdash f \wedge_i ($*bi* $\neg_i\ g) \supset_i$ *prev* $f \wedge_i$ *wprev*$($*bi* $\neg_i\ g)$

using *5 7* **by** *auto*
hence *9*: $\vdash f \wedge_i (bi \neg_i g) \supset_i more \wedge_i wprev\ f \wedge_i wprev(bi \neg_i g)$
by *auto*
hence *10*: $\vdash f \wedge_i (bi \neg_i g) \supset_i wprev\ f \wedge_i wprev(bi \neg_i g)$
by *auto*
hence *11*: $\vdash f \wedge_i (bi \neg_i g) \supset_i wprev\ (f \wedge_i bi \neg_i g)$
by *auto*
hence *12*: $\vdash bi(f \wedge_i (bi \neg_i g) \supset_i wprev\ (f \wedge_i bi \neg_i g)\ )$
by (*rule BiGen*)
have *13*: $\vdash bi(f \wedge_i (bi \neg_i g) \supset_i wprev\ (f \wedge_i bi \neg_i g)\ ) \wedge_i f \wedge_i (bi \neg_i g)$
$\supset_i bi(f \wedge_i (bi \neg_i g))$
by (*rule BiInduct*)
hence *14*: $\vdash bi(f \wedge_i (bi \neg_i g) \supset_i wprev\ (f \wedge_i bi \neg_i g)\ ) \supset_i$
$(\ (f \wedge_i (bi \neg_i g)) \supset_i bi(f \wedge_i (bi \neg_i g)))$
using *prop36* **by** *blast*
have *15*: $\vdash (\ (f \wedge_i (bi \neg_i g)) \supset_i bi(f \wedge_i (bi \neg_i g)))$
using *12 14 MP* **by** *blast*
have *16*: $\vdash bi(f \wedge_i (bi \neg_i g)) \supset_i f \wedge_i (bi \neg_i g)$
by (*rule BiElim*)
have *17*: $\vdash bi(f \wedge_i (bi \neg_i g)) \equiv_i (f \wedge_i (bi \neg_i g))$
using *16 15 itl-prop*(*31*) **by** *blast*
have *18*: $\vdash (f \wedge_i (bi \neg_i g)) \supset_i more$
using *9* **by** *auto*
hence *19*: $\vdash bi\ (f \wedge_i (bi \neg_i g)) \supset_i bi\ more$
using *BiImpBiRule* **by** *blast*
have *20*: $\vdash \neg_i(bi\ more)$
using *DiEmpty* **by** *auto*
have *21*: $\vdash \neg_i(f \wedge_i (bi \neg_i g))$
using *17 19 20* **by** *fastforce*
hence *22*: $\vdash \neg_i f \vee_i \neg_i (bi \neg_i g)$
by *auto*
have *23*: $\vdash \neg_i (bi \neg_i g) \equiv_i di\ g$
by *auto*
from *22 23* **show** *?thesis* **by** *auto*
qed

lemma *DiEqvOrChopMore*:
$\vdash di\ f \equiv_i (f \vee_i f;more)$
proof −
have *1*: $\vdash di\ f \equiv_i f;true_i$ **by** *auto*
hence *2*: $\vdash di\ f \equiv_i f;\ (empty \vee_i more)$ **by** *auto*
have *3*: $\vdash f;\ (empty \vee_i more) \equiv_i f;empty \vee_i f;more$ **by** *auto*
have *4*: $\vdash f;empty \equiv_i f$ **by** (*rule ChopEmpty*)
from *2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *DiAndDiEqvDiAndDiOrDiAndDi*:
$\vdash di\ f \wedge_i di\ g \equiv_i di(f \wedge_i di\ g) \vee_i di(g \wedge_i di\ f)$
proof −
have *1*: $\vdash di\ f \equiv_i (f \vee_i f;more)$

using *DiEqvOrChopMore* **by** *blast*
have *2*: $\vdash di\ g\ \equiv_i (g \vee_i g;more)$
using *DiEqvOrChopMore* **by** *blast*
have *3*: $\vdash di\ f \wedge_i di\ g \equiv_i (f \vee_i f;more) \wedge_i (g \vee_i g;more)$
using *1 2* **by** *auto*
have *4*: $\vdash (f \vee_i f;more) \wedge_i (g \vee_i g;more) \equiv_i$
$(f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more) \vee_i (f;more \wedge_i g;more\ )$
by *auto*
have *5*: $\vdash more \equiv_i true_i;skip$
using *MoreEqvSkipChopTrue SkipTrueEqvTrueSkip prop03* **by** *blast*
hence *6*: $\vdash f;more \equiv_i f;(true_i;skip)$
using *RightChopEqvChop* **by** *blast*
have *7*: $\vdash f;(true_i;skip) \equiv_i (f;true_i);skip$
by (*rule ChopAssoc*)
have *8*: $\vdash f;more \equiv_i prev\ (di\ f)$
using *6 7* **by** (*simp add*: *prev-d-def*)
have *9*: $\vdash g;more \equiv_i g;(true_i;skip)$
using *5 RightChopEqvChop* **by** *blast*
have *10*: $\vdash g;(true_i;skip) \equiv_i (g;true_i);skip$
by (*rule ChopAssoc*)
have *11*: $\vdash g;more \equiv_i prev\ (di\ g)$
using *9 10* **by** (*simp add*: *prev-d-def*)
have *12*: $\vdash f;more \wedge_i g;more \equiv_i prev\ (di\ f) \wedge_i prev\ (di\ g)$
using *8 11* **by** *auto*
hence *13*: $\vdash f;more \wedge_i g;more \equiv_i prev\ (di\ f \wedge_i di\ g)$
by *auto*
have *14*: $\vdash (di\ f \wedge_i di\ g) \equiv_i$
$((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more)) \vee_i (f;more \wedge_i g;more)$
using *3 4* **by** *auto*
have *15*: $\vdash (di\ f \wedge_i di\ g) \equiv_i$
$((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more)) \vee_i prev\ (di\ f \wedge_i di\ g)$
using *13 14 prop28* **by** *blast*
hence *16*: $\vdash (di\ f \wedge_i di\ g) \supset_i$
$((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more)) \vee_i prev\ (di\ f \wedge_i di\ g)$
using *itl-prop*(*31*) **by** *blast*
hence *17*: $\vdash (di\ f \wedge_i di\ g) \wedge_i \neg_i((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more)) \supset_i$
$prev\ (di\ f \wedge_i di\ g)$
using *prop29* **by** *blast*
hence *18*: $\vdash (di\ f \wedge_i di\ g) \supset_i di((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more))$
using *DiIntroLoop* **by** *blast*
have *19*: $\vdash di((f \wedge_i g) \vee_i (\ f \wedge_i g;more) \vee_i (g \wedge_i f;more)) \equiv_i$
$di(f \wedge_i g) \vee_i di(\ f \wedge_i g;more) \vee_i di(g \wedge_i f;more)$
by *auto*
have *20*: $\vdash f\ \ \supset_i di\ f$
using *DiIntro* **by** *blast*
hence *21*: $\vdash f \wedge_i g \supset_i g \wedge_i di\ f$
by *auto*
hence *22*: $\vdash di(f \wedge_i g) \supset_i di(g \wedge_i di\ f)$
using *DiImpDi* **by** *blast*
hence *23*: $\vdash di(f \wedge_i g) \supset_i di(g \wedge_i di\ f) \vee_i di(f \wedge_i di\ g)$

by *auto*
have *24*: $\vdash$ *g*;*more* $\supset_i$ *di g*
by *auto*
hence *25*: $\vdash$ *f* $\wedge_i$ *g*;*more* $\supset_i$ *f* $\wedge_i$ *di g*
by *auto*
hence *26*: $\vdash$ *di*(*f* $\wedge_i$ *g*;*more*) $\supset_i$ *di*(*f* $\wedge_i$ *di g*)
using *DiImpDi* **by** *blast*
hence *27*: $\vdash$ *di*(*f* $\wedge_i$ *g*;*more*) $\supset_i$ *di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
by *auto*
have *28*: $\vdash$ *f*;*more* $\supset_i$ *di f*
by *auto*
hence *29*: $\vdash$ *g* $\wedge_i$ *f*;*more* $\supset_i$ *g* $\wedge_i$ *di f*
by *auto*
hence *30*: $\vdash$ *di*(*g* $\wedge_i$ *f*;*more*) $\supset_i$ *di*(*g* $\wedge_i$ *di f*)
using *DiImpDi* **by** *blast*
hence *31*: $\vdash$ *di*(*g* $\wedge_i$ *f*;*more*) $\supset_i$ *di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
by *auto*
have *32*: $\vdash$ *di*(*f* $\wedge_i$ *g*) $\vee_i$ *di*(*f* $\wedge_i$ *g*;*more*) $\vee_i$ *di*(*g* $\wedge_i$ *f*;*more*) $\supset_i$
di(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
using *23 27 31* **by** *auto*
have *33*: $\vdash$ *di*((*f* $\wedge_i$ *g*) $\vee_i$ (*f* $\wedge_i$ *g*;*more*) $\vee_i$ (*g* $\wedge_i$ *f*;*more*)) $\supset_i$
di(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
using *19 32* **by** *auto*
have *34*: $\vdash$ (*di f* $\wedge_i$ *di g*) $\supset_i$ *di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
using *18 33* **by** *auto*
have *35*: $\vdash$ *f* $\supset_i$ *di f*
using *DiIntro* **by** *blast*
hence *36*: $\vdash$ *f* $\wedge_i$ *di g* $\supset_i$ *di f* $\wedge_i$ *di g*
by *auto*
hence *37*: $\vdash$ *di* (*f* $\wedge_i$ *di g*) $\supset_i$ *di* (*di f* $\wedge_i$ *di g*)
using *DiImpDi* **by** *blast*
have *38*: $\vdash$ *di* (*di f* $\wedge_i$ *di g*) $\equiv_i$ *di f* $\wedge_i$ *di g*
using *DiDiAndEqvDi* **by** *blast*
have *39*: $\vdash$ *di* (*f* $\wedge_i$ *di g*) $\supset_i$ *di f* $\wedge_i$ *di g*
using *37 38* **using** *itl-prop*(*31*) *prop02* **by** *blast*
have *40*: $\vdash$ *g* $\supset_i$ *di g*
using *DiIntro* **by** *blast*
hence *41*: $\vdash$ *g* $\wedge_i$ *di f* $\supset_i$ *di f* $\wedge_i$ *di g*
by *auto*
hence *42*: $\vdash$ *di* (*g* $\wedge_i$ *di f*) $\supset_i$ *di* (*di f* $\wedge_i$ *di g*)
using *DiImpDi* **by** *blast*
have *43*: $\vdash$ *di* (*di f* $\wedge_i$ *di g*) $\equiv_i$ *di f* $\wedge_i$ *di g*
using *DiDiAndEqvDi* **by** *blast*
have *44*: $\vdash$ *di* (*g* $\wedge_i$ *di f*) $\supset_i$ *di f* $\wedge_i$ *di g*
using *42 43* **using** *itl-prop*(*31*) *prop02* **by** *blast*
have *45*: $\vdash$ *di* (*f* $\wedge_i$ *di g*) $\vee_i$ *di* (*g* $\wedge_i$ *di f*) $\supset_i$ *di f* $\wedge_i$ *di g*
using *39 44* *prop30* **by** *blast*
from *34 45* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *DsAndDsEqvDsAndDiOrDsAndDi*:
$\vdash$ *ds f* $\wedge_i$ *ds g* $\equiv_i$ *ds* (*f* $\wedge_i$ *di g*) $\vee_i$ *ds*(*g* $\wedge_i$ *di f*)
proof −
have *1*: $\vdash$ *di f* $\wedge_i$ *di g* $\equiv_i$ *di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*)
  by (*rule DiAndDiEqvDiAndDiOrDiAndDi*)
hence *2*: $\vdash$ (*di f* $\wedge_i$ *di g*);*skip* $\equiv_i$ (*di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*));*skip*
  by (*rule LeftChopEqvChop*)
have *3*: $\vdash$ (*di f* $\wedge_i$ *di g*);*skip* $\equiv_i$ (*di f*);*skip* $\wedge_i$ (*di g*);*skip*
  using *ChopSkipAndChopSkip* **using** *itl-prop*(*30*) **by** *blast*
have *4*: $\vdash$ (*di f*);*skip* $\wedge_i$ (*di g*);*skip* $\equiv_i$ (*di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*));*skip*
  using *2 3* **by** *auto*
have *5*: $\vdash$ (*di*(*f* $\wedge_i$ *di g*) $\vee_i$ *di*(*g* $\wedge_i$ *di f*));*skip* $\equiv_i$ *di*(*f* $\wedge_i$ *di g*);*skip* $\vee_i$ *di*(*g* $\wedge_i$ *di f*);*skip*
  using *OrChopEqv* **by** *blast*
have *6*: $\vdash$ *ds f* $\equiv_i$ (*di f*);*skip*
  using *DsDi* **by** *blast*
have *7*: $\vdash$ *ds g* $\equiv_i$ (*di g*);*skip*
  using *DsDi* **by** *blast*
have *8*: $\vdash$ (*di f*);*skip* $\wedge_i$ (*di g*);*skip* $\equiv_i$ *ds f* $\wedge_i$ *ds g*
  using *6 7* **by** *auto*
have *9*: $\vdash$ *ds*(*f* $\wedge_i$ *di g*) $\equiv_i$ *di*(*f* $\wedge_i$ *di g*);*skip*
  using *DsDi* **by** *blast*
have *10*: $\vdash$ *ds*(*g* $\wedge_i$ *di f*) $\equiv_i$ *di*(*g* $\wedge_i$ *di f*);*skip*
  using *DsDi* **by** *blast*
have *11*: $\vdash$ *di*(*f* $\wedge_i$ *di g*);*skip* $\vee_i$ *di*(*g* $\wedge_i$ *di f*);*skip* $\equiv_i$ *ds*(*f* $\wedge_i$ *di g*) $\vee_i$ *ds*(*g* $\wedge_i$ *di f*)
  using *9 10* **by** *auto*
from *4 5 8 11* **show** *?thesis* **by** *simp*
qed

lemma *BsEqvBiMoreImpChop*:
$\vdash$ *bs f* $\equiv_i$ *bi*(*more* $\supset_i$ *f*;*skip*)
proof −
have *1*: $\vdash$ *bs f* $\equiv_i$ *empty* $\vee_i$ (*bi f*;*skip*)
  by (*simp add*: *bs-d-def*)
have *2*: $\vdash$ $\neg_i$($\neg_i$(*bi f*);*skip*) $\equiv_i$ *empty* $\vee_i$ (*bi f*;*skip*)
  using *NotNotChopSkip* **by** *blast*
have *3*: $\vdash$ $\neg_i$($\neg_i$(*bi f*);*skip*) $\equiv_i$ $\neg_i$(*di* $\neg_i$ *f*;*skip*)
  by *auto*
have *4*: $\vdash$ $\neg_i$(*di* $\neg_i$ *f*;*skip*) $\equiv_i$ $\neg_i$(($\neg_i$ *f* ;*true*$_i$);*skip*)
  by (*simp add*:*di-d-def*)
have *5*: $\vdash$ $\neg_i$(($\neg_i$ *f* ;*true*$_i$);*skip*) $\equiv_i$ $\neg_i$($\neg_i$ *f* ;(*true*$_i$;*skip*))
  using *ChopAssocB prop01* **by** *blast*
have *6*: $\vdash$ $\neg_i$($\neg_i$ *f* ;(*true*$_i$;*skip*)) $\equiv_i$ $\neg_i$($\neg_i$ *f* ;(*skip*;*true*$_i$))
  using *SkipTrueEqvTrueSkip* **using** *TrueChopSkipEqvSkipChopTrue RightChopEqvChop prop01* **by** *blast*
have *7*: $\vdash$ $\neg_i$($\neg_i$ *f* ;(*skip*;*true*$_i$)) $\equiv_i$ $\neg_i$(($\neg_i$ *f* ;*skip*);*true*$_i$)
  using *ChopAssoc prop01* **by** *blast*
have *8*: $\vdash$ $\neg_i$(($\neg_i$ *f* ;*skip*);*true*$_i$) $\equiv_i$ $\neg_i$(*di* ($\neg_i$ *f*;*skip*))
  by (*simp add*: *di-d-def*)
have *9*: $\vdash$ $\neg_i$(*di* ($\neg_i$ *f*;*skip*)) $\equiv_i$ *bi* ($\neg_i$($\neg_i$ *f* ;*skip*))
  using *NotDiEqvBiNot* **by** *blast*
have *10*: $\vdash$ *bi* ($\neg_i$($\neg_i$ *f* ;*skip*)) $\equiv_i$ *bi*(*empty* $\vee_i$ (*f*;*skip*))

using *NotNotChopSkip* **using** *BiEqvBi* **by** *blast*
have *11*: $\vdash bi(\ empty \lor_i (f;skip)) \equiv_i bi(\ \neg_i\ more \lor_i (f;skip))$
by *auto*
from *1 2 3 4 5 6 7 8 9 10 11* **show** *?thesis* **by** *auto*
qed

6.4.3 First occurrence

lemma *FstWithAndImp*:
$\vdash \rhd f \land_i g \supset_i \rhd\ (\ f \land_i g)$
proof –
have *1*: $\vdash \rhd f \land_i g \equiv_i f \land_i (bs\ \neg_i f) \land_i g$
by (*simp add*: *first-d-def*)
have *2*: $\vdash f \land_i (bs\ \neg_i f) \land_i g \equiv_i f \land_i \neg_i(ds\ f) \land_i g$
using *NotDsEqvBsNot* **using** *itl-prop*(*30*) *prop05 prop06* **by** *blast*
have *3*: $\vdash \neg_i(ds\ f) \supset_i \neg_i(ds(f \land_i g))$
using *DsAndImpElimL* **using** *prop27* **by** *blast*
hence *4*: $\vdash f \land_i \neg_i(ds\ f) \land_i g \supset_i f \land_i g \land_i \neg_i(ds(f \land_i g))$
by *auto*
have *5*: $\vdash f \land_i g \land_i \neg_i(ds(f \land_i g)) \equiv_i f \land_i g \land_i (bs\ \neg_i(f \land_i g))$
using *NotDsEqvBsNot* **using** *prop05* **by** *blast*
have *6*: $\vdash f \land_i g \land_i (bs\ \neg_i(f \land_i g)) \equiv_i \rhd(f \land_i g)$
by (*simp add*: *first-d-def*)
from *1 2 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *FstWithOrEqv*:
$\vdash \rhd(f \lor_i g) \equiv_i (\rhd f \land_i bs\ \neg_i g) \lor_i (\rhd g \land_i bs\ \neg_i f)$
proof –
have *1*: $\vdash \rhd(f \lor_i g) \equiv_i (f \lor_i g) \land_i bs\ \neg_i(f \lor_i g)$
by (*simp add*: *first-d-def*)
have *2*: $\vdash \neg_i(f \lor_i g) \equiv_i (\neg_i f \land_i \neg_i g)$
by *auto*
hence *3*: $\vdash bs\ \neg_i(f \lor_i g) \equiv_i bs\ (\neg_i f \land_i \neg_i g)$
using *BsEqvRule* **by** *blast*
have *4*: $\vdash bs\ (\neg_i f \land_i \neg_i g) \equiv_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g$
using *BsAndEqv itl-prop*(*30*) **by** *blast*
have *5*: $\vdash (f \lor_i g) \land_i bs\ \neg_i(f \lor_i g) \equiv_i (f \lor_i g) \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g$
using *3 4* **by** *auto*
have *6*: $\vdash (f \lor_i g) \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g \equiv_i$
$(f \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g) \lor_i (g \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g)$
by *auto*
have *7*: $\vdash (f \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g) \equiv_i \rhd f \land_i\ bs\ \neg_i\ g$
by (*simp add*: *first-d-def*)
have *8*: $\vdash (g \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g) \equiv_i (g \land_i bs\ \neg_i\ g \land_i bs\ \neg_i\ f)$
by *auto*
have *9*: $\vdash (g \land_i bs\ \neg_i\ g \land_i bs\ \neg_i\ f) \equiv_i \rhd g \land_i\ bs\ \neg_i\ f$
by (*simp add*: *first-d-def*)
have *10*: $\vdash (f \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g) \lor_i (g \land_i bs\ \neg_i\ f \land_i bs\ \neg_i\ g) \equiv_i$
$(\rhd f \land_i\ bs\ \neg_i\ g) \lor_i (\rhd g \land_i\ bs\ \neg_i\ f)$

using *7 8 9* **by** *auto*
from *1 5 6 10* **show** *?thesis* **by** *auto*
qed

lemma *FstFstAndEqvFstAnd*:
$\vdash \triangleright(\triangleright f \wedge_i g) \equiv_i \triangleright f \wedge_i g$
proof −
have *1*: $\vdash \triangleright f \wedge_i g \equiv_i f \wedge_i (bs\ \neg_i f) \wedge_i g$ **by** (*simp add*: *first-d-def*)
hence *2*: $\vdash \triangleright f \wedge_i g \supset_i (bs\ \neg_i f)$ **by** *auto*
hence *3*: $\vdash \triangleright f \wedge_i g \supset_i \triangleright f \wedge_i g \wedge_i (bs\ \neg_i f)$ **by** *auto*
have *4*: $\vdash \neg_i f \supset_i \neg_i f \vee_i \neg_i(bs\ \neg_i f) \vee_i \neg_i g$ **by** *auto*
hence *5*: $\vdash bs\ (\neg_i f) \supset_i bs(\neg_i f \vee_i \neg_i(bs\ \neg_i f) \vee_i \neg_i g)$ **using** *BsImpBsRule* **by** *blast*
have *6*: $\vdash \neg_i f \vee_i \neg_i(bs\ \neg_i f) \vee_i \neg_i g \equiv_i \neg_i(f \wedge_i bs\ \neg_i f \wedge_i g)$ **by** *auto*
hence *7*: $\vdash bs(\neg_i f \vee_i \neg_i(bs\ \neg_i f) \vee_i \neg_i g) \equiv_i bs(\neg_i(f \wedge_i bs\ \neg_i f \wedge_i g))$ **using** *BsEqvRule* **by** *blast*
have *8*: $\vdash f \wedge_i bs\ \neg_i f \wedge_i g \equiv_i \triangleright f \wedge_i g$ **by** (*simp add*: *first-d-def*)
hence *9*: $\vdash \neg_i(f \wedge_i bs\ \neg_i f \wedge_i g) \equiv_i \neg_i(\triangleright f \wedge_i g)$ **by** *auto*
hence *10*: $\vdash bs\ \neg_i(f \wedge_i bs\ \neg_i f \wedge_i g) \equiv_i bs\ \neg_i(\triangleright f \wedge_i g)$ **using** *BsEqvRule* **by** *blast*
have *11*: $\vdash \triangleright f \wedge_i g \supset_i \triangleright f \wedge_i g \wedge_i\ bs\ \neg_i(\triangleright f \wedge_i g)$ **using** *3 5 7 10* **by** *auto*
hence *12*: $\vdash \triangleright f \wedge_i g \supset_i \triangleright(\triangleright f \wedge_i g)$ **by** (*simp add*: *first-d-def*)
have *13*: $\vdash \triangleright(\triangleright f \wedge_i g) \equiv_i \triangleright f \wedge_i g \wedge_i\ bs\ \neg_i(\triangleright f \wedge_i g)$ **by** (*simp add*: *first-d-def*)
hence *14*: $\vdash \triangleright(\triangleright f \wedge_i g) \supset_i \triangleright f \wedge_i g$ **by** *auto*
from *12 14* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *FstTrue*:
$\vdash \triangleright\ true_i \equiv_i empty$
proof −
have *1*: $\vdash \triangleright\ true_i \equiv_i true_i \wedge_i bs\ \neg_i\ true_i$ **by** (*simp add*: *first-d-def*)
have *2*: $\vdash bs\ \neg_i\ true_i \equiv_i empty \vee_i (bi\ \neg_i\ true_i);skip$ **by** (*simp add*: *bs-d-def*)
have *3*: $\vdash \neg_i(bi\ \neg_i\ true_i)$ **by** *auto*
hence *4*: $\vdash \neg_i((bi\ \neg_i\ true_i);skip)$ **by** *auto*
have *5*: $\vdash bs\ \neg_i\ true_i \equiv_i empty$ **using** *2 4* **by** *auto*
from *1 5* **show** *?thesis* **by** *auto*
qed

lemma *FstFalse*:
$\vdash \neg_i(\triangleright\ false_i)$
proof −
have *1*: $\vdash \triangleright\ false_i \equiv_i false_i \wedge_i bs\ true_i$ **by** (*simp add*: *first-d-def*)
from *1* **show** *?thesis* **by** *auto*
qed

lemma *FstChopFalseEqvFalse*:
$\vdash \neg_i(\triangleright f\ ;\ false_i)$
by *auto*

lemma *FstEmpty*:
$\vdash \triangleright\ empty \equiv_i empty$
proof −
have *1*: $\vdash \triangleright\ empty \equiv_i empty \wedge_i bs\ \neg_i\ empty$ **by** (*simp add*: *first-d-def*)

have *2*: $\vdash$ *bs* $\neg_i$ *empty* $\equiv_i$ *empty* $\vee_i$ *bi* $\neg_i$ *empty*;*skip* **by** (*simp add*: *bs-d-def*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstAndEmptyEqvAndEmpty*:
$\vdash$ $\triangleright f$ $\wedge_i$ *empty* $\equiv_i$ *f* $\wedge_i$ *empty*
proof −
have *1*: $\vdash$ $\triangleright f$ $\wedge_i$ *empty* $\equiv_i$ *f* $\wedge_i$ *bs* $\neg_i$ *f* $\wedge_i$ *empty* **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ *bs* $\neg_i$ *f* $\equiv_i$ *empty* $\vee_i$ *bi* $\neg_i f$;*skip* **by** (*simp add*: *bs-d-def*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstEmptyOrEqvEmpty*:
$\vdash$ $\triangleright$(*empty* $\vee_i$ *f*) $\equiv_i$ *empty*
proof −
have *1*: $\vdash$ $\triangleright$(*empty* $\vee_i$ *f*) $\equiv_i$ ($\triangleright$*empty* $\wedge_i$ *bs* $\neg_i$ *f*) $\vee_i$ ($\triangleright f$ $\wedge_i$ *bs* $\neg_i$ *empty*) **using** *FstWithOrEqv* **by** *blast*
have *2*: $\vdash$ $\neg_i$ *empty* $\equiv_i$ *more* **by** *auto*
hence *3*: $\vdash$ *bs* $\neg_i$ *empty* $\equiv_i$ *bs more* **using** *BsEqvRule* **by** *blast*
have *4*: $\vdash$ *bs more* $\equiv_i$ *empty* **using** *BsMoreEqvEmpty* **by** *blast*
have *5*: $\vdash$ $\triangleright f$ $\wedge_i$ *bs* $\neg_i$ *empty* $\equiv_i$ ($\triangleright f$ $\wedge_i$ *empty*) **using** *3 4* **by** *auto*
have *6*: $\vdash$ $\triangleright$*empty* $\equiv_i$ *empty* **using** *FstEmpty* **by** *blast*
hence *7*: $\vdash$ ($\triangleright$*empty* $\wedge_i$ *bs* $\neg_i$ *f*) $\equiv_i$ (*empty* $\wedge_i$ *bs* $\neg_i$ *f*) **by** *auto*
have *8*: $\vdash$ *empty* $\wedge_i$ *bs* $\neg_i$ *f* $\equiv_i$ *empty* $\wedge_i$(*empty* $\vee_i$ *bi* $\neg_i$ *f*;*skip*) **by** (*simp add*:*bs-d-def*)
have *9*: $\vdash$ *empty* $\wedge_i$(*empty* $\vee_i$ *bi* $\neg_i$ *f*;*skip*) $\equiv_i$ *empty* **by** *auto*
have *10*: $\vdash$ *empty* $\wedge_i$ *bs* $\neg_i$ *f* $\equiv_i$ *empty* **using** *8 9* **by** *auto*
have *11*: $\vdash$ ($\triangleright$*empty* $\wedge_i$ *bs* $\neg_i$ *f*) $\vee_i$ ($\triangleright f$ $\wedge_i$ *bs* $\neg_i$ *empty*) $\equiv_i$
empty $\vee_i$ ($\triangleright f$ $\wedge_i$ *empty*) **using** *7 10 5* **by** *auto*
have *12*: $\vdash$ *empty* $\vee_i$ ($\triangleright f$ $\wedge_i$ *empty*) $\equiv_i$ *empty* **by** *auto*
from *1 11 12* **show** *?thesis* **by** *auto*
qed

lemma *ChopEmptyAndEmpty*:
$\vdash$ *f*;*g* $\wedge_i$ *empty* $\equiv_i$ *f* $\wedge_i$ *g* $\wedge_i$ *empty*
apply *simp-all*
by (*metis interval-prefix-intlen interval-suffix-zero le-zero-eq*)

lemma *FstChopEmptyEqvFstChopFstEmpty*:
$\vdash$ $\triangleright f$;*g* $\wedge_i$ *empty* $\equiv_i$ $\triangleright f$;$\triangleright g$ $\wedge_i$ *empty*
proof −
have *1*: $\vdash$ $\triangleright f$;*g* $\wedge_i$ *empty* $\equiv_i$ $\triangleright f$ $\wedge_i$ *g* $\wedge_i$ *empty* **using** *ChopEmptyAndEmpty* **by** *blast*
have *2*: $\vdash$ *g* $\wedge_i$ *empty* $\equiv_i$ $\triangleright g$ $\wedge_i$ *empty* **using** *FstAndEmptyEqvAndEmpty* **using** *itl-prop*(*30*) **by** *blast*
hence *3*: $\vdash$ $\triangleright f$ $\wedge_i$ *g* $\wedge_i$ *empty* $\equiv_i$ $\triangleright f$ $\wedge_i$ $\triangleright g$ $\wedge_i$ *empty* **by** *auto*
have *4*: $\vdash$ $\triangleright f$;$\triangleright g$ $\wedge_i$ *empty* $\equiv_i$ $\triangleright f$ $\wedge_i$ $\triangleright g$ $\wedge_i$ *empty* **using** *ChopEmptyAndEmpty* **by** *blast*
from *1 3 4* **show** *?thesis* **by** *auto*
qed

lemma *ChopSkipImpMore*:
$\vdash$ *f*;*skip* $\supset_i$ *more*
proof −
have *1*: $\vdash$ $\neg_i$(*f*;*skip* $\wedge_i$ *empty*) **by** *auto*

hence *2*: $\vdash \neg_i(f;skip) \vee_i more$ **by** *auto*
from *2* **show** *?thesis* **by** *auto*
qed

lemma *MoreEqvMoreChopTrue*:
$\vdash more \equiv_i more;true_i$
proof –
have *1*: $\vdash more \equiv_i skip;true_i$
using *MoreEqvSkipChopTrue* **by** *blast*
have *2*: $\vdash true_i \equiv_i true_i;true_i$
by *auto*
hence *3*: $\vdash skip;true_i \equiv_i skip;(true_i;true_i)$
using *RightChopEqvChop* **by** *blast*
have *4*: $\vdash skip;(true_i;true_i) \equiv_i (skip;true_i);true_i$
using *ChopAssoc* **by** *blast*
have *5*: $\vdash (skip;true_i);true_i \equiv_i more;true_i$
using *MoreEqvSkipChopTrue* **by** (*simp add*: *more-d-def next-d-def*)
from *1 3 4 5* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *BiEmptyEqvEmpty*:
$\vdash bi\ empty \equiv_i empty$
proof –
have *1*: $\vdash bi\ empty \equiv_i \neg_i\ (di\ \neg_i\ empty)$ **by** (*simp add*: *bi-d-def*)
have *2*: $\vdash \neg_i\ (di\ \neg_i\ empty) \equiv_i \neg_i\ (\neg_i\ empty;true_i)$ **by** (*simp add*: *di-d-def*)
have *3*: $\vdash \neg_i\ (\neg_i\ empty;true_i) \equiv_i \neg_i\ (more;true_i)$ **by** *auto*
have *4*: $\vdash more;true_i \equiv_i more$ **using** *MoreEqvMoreChopTrue* **by** *auto*
hence *5*: $\vdash \neg_i(more;true_i) \equiv_i \neg_i\ more$ **using** *prop01* **by** *blast*
from *1 2 3 5* **show** *?thesis* **by** *auto*
qed

lemma *FstMoreEqvSkip*:
$\vdash \triangleright\ more \equiv_i skip$
proof –
have *1*: $\vdash \triangleright\ more \equiv_i more \wedge_i bs\ \neg_i\ more$ **by** (*simp add*: *first-d-def*)
have *2*: $\vdash more \wedge_i bs\ \neg_i\ more \equiv_i more \wedge_i (empty \vee_i bi\ \neg_i\ more;skip)$ **by** (*simp add*:*bs-d-def*)
have *3*: $\vdash more \wedge_i (empty \vee_i bi\ \neg_i\ more;skip) \equiv_i more \wedge_i bi\ \neg_i\ more;skip$ **by** *auto*
have *4*: $\vdash more \wedge_i ((bi\ \neg_i\ more);skip) \equiv_i ((bi\ \neg_i\ more);skip)$ **using** *ChopSkipImpMore* **by** *auto*
have *5*: $\vdash ((bi\ \neg_i\ more);skip) \equiv_i bi\ empty;skip$ **by** *auto*
have *6*: $\vdash bi\ empty \equiv_i empty$ **using** *BiEmptyEqvEmpty* **by** *auto*
hence *7*: $\vdash bi\ empty;skip \equiv_i empty;skip$ **using** *LeftChopEqvChop* **by** *blast*
have *8*: $\vdash empty;skip \equiv_i skip$ **using** *EmptyChop* **by** *blast*
from *1 2 3 4 5 7 8* **show** *?thesis* **by** (*metis prop03*)
qed

lemma *FstEqvBsNotAndDi*:
$\vdash \triangleright f \equiv_i bs\ \neg_i\ f \wedge_i di\ f$
proof –
have *1*: $\vdash bs\ \neg_i\ f \equiv_i \neg_i(ds\ f)$ **by** (*simp add*: *ds-d-def*)
hence *2*: $\vdash bs\ \neg_i\ f \wedge_i di\ f \equiv_i \neg_i(ds\ f) \wedge_i di\ f$ **by** *auto*

have *3*: $\vdash$ *di f* $\equiv_i$ (*ds f* $\vee_i$ *f*) **using** *OrDsEqvDi* **by** *auto*
hence *4* : $\vdash$ $\neg_i$(*ds f*) $\wedge_i$ *di f* $\equiv_i$ $\neg_i$(*ds f*) $\wedge_i$ (*ds f* $\vee_i$ *f*) **by** *auto*
have *5* : $\vdash$ $\neg_i$(*ds f*) $\wedge_i$ (*ds f* $\vee_i$ *f*) $\equiv_i$ $\neg_i$(*ds f*) $\wedge_i$ *f* **by** *auto*
have *6* : $\vdash$ $\neg_i$(*ds f*) $\wedge_i$ *f* $\equiv_i$ *f* $\wedge_i$ *bs* $\neg_i$ *f* **using** *1* **by** *auto*
from *2 4 5 6* **show** *?thesis* **by** (*simp add*: *first-d-def*)
qed

lemma *FstOrDiEqvDi*:
$\vdash$ $\triangleright$*f* $\vee_i$ *di f* $\equiv_i$ *di f*
proof –
have *1*: $\vdash$ $\triangleright$*f* $\vee_i$ *di f* $\equiv_i$ (*f* $\wedge_i$ *bs* $\neg_i$*f*) $\vee_i$ *di f* **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ (*f* $\wedge_i$ *bs* $\neg_i$*f*) $\vee_i$ *di f* $\equiv_i$ (*f* $\vee_i$ *di f*) $\wedge_i$ (*bs* $\neg_i$ *f* $\vee_i$ *di f*) **by** *auto*
have *3*: $\vdash$ (*f* $\vee_i$ *di f*) $\equiv_i$ *di f* **by** *auto*
hence *4*: $\vdash$ (*f* $\vee_i$ *di f*) $\wedge_i$ (*bs* $\neg_i$ *f* $\vee_i$ *di f*) $\equiv_i$ *di f* $\wedge_i$ (*bs* $\neg_i$ *f* $\vee_i$ *di f*) **by** *auto*
have *5*: $\vdash$ *di f* $\wedge_i$ (*bs* $\neg_i$ *f* $\vee_i$ *di f*) $\equiv_i$ *di f* **by** *auto*
from *1 2 4 5* **show** *?thesis* **by** *auto*
qed

lemma *FstAndDiEqvFst*:
$\vdash$ $\triangleright$*f* $\wedge_i$ *di f* $\equiv_i$ $\triangleright$*f*
proof –
have *1*: $\vdash$ $\triangleright$*f* $\wedge_i$ *di f* $\equiv_i$ *f* $\wedge_i$ *bs* $\neg_i$*f* $\wedge_i$ *di f* **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ *f* $\wedge_i$ *di f* $\equiv_i$ *f* **by** *auto*
hence *3*: $\vdash$ *f* $\wedge_i$ *bs* $\neg_i$*f* $\wedge_i$ *di f* $\equiv_i$ *f* $\wedge_i$ *bs* $\neg_i$*f* **by** *auto*
from *1 3* **show** *?thesis* **by** (*simp add*: *first-d-def*)
qed

lemma *EmptyChopSkipInduct*:
assumes $\vdash$ *empty* $\supset_i$ *f*
$\vdash$ *prev f* $\supset_i$ *f*
shows $\vdash$ *f*
proof –
have *1*: $\vdash$ *empty* $\supset_i$ *f* **using** *assms*(*1*) **by** *auto*
have *2*: $\vdash$ *prev f* $\supset_i$ *f* **using** *assms*(*2*) **by** *blast*
have *3*: $\vdash$ (*empty* $\vee_i$ *prev f*) $\supset_i$ *f* **using** *1 2 prop30* **by** *blast*
have *4*: $\vdash$ *wprev f* $\equiv_i$ (*empty* $\vee_i$ *prev f*) **by** *auto*
hence *5*: $\vdash$ *wprev f* $\supset_i$ *f* **using** *3* **using** *itl-prop*(*31*) *prop02* **by** *blast*
hence *6*: $\vdash$ $\neg_i$*f* $\supset_i$ $\neg_i$ (*wprev f*) **using** *prop27* **by** *blast*
hence *7*: $\vdash$ $\neg_i$*f* $\supset_i$ *prev* ($\neg_i$ *f*) **by** *auto*
hence *8*: $\vdash$ $\neg_i$ $\neg_i$ *f* **by** (*rule PrevLoop*)
from *8* **show** *?thesis* **by** *auto*
qed

lemma *DiAndEmptyEqvAndEmpty*:
$\vdash$ *di f* $\wedge_i$ *empty* $\equiv_i$ *f* $\wedge_i$ *empty*
proof –
have *1* : $\vdash$ *di f* $\equiv_i$ (*f* $\vee_i$ *di f*;*skip*) **using** *DiEqvOrDiChopSkipB* **by** *blast*
hence *2*: $\vdash$ *di f* $\wedge_i$ *empty* $\equiv_i$ (*f* $\vee_i$ *di f*;*skip*) $\wedge_i$ *empty* **using** *prop06* **by** *blast*
have *3* : $\vdash$ (*f* $\vee_i$ *di f*;*skip*) $\wedge_i$ *empty* $\equiv_i$ (*f* $\wedge_i$ *empty*) $\vee_i$ (*di f*;*skip* $\wedge_i$ *empty*) **by** *auto*

have *4*: $\vdash\ \neg_i(di\ f;skip \wedge_i empty)$ **by** *auto*
hence *5* : $\vdash (f \wedge_i empty) \vee_i (di\ f;skip \wedge_i empty) \equiv_i (f \wedge_i empty)$ **by** *auto*
from *2 3 5* **show** *?thesis* **by** *auto*
qed

lemma *MoreImpImpChopSkipEqv*:
$\vdash more \supset_i ((f\supset_i g);skip \equiv_i ((f;skip)\supset_i(g;skip)))$
proof −
have *1*: $\vdash more \wedge_i (f\supset_i g);skip \equiv_i more \wedge_i\ (\neg_i f \vee_i g);skip$
by *auto*
have *2*: $\vdash (\neg_i f \vee_i g);skip \equiv_i \neg_i f;skip \vee_i g;skip$
using *OrChopEqv* **by** *auto*
hence *3*: $\vdash more \wedge_i\ (\neg_i f \vee_i g);skip \equiv_i more \wedge_i(\neg_i f;skip \vee_i g;skip)$
by *auto*
have *4*: $\vdash \neg_i(\neg_i f;skip) \equiv_i empty \vee_i (f;skip)$
using *NotNotChopSkip* **by** *blast*
hence *5*: $\vdash (\neg_i f;skip) \equiv_i \neg_i(empty \vee_i (f;skip))$
using *itl-prop(30) itl-prop(33) itl-prop(4) prop03* **by** *blast*
have *6*: $\vdash \neg_i(empty \vee_i (f;skip)) \equiv_i (more \wedge_i \neg_i(f;skip))$
by *auto*
have *7*: $\vdash (\neg_i f;skip \vee_i g;skip) \equiv_i ((more \wedge_i \neg_i(f;skip)) \vee_i g;skip)$
using *5 6* **by** *auto*
hence *8*: $\vdash more \wedge_i(\neg_i f;skip \vee_i g;skip) \equiv_i more \wedge_i ((more \wedge_i \neg_i(f;skip)) \vee_i g;skip)$
by *auto*
have *9*: $\vdash more \wedge_i ((more \wedge_i \neg_i(f;skip)) \vee_i g;skip) \equiv_i more \wedge_i (\neg_i(f;skip) \vee_i g;skip)$
by *auto*
have *10*: $\vdash more \wedge_i (\neg_i(f;skip) \vee_i g;skip) \equiv_i more \wedge_i ((f;skip)\supset_i(g;skip))$
by *auto*
have *11*: $\vdash more \wedge_i (f\supset_i g);skip \equiv_i more \wedge_i ((f;skip)\supset_i(g;skip))$
using *1 2 3 8 9 10* **by** *auto*
from *11* **show** *?thesis* **using** *prop31* **using** *MP itl-prop(31)* **by** *blast*
qed

lemma *MoreImpImpPrevEqv*:
$\vdash more \supset_i (prev(f\supset_i g) \equiv_i (prev\ f\supset_i prev\ g))$
using *MoreImpImpChopSkipEqv* **by** *auto*

lemma *DiEqvDiFst*:
$\vdash di\ f \equiv_i di\ (\rhd f)$
proof −
have *1*: $\vdash\ di\ (\rhd f) \equiv_i di\ (f \wedge_i bs\ \neg_i f)$
by (*simp add*: *first-d-def*)
have *2*: $\vdash di\ (f \wedge_i bs\ \neg_i f) \supset_i di\ f \wedge_i di\ (bs\ \neg_i f)$
using *DiAndImpAnd* **by** *auto*
hence *3*: $\vdash di\ (f \wedge_i bs\ \neg_i f) \supset_i di\ f$
by *auto*
have *4*: $\vdash di\ (\rhd f) \supset_i di\ f$ **using** *1 3*
by *auto*
have *5*: $\vdash di\ f \wedge_i empty \equiv_i f \wedge_i empty$
using *DiAndEmptyEqvAndEmpty* **by** *blast*

have *6*: $\vdash \triangleright f \wedge_i empty \equiv_i f \wedge_i empty$
  using *FstAndEmptyEqvAndEmpty* **by** *auto*
have *7*: $\vdash di\ f \wedge_i empty \supset_i \triangleright f$
  using *5 6* **by** *auto*
have *8*: $\vdash \triangleright f \supset_i di\ (\triangleright f)$
  using *DiIntro* **by** *auto*
have *9*: $\vdash\ di\ f \wedge_i empty \supset_i di\ (\triangleright f)$
  using *7 8* **using** *prop02* **by** *blast*
hence *10*: $\vdash empty \supset_i (di\ f \supset_i di\ (\triangleright f))$
  by *auto*
have *11*: $\vdash prev\ (\ di\ f \supset_i di\ (\triangleright f)) \supset_i more$
  by *auto*
have *12*: $\vdash more \supset_i (\ prev\ (\ di\ f \supset_i di\ (\triangleright f)) \equiv_i (prev(di\ f) \supset_i prev(di\ (\triangleright f))))$
  using *MoreImpImpPrevEqv* **by** *auto*
have *13*: $\vdash more \wedge_i prev\ (\ di\ f \supset_i di\ (\triangleright f)) \equiv_i more \wedge_i (prev(di\ f) \supset_i prev(di\ (\triangleright f)))$
  using *12 prop31* **by** *auto*
have *14*: $\vdash prev\ (\ di\ f \supset_i di\ (\triangleright f)) \equiv_i more \wedge_i (prev(di\ f) \supset_i prev(di\ (\triangleright f)))$
  using *11* **by** *auto*
have *15*: $\vdash di\ f \equiv_i f \vee_i ds\ f$
  using *OrDsEqvDi* **by** *auto*
have *16*: $\vdash di\ f \equiv_i di\ f \wedge_i (bs \neg_i f \vee_i \neg_i(bs \neg_i f))$
  by *auto*
have *17*: $\vdash di\ f \wedge_i (bs \neg_i f \vee_i \neg_i(bs \neg_i f)) \equiv_i (di\ f \wedge_i bs \neg_i f) \vee_i (di\ f \wedge_i \neg_i(bs \neg_i f))$
  by *auto*
have *18*: $\vdash (di\ f \wedge_i bs \neg_i f) \equiv_i (f \vee_i ds\ f) \wedge_i bs \neg_i f$
  using *15* **by** *auto*
have *19*: $\vdash (f \vee_i ds\ f) \wedge_i bs \neg_i f \equiv_i (f \wedge_i bs \neg_i f) \vee_i (ds\ f \wedge_i bs \neg_i f)$
  by *auto*
have *20*: $\vdash \neg_i(ds\ f \wedge_i bs \neg_i f)$
  by (*simp add*: *ds-d-def*)
have *21*: $\vdash (f \wedge_i bs \neg_i f) \vee_i (ds\ f \wedge_i bs \neg_i f) \equiv_i (f \wedge_i bs \neg_i f)$
  using *20* **by** *auto*
have *22*: $\vdash (di\ f \wedge_i bs \neg_i f) \equiv_i (f \wedge_i bs \neg_i f)$
  using *18 19 21* **by** *auto*
have *23*: $\vdash (f \wedge_i bs \neg_i f) \equiv_i \triangleright f$
  by (*simp add*: *first-d-def*)
have *24*: $\vdash (\triangleright f) \supset_i di\ (\triangleright f)$
  using *DiIntro* **by** *auto*
have *25*: $\vdash (f \wedge_i bs \neg_i f) \supset_i di\ (\triangleright f)$
  using *23 24* **by** *auto*
have *26*: $\vdash (di\ f \wedge_i bs \neg_i f) \supset_i di\ (\triangleright f)$
  using *25 22* **by** *auto*
hence *27*: $\vdash (di\ f \wedge_i bs \neg_i f \wedge_i (prev\ (di\ f \supset_i di\ (\triangleright f)))) \supset_i di\ (\triangleright f)$
  by *auto*
have *28*: $\vdash di\ f \wedge_i \neg_i(bs \neg_i f) \equiv_i di\ f \wedge_i ds\ f$
  by (*simp add*: *ds-d-def*)
hence *29*: $\vdash di\ f \wedge_i \neg_i(bs \neg_i f) \wedge_i (prev\ (di\ f \supset_i di\ (\triangleright f))) \ \equiv_i$
    $di\ f \wedge_i ds\ f \wedge_i\ (prev\ (di\ f \supset_i di\ (\triangleright f)))$
  by *auto*
have *30*: $\vdash\ ds\ f \equiv_i prev(di\ f)$

using *DsDi* **by** (*metis prev-d-def*)
hence *31*: $\vdash$ *di f* $\wedge_i$ *ds f* $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\equiv_i$
di f $\wedge_i$ *prev*(*di f*) $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*)))
by *auto*
have *32*: $\vdash$ *prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*)) $\supset_i$ (*prev*(*di f*) $\supset_i$ *prev*(*di* ($\triangleright$*f*)))
using *14* **by** *auto*
hence *33*: $\vdash$ *di f* $\wedge_i$ *prev*(*di f*) $\wedge_i$ *prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*)) $\supset_i$
di f $\wedge_i$ *prev*(*di f*) $\wedge_i$ (*prev*(*di f*) $\supset_i$ *prev*(*di* ($\triangleright$*f*)))
by *auto*
have *34*: $\vdash$ *di f* $\wedge_i$ *prev*(*di f*) $\wedge_i$ (*prev*(*di f*) $\supset_i$ *prev*(*di* ($\triangleright$*f*))) $\supset_i$ *prev*(*di* ($\triangleright$*f*))
by *auto*
have *35*: $\vdash$ *prev*(*di* ($\triangleright$*f*))$\equiv_i$ (*di* ($\triangleright$*f*));*skip*
by (*simp add*: *prev-d-def*)
have *36*: $\vdash$ (*di* ($\triangleright$*f*));*skip* $\supset_i$ *di*(*di* ($\triangleright$*f*))
using *ChopImpDi* **by** *auto*
have *37*: $\vdash$ *di*(*di* ($\triangleright$*f*)) $\equiv_i$ *di* ($\triangleright$*f*)
using *DiEqvDiDi* **using** *itl-prop*(*30*) **by** *blast*
have *38*: $\vdash$ *di f* $\wedge_i$ *prev*(*di f*) $\wedge_i$ (*prev*(*di f*) $\supset_i$ *prev*(*di* ($\triangleright$*f*))) $\supset_i$ *di* ($\triangleright$*f*)
using *37 36 35 34 itl-prop*(*31*) *prop02* **by** *blast*
have *39*: $\vdash$ *di f* $\wedge_i$ $\neg_i$(*bs* $\neg_i$ *f*) $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\supset_i$ *di* ($\triangleright$*f*)
using *29 31 33 38* **by** (*meson itl-prop*(*31*) *prop02*)
hence *40*: $\vdash$ $\neg_i$(*bs* $\neg_i$ *f*) $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\supset_i$ (*di f* $\supset_i$ *di* ($\triangleright$*f*))
using *prop32* **by** *blast*
have *41*: $\vdash$ *bs* $\neg_i$ *f* $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\supset_i$(*di f* $\supset_i$ *di* ($\triangleright$*f*))
using *27 prop32* **by** *blast*
have *42*: $\vdash$ ($\neg_i$(*bs* $\neg_i$ *f*) $\vee_i$ *bs* $\neg_i$ *f*) $\wedge_i$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\supset_i$(*di f* $\supset_i$ *di* ($\triangleright$*f*))
using *40 41 prop33* **by** *blast*
have *43*: $\vdash$ ($\neg_i$(*bs* $\neg_i$ *f*) $\vee_i$ *bs* $\neg_i$ *f*)
by *auto*
have *44*: $\vdash$ (*prev* (*di f* $\supset_i$ *di* ($\triangleright$ *f*))) $\supset_i$(*di f* $\supset_i$ *di* ($\triangleright$*f*))
using *42 43 prop34* **by** *blast*
have *45*: $\vdash$ *di f* $\supset_i$ *di* ($\triangleright$*f*)
using *10 44 EmptyChopSkipInduct* **by** *blast*
from *4 45* **show** *?thesis* **by** *auto*
qed

lemma *FstDiEqvFst*:
$\vdash$ $\triangleright$(*di f*) $\equiv_i$ $\triangleright$*f*
proof −
have *1*: $\vdash$ $\triangleright$(*di f*) $\equiv_i$ *di f* $\wedge_i$ *bs* $\neg_i$ (*di f*) **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ $\neg_i$ (*di f*) $\equiv_i$ *bi* $\neg_i$ *f* **by** *auto*
hence *3*: $\vdash$ *bs* $\neg_i$ (*di f*) $\equiv_i$ *bs* (*bi* $\neg_i$ *f*) **using** *BsEqvRule* **by** *blast*
have *4*: $\vdash$ *bs* (*bi* $\neg_i$ *f*) $\equiv_i$ *bs* ($\neg_i$ *f*) **using** *BsEqvBsBi* **using** *itl-prop*(*30*) **by** *blast*
hence *5*: $\vdash$ *di f* $\wedge_i$ *bs* $\neg_i$ (*di f*) $\equiv_i$ *di f* $\wedge_i$ *bs* ($\neg_i$ *f*) **using** *3* **by** *auto*
have *6* : $\vdash$ *di f* $\equiv_i$ *f* $\vee_i$ *ds f* **using** *OrDsEqvDi* **using** *itl-prop*(*30*) **by** *blast*
hence *7*: $\vdash$ *di f* $\wedge_i$ *bs* ($\neg_i$ *f*) $\equiv_i$ (*f* $\vee_i$ *ds f*) $\wedge_i$ *bs* ($\neg_i$ *f*) **by** *auto*
have *8*: $\vdash$ (*f* $\vee_i$ *ds f*) $\wedge_i$ *bs* ($\neg_i$ *f*) $\equiv_i$ (*f* $\wedge_i$ *bs* ($\neg_i$ *f*)) $\vee_i$ (*ds f* $\wedge_i$ *bs* ($\neg_i$ *f*)) **by** *auto*
have *9*: $\vdash$ $\neg_i$(*ds f* $\wedge_i$ *bs* ($\neg_i$ *f*)) **by** (*simp add*: *ds-d-def*)
have *10*: $\vdash$ (*f* $\wedge_i$ *bs* ($\neg_i$ *f*)) $\equiv_i$ $\triangleright$*f* **by** (*simp add*: *first-d-def*)
from *1 5 7 8 9 10* **show** *?thesis* **by** *auto*

qed

lemma *DiAndFstOrEqvFstOrDiAnd*:
$\vdash$ *di f* $\wedge_i$ ($\triangleright f$ $\vee_i$ *g*) $\equiv_i$ $\triangleright f$ $\vee_i$ (*di f* $\wedge_i$ *g*)
proof –
have *1*: $\vdash$ *di f* $\wedge_i$ ($\triangleright f$ $\vee_i$ *g*) $\equiv_i$ ($\triangleright f$ $\wedge_i$ *di f*) $\vee_i$ (*di f* $\wedge_i$ *g*) **by** *auto*
have *2*: $\vdash$ ($\triangleright f$ $\wedge_i$ *di f*) $\equiv_i$ $\triangleright f$ **using** *FstAndDiEqvFst* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *DiOrFstAndEqvDi*:
$\vdash$ *di f* $\vee_i$ ($\triangleright f$ $\wedge_i$ *g*) $\equiv_i$ *di f*
proof –
have *1*: $\vdash$ *di f* $\vee_i$ ($\triangleright f$ $\wedge_i$ *g*) $\equiv_i$ ($\triangleright f$ $\vee_i$ *di f*) $\wedge_i$ (*di f* $\vee_i$ *g*) **by** *auto*
have *2*: $\vdash$ ($\triangleright f$ $\vee_i$ *di f*) $\equiv_i$ *di f* **using** *FstOrDiEqvDi* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstDiAndDiEqv*:
$\vdash$ $\triangleright$(*di f* $\wedge_i$ *di g*) $\equiv_i$ ($\triangleright f$ $\wedge_i$ *di g*) $\vee_i$ ($\triangleright g$ $\wedge_i$ *di f*)
proof –
have *1*: $\vdash$ $\triangleright$(*di f* $\wedge_i$ *di g*) $\equiv_i$ *di f* $\wedge_i$ *di g* $\wedge_i$ *bs* $\neg_i$(*di f* $\wedge_i$ *di g*) **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ $\neg_i$(*di f* $\wedge_i$ *di g*) $\equiv_i$ *bi* $\neg_i f$ $\vee_i$ *bi* $\neg_i g$ **by** *auto*
hence *3*: $\vdash$ *bs* $\neg_i$(*di f* $\wedge_i$ *di g*) $\equiv_i$ *bs*(*bi* $\neg_i f$ $\vee_i$ *bi* $\neg_i g$) **using** *BsEqvRule* **by** *blast*
hence *4*: $\vdash$ *di f* $\wedge_i$ *di g* $\wedge_i$ *bs* $\neg_i$(*di f* $\wedge_i$ *di g*) $\equiv_i$
di f $\wedge_i$ *di g* $\wedge_i$ *bs*(*bi* $\neg_i f$ $\vee_i$ *bi* $\neg_i g$) **by** *auto*
have *5*: $\vdash$ *bs* $\neg_i f$ $\vee_i$ *bs* $\neg_i g$ $\equiv_i$ *bs*(*bi* $\neg_i f$ $\vee_i$ *bi* $\neg_i g$) **using** *BsOrBsEqvBsBiOrBi* **by** *blast*
hence *6*: $\vdash$ *di f* $\wedge_i$ *di g* $\wedge_i$ *bs*(*bi* $\neg_i f$ $\vee_i$ *bi* $\neg_i g$) $\equiv_i$
di f $\wedge_i$ *di g* $\wedge_i$(*bs* $\neg_i f$ $\vee_i$ *bs* $\neg_i g$) **by** *auto*
have *7*: $\vdash$ *di f* $\wedge_i$ *di g* $\wedge_i$(*bs* $\neg_i f$ $\vee_i$ *bs* $\neg_i g$) $\equiv_i$
(*bs* $\neg_i f$ $\wedge_i$ *di f* $\wedge_i$ *di g*) $\vee_i$ (*di f* $\wedge_i$ *bs* $\neg_i g$ $\wedge_i$ *di g*) **by** *auto*
have *8*: $\vdash$ $\triangleright f$ $\equiv_i$ *bs* $\neg_i f$ $\wedge_i$ *di f* **using** *FstEqvBsNotAndDi* **by** *blast*
hence *9*: $\vdash$ *bs* $\neg_i f$ $\wedge_i$ *di f* $\wedge_i$ *di g* $\equiv_i$ $\triangleright f$ $\wedge_i$ *di g* **by** *auto*
have *10*: $\vdash$ $\triangleright g$ $\equiv_i$ *bs* $\neg_i g$ $\wedge_i$ *di g* **using** *FstEqvBsNotAndDi* **by** *blast*
hence *11*: $\vdash$ *di f* $\wedge_i$ *bs* $\neg_i g$ $\wedge_i$ *di g* $\equiv_i$ *di f* $\wedge_i$ $\triangleright g$ **by** *auto*
have *12*: $\vdash$ *di f* $\wedge_i$ *di g* $\wedge_i$(*bs* $\neg_i f$ $\vee_i$ *bs* $\neg_i g$) $\equiv_i$
($\triangleright f$ $\wedge_i$ *di g*) $\vee_i$ (*di f* $\wedge_i$ $\triangleright g$) **using** *7 9 11* **by** *auto*
from *1 4 6 12* **show** *?thesis* **by** *auto*
qed

lemma *BiNotFstEqvBiNot*:
$\vdash$ *bi* $\neg_i$ ($\triangleright f$) $\equiv_i$ *bi* $\neg_i$ *f*
proof –
have *1*: $\vdash$ *di f* $\equiv_i$ *di* ($\triangleright f$) **using** *DiEqvDiFst* **by** *blast*
hence *2*: $\vdash$ $\neg_i$(*di f*) $\equiv_i \neg_i$(*di* ($\triangleright f$)) **by** *auto*
from *1 2* **show** *?thesis* **using** *NotDiEqvBiNot itl-prop*(*30*) *prop03* **by** *blast*
qed

lemma *BsNotFstEqvBsNot*:
$\vdash$ *bs* $\neg_i$ ($\triangleright f$) $\equiv_i$ *bs* $\neg_i$ *f*

proof −
have $1: \vdash bs\ \neg_i\ (\rhd f) \equiv_i empty \lor_i bi\ \neg_i\ (\rhd f);skip$ **by** (*simp add*: *bs-d-def*)
have $2: \vdash bi\ \neg_i\ (\rhd f) \equiv_i bi\ \neg_i\ f$ **using** *BiNotFstEqvBiNot* **by** *blast*
hence $3: \vdash bi\ \neg_i\ (\rhd f);skip \equiv_i bi\ \neg_i\ f;skip$ **using** *LeftChopEqvChop* **by** *blast*
hence $4: \vdash empty \lor_i bi\ \neg_i\ (\rhd f);skip \equiv_i empty \lor_i bi\ \neg_i\ f;skip$ **by** *auto*
from *1 4* **show** *?thesis* **by** (*simp add*: *bs-d-def*)
qed

lemma *BsFalseEqvEmpty*:
$\vdash bs\ false_i \equiv_i empty$
proof −
have $1: \vdash bs\ false_i \equiv_i empty \lor_i bi\ false_i;skip$ **by** (*simp add*: *bs-d-def*)
have $2: \vdash \neg_i(bi\ false_i;skip)$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstState*:
$\vdash \rhd\ (init\ w) \equiv_i empty \land_i init\ w$
proof −
have $1: \vdash \rhd\ (init\ w) \equiv_i init\ w \land_i bs\ \neg_i(init\ w)$ **by** (*simp add*: *first-d-def*)
hence $2: \vdash \rhd\ (init\ w) \supset_i init\ w$ **by** *auto*
have $3: \vdash init\ w \supset_i bs\ (init\ w)$ **using** *StateImpBs* **by** *auto*
have $4: \vdash \rhd\ (init\ w) \supset_i bs\ (init\ w)$ **using** *2 3* **by** *auto*
have $5: \vdash \rhd\ (init\ w) \supset_i bs\ \neg_i(init\ w)$ **using** *1* **by** *auto*
have $6: \vdash \rhd\ (init\ w) \supset_i bs\ (init\ w) \land_i bs\ \neg_i(init\ w)$ **using** *4 5* **by** *auto*
have $7: \vdash bs\ (init\ w) \land_i bs\ \neg_i(init\ w) \equiv_i bs((init\ w) \land_i \neg_i(init\ w))$ **using** *BsAndEqv* **by** *blast*
have $8: \vdash (init\ w) \land_i \neg_i(init\ w) \equiv_i false_i$ **by** *auto*
hence $9: \vdash bs((init\ w) \land_i \neg_i(init\ w)) \equiv_i bs\ false_i$ **using** *BsEqvRule* **by** *blast*
have $10: \vdash bs\ false_i \equiv_i empty$ **using** *BsFalseEqvEmpty* **by** *auto*
have $11: \vdash \rhd\ (init\ w) \supset_i empty$ **using** *10 9 7 6* **by** *auto*
have $12: \vdash \rhd\ (init\ w) \supset_i empty \land_i init\ w$ **using** *11 2* **by** *auto*
have $13: \vdash empty \land_i init\ w \supset_i empty$ **by** *auto*
hence $14: \vdash empty \land_i init\ w \supset_i empty \lor_i bi\ \neg_i(init\ w);skip$ **by** *auto*
hence $15: \vdash empty \land_i init\ w \supset_i bs\ \neg_i(init\ w)$ **by** (*simp add*: *bs-d-def*)
have $16: \vdash empty \land_i init\ w \supset_i init\ w$ **by** *auto*
have $17: \vdash empty \land_i init\ w \supset_i init\ w \land_i bs\ \neg_i(init\ w)$ **using** *16 15* **by** *auto*
hence $18: \vdash empty \land_i init\ w \supset_i \rhd(init\ w)$ **by** (*simp add*: *first-d-def*)
from *12 18* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *FstStateAndBsNotEmpty*:
$\vdash \rhd\ (init\ w) \land_i bs\ \neg_i\ empty \equiv_i \rhd\ (init\ w)$
proof −
have $1: \vdash \rhd\ (init\ w) \land_i bs\ \neg_i\ empty \equiv_i \rhd\ (init\ w) \land_i bs\ more$
using *BsEqvRule NotEmptyEqvMore prop05* **by** *blast*
have $2: \vdash \rhd\ (init\ w) \land_i bs\ more \equiv_i \rhd\ (init\ w) \land_i empty$
using *BsMoreEqvEmpty prop05* **by** *blast*
have $3: \vdash \rhd\ (init\ w) \equiv_i empty \land_i (init\ w)$
using *FstState* **by** *blast*
hence $4: \vdash \rhd\ (init\ w) \land_i empty \equiv_i empty \land_i (init\ w) \land_i empty$

by *auto*
have *5*: $\vdash$ *empty* $\wedge_i$ (*init w*) $\wedge_i$ *empty* $\equiv_i$ *empty* $\wedge_i$ (*init w*)
by *auto*
have *6*: $\vdash$ *empty* $\wedge_i$ (*init w*) $\equiv_i$ $\triangleright$(*init w*)
using *FstState* **using** *itl-prop*(*30*) **by** *blast*
from *1 2 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *FstStateImpFstStateOr*:
$\vdash$ $\triangleright$(*init w*) $\supset_i$ $\triangleright$(*init w* $\vee_i$ *f*)
proof −
have *1*: $\vdash$ $\triangleright$(*init w*) $\equiv_i$ *empty* $\wedge_i$ *init w*
using *FstState* **by** *blast*
have *2*: $\vdash$ *empty* $\wedge_i$ *init w* $\equiv_i$ *empty* $\wedge_i$ (*empty* $\vee_i$ *bi* $\neg_i$ *f*;*skip*) $\wedge_i$ *init w*
by *auto*
have *3*: $\vdash$ *empty* $\wedge_i$ (*empty* $\vee_i$ *bi* $\neg_i$ *f*;*skip*) $\wedge_i$ *init w* $\equiv_i$
empty $\wedge_i$ *bs* $\neg_i$ *f* $\wedge_i$ *init w*
by (*simp add*: *bs-d-def*)
have *4*: $\vdash$ *empty* $\wedge_i$ *bs* $\neg_i$ *f* $\wedge_i$ *init w* $\equiv_i$ *empty* $\wedge_i$ *init w* $\wedge_i$ *bs* $\neg_i$ *f*
by *auto*
have *5*: $\vdash$ *empty* $\wedge_i$ *init w* $\equiv_i$ $\triangleright$ (*init w*)
using *FstState itl-prop*(*30*) **by** *blast*
hence *6*: $\vdash$ *empty* $\wedge_i$ *init w* $\wedge_i$ *bs* $\neg_i$ *f* $\equiv_i$ $\triangleright$ (*init w*) $\wedge_i$ *bs* $\neg_i$ *f*
by *auto*
have *7*: $\vdash$ $\triangleright$ (*init w*) $\wedge_i$ *bs* $\neg_i$ *f* $\supset_i$ ($\triangleright$ (*init w*) $\wedge_i$ *bs* $\neg_i$ *f*) $\vee_i$ ($\triangleright$*f* $\wedge_i$ *bs* $\neg_i$(*init w*))
by *auto*
have *8*: $\vdash$ $\triangleright$(*init w* $\vee_i$ *f*) $\equiv_i$ ($\triangleright$ (*init w*) $\wedge_i$ *bs* $\neg_i$ *f*) $\vee_i$ ($\triangleright$*f* $\wedge_i$ *bs* $\neg_i$(*init w*))
using *FstWithOrEqv* **by** *blast*
from *1 2 3 4 5 6 7 8* **show** *?thesis* **by** *auto*
qed

lemma *FstLenSame*:
($\forall$ σ. (σ $\models$ *di* ($\triangleright$*f* $\wedge_i$ *len*(*i*)) $\wedge_i$ *di* ($\triangleright$*f* $\wedge_i$ *len*(*j*))) $\longrightarrow$ (*i*=*j*))
using *FstLenSamesem DiLenFstsem* **by** (*metis and-defs*)

lemma *FstAndLenSame*:
($\forall$ σ. (σ $\models$ *di* ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*)) $\wedge_i$ *di* ($\triangleright$*f* $\wedge_i$ *g2* $\wedge_i$ *len*(*j*))) $\longrightarrow$ (*i*=*j*))
using *DiLenFstAndsem* **by** (*metis and-defs linorder-neqE-nat not-defs*)

lemma *FstLenSameChop*:
($\forall$ σ. (σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*h1* $\wedge_i$ ($\triangleright$*f* $\wedge_i$ *g2* $\wedge_i$ *len*(*j*));*h2*) $\longrightarrow$ (*i*=*j*))
proof
fix σ
show (σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*h1* $\wedge_i$ ($\triangleright$*f* $\wedge_i$ *g2* $\wedge_i$ *len*(*j*));*h2*) $\longrightarrow$ (*i*=*j*)
proof
assume *0*: (σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*h1* $\wedge_i$ ($\triangleright$*f* $\wedge_i$ *g2* $\wedge_i$ *len*(*j*));*h2*)
have *1*: (σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*h1*) **using** *0* **by** *auto*
have *2*: (σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*h1*) $\longrightarrow$
(σ $\models$ ($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*));*true*$_i$) **by** *auto*
have *3*: (σ $\models$ *di*($\triangleright$*f* $\wedge_i$ *g1* $\wedge_i$ *len*(*i*))) **using** *1 2* **by** *auto*

have *4*: $(\sigma \models\ (\triangleright f \wedge_i g2 \wedge_i len(j));h2)$ **using** *0* **by** *auto*
have *5*: $(\sigma \models\ (\triangleright f \wedge_i g2 \wedge_i len(j));h2) \longrightarrow$
$(\sigma \models\ (\triangleright f \wedge_i g2 \wedge_i len(j));true_i)$ **by** *auto*
have *6*: $(\sigma \models\ di(\triangleright f \wedge_i g2 \wedge_i len(j)))$ **using** *4 5* **by** *auto*
have *7*: $(\sigma \models\ di(\triangleright f \wedge_i g1 \wedge_i len(i)) \wedge_i di(\triangleright f \wedge_i g2 \wedge_i len(j)))$ **using** *3 6* **by** *auto*
thus $(i{=}j)$ **using** *FstAndLenSame* **by** *blast*
qed
qed

lemma *DiImpExistsOneDiLenAndFst*:
$(\forall \sigma.\ (\sigma \models di\ f) \longrightarrow (\exists!\ k.\ (\sigma \models di(\ \triangleright f \wedge_i len(k)))))$
proof
fix σ
show $(\sigma \models di\ f) \longrightarrow (\exists!\ k.\ (\sigma \models di(\ \triangleright f \wedge_i len(k))))$
proof
assume *0*: $(\sigma \models di\ f)$
have *1*: $(\sigma \models di\ (\triangleright\ f))$ **using** *0 DiEqvDiFst valid-def* **by** *auto*
have *2*: $(\sigma \models \triangleright\ f) = (\ (\sigma \models \triangleright\ f) \wedge (\exists k.\ (\sigma \models len(k))))$ **using** *AndExistsLen valid-def* **by** *auto*
have *3*: $((\sigma \models \triangleright\ f) \wedge (\exists k.\ (\sigma \models len(k)))) =$
$(\exists k.\ \ (\sigma \models \triangleright\ f) \wedge\ \ (\sigma \models len(k)))$ **by** *auto*
have *4*: $(\sigma \models di(\triangleright\ f)) = (\exists k.\ (\sigma \models di(\triangleright f \wedge_i len(k))))$ **using** *2 3 DiEqvDi* **by** *auto*
have *5*: $(\exists k.\ (\sigma \models di(\triangleright f \wedge_i len(k))))$ **using** *1* **by** *auto*
then obtain i **where** *6*: $(\sigma \models di(\triangleright f \wedge_i len(i)))$ **by** *blast*
from *5* **obtain** j **where** *7*: $(\sigma \models di(\triangleright f \wedge_i len(j)))$ **by** *blast*
have *8*: $(\sigma \models di(\triangleright f \wedge_i len(i))) \wedge (\sigma \models di(\triangleright f \wedge_i len(j)))$ **using** *6 7* **by** *auto*
hence *9*: $(\sigma \models di(\triangleright f \wedge_i len(i)) \wedge_i di(\triangleright f \wedge_i len(j)))$ **by** *simp*
hence *10*: $i{=}j$ **using** *FstLenSame* **by** *blast*
have *11*: $\bigwedge j.\ (\sigma \models di(\triangleright f \wedge_i len(j))) \longrightarrow (j{=}i)$ **using** *9 10* **using** *FstLenSame and-defs* **by** *blast*
thus $(\exists!\ k.\ (\sigma \models di(\ \triangleright f \wedge_i len(k)\ )))$ **using** *11 5* **by** *blast*
qed
qed

lemma *LFstAndDist-help*:
$(\sigma \models ((\triangleright f \wedge_i g1) \wedge_i len(k));h1\ \ \wedge_i ((\triangleright f \wedge_i g2) \wedge_i len(k));h2) =$
$(\sigma \models ((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2)\ )$
using *LFixedAndDistr* **using** *itl-eq* **by** *blast*

lemma *LFstAndDist-help-1*:
$(\exists\ k.\ (\sigma \models ((\triangleright f \wedge_i g1) \wedge_i len(k));h1\ \ \wedge_i ((\triangleright f \wedge_i g2) \wedge_i len(k));h2)) =$
$(\exists\ k.\ (\sigma \models ((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2)\ ))$
proof
assume *0*: $\exists k.\ \sigma \models\ ((\triangleright\ f \wedge_i g1) \wedge_i len\ k)\ ;\ h1\ \ \wedge_i\ \ ((\triangleright\ f \wedge_i g2) \wedge_i len\ k)\ ;\ h2$
obtain k **where** *1*: $\sigma \models\ ((\triangleright\ f \wedge_i g1) \wedge_i len\ k)\ ;\ h1\ \ \wedge_i\ \ ((\triangleright\ f \wedge_i g2) \wedge_i len\ k)\ ;\ h2$
using *0* **by** *auto*
hence *2*: $(\sigma \models ((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2))$
using *LFstAndDist-help* **by** *blast*
show $(\exists\ k.\ (\sigma \models ((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2)\ ))$
using *2* **by** *auto*
next
assume *3*: $(\exists\ k.\ (\sigma \models ((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2)\ ))$

obtain *k* **where** *4*: $(\sigma \models((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2))$
using *3* **by** *auto*
hence *5*: $(\sigma \models((\triangleright f \wedge_i g1) \wedge_i len(k));h1 \ \wedge_i ((\triangleright f \wedge_i g2) \wedge_i len(k));h2)$
using *LFstAndDist-help* **by** *blast*
show $(\exists\ k.\ (\sigma \models((\triangleright f \wedge_i g1) \wedge_i len(k));h1 \ \wedge_i ((\triangleright f \wedge_i g2) \wedge_i len(k));h2))$
using *5* **by** *auto*
qed

lemma *LFstAndDistrsem*:
$(\forall\ \sigma.\ (\sigma \models(\triangleright f \wedge_i g1);h1 \wedge_i (\triangleright f \wedge_i g2);h2 \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);(h1 \wedge_i h2)))$
proof
fix σ
show $(\sigma \models(\triangleright f \wedge_i g1);h1 \wedge_i (\triangleright f \wedge_i g2);h2 \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);(h1 \wedge_i h2))$
proof $-$
have *1*: $(\sigma \models(\triangleright f \wedge_i g1);h1) = (\exists\ i.\ (\sigma \models(\triangleright f \wedge_i g1 \wedge_i len(i));h1))$
using *AndExistsLenChop* **by** *auto*
have *2*: $(\sigma \models(\triangleright f \wedge_i g2);h2) = (\exists\ j.\ (\sigma \models(\triangleright f \wedge_i g2 \wedge_i len(j));h2))$
using *AndExistsLenChop* **by** *auto*
have *3*: $(\sigma \models(\triangleright f \wedge_i g1);h1 \wedge_i (\triangleright f \wedge_i g2);h2) =$
$((\exists\ i\, j.\ (\sigma \models(\triangleright f \wedge_i g1 \wedge_i len(i));h1 \ \wedge_i$
$(\triangleright f \wedge_i g2 \wedge_i len(j));h2)$
$)$
using *1 2* **by** *auto*
have *4*: $((\exists\ i\, j.\ (\sigma \models(\triangleright f \wedge_i g1 \wedge_i len(i));h1 \ \wedge_i$
$(\triangleright f \wedge_i g2 \wedge_i len(j));h2)$
$) =$
$((\exists\ k.\ (\sigma \models(\triangleright f \wedge_i g1 \wedge_i len(k));h1 \ \wedge_i$
$(\triangleright f \wedge_i g2 \wedge_i len(k));h2)$
$)$
using *FstLenSameChop* **by** *blast*
have *5*: $(\exists\ k.\ (\sigma \models((\triangleright f \wedge_i g1) \wedge_i len(k));h1 \ \wedge_i ((\triangleright f \wedge_i g2) \wedge_i len(k));h2)) =$
$(\exists\ k.\ (\sigma \models((\triangleright f \wedge_i g1) \wedge_i (\triangleright f \wedge_i g2) \wedge_i len(k));(h1 \wedge_i h2)))$
using *LFstAndDist-help-1* **by** *blast*
have *6* : $(\exists\ k.\ (\sigma \models(\triangleright f \wedge_i g1 \wedge_i \triangleright f \wedge_i g2 \wedge_i len(k));(h1 \wedge_i h2))) =$
$(\sigma \models(\triangleright f \wedge_i g1 \wedge_i \triangleright f \wedge_i g2);(h1 \wedge_i h2))$
using *AndExistsLenChop* **by** *auto*
have *7* : $(\sigma \models(\triangleright f \wedge_i g1 \wedge_i \triangleright f \wedge_i g2);(h1 \wedge_i h2)) =$
$(\sigma \models(\triangleright f \wedge_i g1 \wedge_i g2);(h1 \wedge_i h2))$
by *auto*
from *3 4 5 6 7* **show** *?thesis* **by** *auto*
qed
qed

lemma *LFstAndDistr*:
$\vdash (\triangleright f \wedge_i g1);h1 \wedge_i (\triangleright f \wedge_i g2);h2 \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);(h1 \wedge_i h2)$
using *LFstAndDistrsem* **by** *simp*

lemma *LFstAndDistrA*:
$\vdash (\triangleright f \wedge_i g1);h \wedge_i (\triangleright f \wedge_i g2);h \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);h$
proof $-$

have *1*: $\vdash (\triangleright f \wedge_i g1);h \wedge_i (\triangleright f \wedge_i g2);h \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);(h \wedge_i h)$ **using** *LFstAndDistr* **by** *blast*
have *2*: $\vdash (\triangleright f \wedge_i g1 \wedge_i g2);(h \wedge_i h) \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);h$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *LFstAndDistrB*:
$\vdash (\triangleright f \wedge_i g);h1 \wedge_i (\triangleright f \wedge_i g);h2 \equiv_i (\triangleright f \wedge_i g);(h1 \wedge_i h2)$
proof –
have *1*: $\vdash (\triangleright f \wedge_i g);h1 \wedge_i (\triangleright f \wedge_i g);h2 \equiv_i (\triangleright f \wedge_i g \wedge_i g);(h1 \wedge_i h2)$ **using** *LFstAndDistr* **by** *blast*
have *2*: $\vdash (\triangleright f \wedge_i g \wedge_i g);(h1 \wedge_i h2) \equiv_i (\triangleright f \wedge_i g);(h1 \wedge_i h2)$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *LFstAndDistrC*:
$\vdash (\triangleright f\ );h1 \wedge_i (\triangleright f\ );h2 \equiv_i (\triangleright f);(h1 \wedge_i h2)$
proof –
have *1*: $\vdash (\triangleright f \wedge_i true_i);h1 \wedge_i (\triangleright f \wedge_i true_i);h2 \equiv_i (\triangleright f \wedge_i true_i \wedge_i true_i);(h1 \wedge_i h2)$
using *LFstAndDistr* **by** *blast*
have *2*: $\vdash (\triangleright f \wedge_i true_i);h1 \equiv_i (\triangleright f\ );h1$
by *auto*
have *3*: $\vdash (\triangleright f \wedge_i true_i);h2 \equiv_i (\triangleright f\ );h2$
by *auto*
have *4*: $\vdash (\triangleright f \wedge_i true_i \wedge_i true_i);(h1 \wedge_i h2) \equiv_i (\triangleright f);(h1 \wedge_i h2)$
by *auto*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *LFstAndDistrD*:
$\vdash di(\triangleright f \wedge_i g1) \wedge_i di(\triangleright f \wedge_i g2) \equiv_i di(\triangleright f \wedge_i g1 \wedge_i g2)$
proof –
have *1*: $\vdash (\triangleright f \wedge_i g1);true_i \wedge_i (\triangleright f \wedge_i g2);true_i \equiv_i (\triangleright f \wedge_i g1 \wedge_i g2);(true_i \wedge_i true_i)$
using *LFstAndDistr* **by** *blast*
have *2*: $\vdash (\triangleright f \wedge_i g1);true_i \equiv_i di(\triangleright f \wedge_i g1)$
by (*simp add*: *di-d-def*)
have *3*: $\vdash (\triangleright f \wedge_i g2);true_i \equiv_i di(\triangleright f \wedge_i g2)$
by (*simp add*: *di-d-def*)
have *4*: $\vdash (\triangleright f \wedge_i g1 \wedge_i g2);(true_i \wedge_i true_i) \equiv_i di(\triangleright f \wedge_i g1 \wedge_i g2)$
by (*simp add*: *di-d-def*)
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *LstAndDistr*:
$\vdash h1;(\triangleleft f \wedge_i g1) \wedge_i h2;(\triangleleft f \wedge_i g2) \equiv_i (h1 \wedge_i h2);(\triangleleft f \wedge_i g1 \wedge_i g2)$
proof –
have *1*: $\vdash (\triangleright(f^r) \wedge_i g1^r);(h1^r) \wedge_i (\triangleright(f^r) \wedge_i (g2^r));(h2^r) \equiv_i$
$(\triangleright(f^r) \wedge_i (g1^r) \wedge_i (g2^r));((h1^r) \wedge_i (h2^r))$ **using** *LFstAndDistr* **by** *blast*
hence *2*: $\vdash ((\triangleright(f^r) \wedge_i g1^r);(h1^r) \wedge_i (\triangleright(f^r) \wedge_i (g2^r));(h2^r))^r \equiv_i$
$((\triangleright(f^r) \wedge_i (g1^r) \wedge_i (g2^r));((h1^r) \wedge_i (h2^r)))^r$ **using** *1 REqvRule* **by** *blast*
have *3*: $\vdash ((\triangleright(f^r) \wedge_i g1^r);(h1^r) \wedge_i (\triangleright(f^r) \wedge_i (g2^r));(h2^r))^r \equiv_i$

$((\triangleright(f^r) \wedge_i g1^r);(h1^r))^r \wedge_i ((\triangleright(f^r) \wedge_i (g2^r));(h2^r))^r$
using *RAnd* **by** *blast*
have *4*: $\vdash ((\triangleright(f^r) \wedge_i g1^r);(h1^r))^r \wedge_i ((\triangleright(f^r) \wedge_i (g2^r));(h2^r))^r \equiv_i$
$(h1^r)^r;(\triangleright(f^r) \wedge_i g1^r)^r \wedge_i (h2^r)^r;(\triangleright(f^r) \wedge_i (g2^r))^r$
by *simp*
have *5*: $\vdash (h1^r)^r \equiv_i h1$ **using** *EqvReverseReverse itl-prop(30)* **by** *blast*
have *6*: $\vdash (h2^r)^r \equiv_i h2$ **using** *EqvReverseReverse itl-prop(30)* **by** *blast*
have *7*: $\vdash (g1^r)^r \equiv_i g1$ **using** *EqvReverseReverse itl-prop(30)* **by** *blast*
have *8*: $\vdash (g2^r)^r \equiv_i g2$ **using** *EqvReverseReverse itl-prop(30)* **by** *blast*
have *9*: $\vdash (f^r)^r \equiv_i f$**using** *EqvReverseReverse itl-prop(30)* **by** *blast*
have *10*: $\vdash (\triangleright(f^r) \wedge_i g1^r)^r \equiv_i (\triangleright(f^r))^r \wedge_i (g1^r)^r$ **using** *RAnd* **by** *blast*
have *11*: $\vdash (\triangleright(f^r) \wedge_i g2^r)^r \equiv_i (\triangleright(f^r))^r \wedge_i (g2^r)^r$ **using** *RAnd* **by** *blast*
have *12*: $\vdash (\triangleright(f^r))^r \equiv_i \triangleleft (f)$ **using** *RRFirstEqvLast* **by** *blast*
have *13*: $\vdash (\triangleright(f^r))^r \wedge_i (g1^r)^r \equiv_i \triangleleft f \wedge_i g1$ **using** *12 7* **by** *auto*
have *14*: $\vdash (\triangleright(f^r))^r \wedge_i (g2^r)^r \equiv_i \triangleleft f \wedge_i g2$ **using** *12 8* **by** *auto*
have *15*: $\vdash (h1^r)^r;(\triangleright(f^r) \wedge_i g1^r)^r \wedge_i (h2^r)^r;(\triangleright(f^r) \wedge_i (g2^r))^r \equiv_i$
$h1;(\triangleleft f \wedge_i g1) \wedge_i h2;(\triangleleft f \wedge_i g2)$ **using** *14 13 10 11 5 6* **by** *auto*
have *16*: $\vdash ((\triangleright(f^r) \wedge_i (g1^r) \wedge_i (g2^r));((h1^r) \wedge_i (h2^r)))^r \equiv_i$
$((h1^r) \wedge_i (h2^r))^r;((\triangleright (f^r)) \wedge_i (g1^r) \wedge_i (g2^r))^r$ **by** *simp*
have *17*: $\vdash ((\triangleright (f^r)) \wedge_i (g1^r) \wedge_i (g2)^r)^r \equiv_i ((\triangleright (f^r))^r \wedge_i (g1^r)^r \wedge_i (g2^r)^r)$ **using** *RAnd* **by** *auto*
have *18*: $\vdash ((\triangleright (f^r))^r \wedge_i (g1^r)^r \wedge_i (g2^r)^r) \equiv_i \triangleleft f \wedge_i g1 \wedge_i g2$ **using** *12 7 8* **by** *auto*
have *19*: $\vdash ((h1^r) \wedge_i (h2^r))^r \equiv_i h1 \wedge_i h2$ **using** *RRAnd* **by** *auto*
have *20*: $\vdash ((h1^r) \wedge_i (h2^r))^r;((\triangleright (f^r)) \wedge_i (g1^r) \wedge_i (g2^r))^r \equiv_i$
$(h1 \wedge_i h2);(\triangleleft f \wedge_i g1 \wedge_i g2)$ **using** *19 17 18* **by** *auto*
from *20 15 1 2 3 4* **show** *?thesis* **by** *simp*
qed

lemma *LstAndDistrA*:
$\vdash h;(\triangleleft f \wedge_i g1) \wedge_i h;(\triangleleft f \wedge_i g2) \equiv_i h;(\triangleleft f \wedge_i g1 \wedge_i g2)$
proof –
have *1*: $\vdash h;(\triangleleft f \wedge_i g1) \wedge_i h;(\triangleleft f \wedge_i g2) \equiv_i (h \wedge_i h);(\triangleleft f \wedge_i g1 \wedge_i g2)$
using *LstAndDistr* **by** *blast*
have *2*: $\vdash (h \wedge_i h);(\triangleleft f \wedge_i g1 \wedge_i g2) \equiv_i h;(\triangleleft f \wedge_i g1 \wedge_i g2)$
by *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *LstAndDistrB*:
$\vdash h1;(\triangleleft f \wedge_i g) \wedge_i h2;(\triangleleft f \wedge_i g) \equiv_i (h1 \wedge_i h2);(\triangleleft f \wedge_i g)$
proof –
have *1*: $\vdash h1;(\triangleleft f \wedge_i g) \wedge_i h2;(\triangleleft f \wedge_i g) \equiv_i (h1 \wedge_i h2);(\triangleleft f \wedge_i g \wedge_i g)$
using *LstAndDistr* **by** *blast*
have *2*: $\vdash (h1 \wedge_i h2);(\triangleleft f \wedge_i g \wedge_i g) \equiv_i (h1 \wedge_i h2);(\triangleleft f \wedge_i g)$
by *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *LstAndDistrC*:
$\vdash h1;(\triangleleft f) \wedge_i h2;(\triangleleft f) \equiv_i (h1 \wedge_i h2);(\triangleleft f)$
proof –

have *1*: $\vdash h1;(\lhd\ f \wedge_i true_i) \wedge_i h2;(\lhd\ f \wedge_i true_i) \equiv_i (h1 \wedge_i h2);(\lhd\ f \wedge_i true_i \wedge_i true_i\ )$
  using *LstAndDistr* **by** *blast*
have *2*: $\vdash (h1 \wedge_i h2);(\lhd\ f \wedge_i true_i \wedge_i true_i\ ) \equiv_i (h1 \wedge_i h2);(\lhd\ f\ )$
  by *auto*
have *3*: $\vdash h1;(\lhd\ f \wedge_i true_i) \equiv_i h1;(\lhd\ f\ )$
  by *auto*
have *4*: $\vdash h2;(\lhd\ f \wedge_i true_i) \equiv_i h2;(\lhd\ f\ )$
  by *auto*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *LstAndDistrD*:
$\vdash\ \Diamond(\lhd\ f \wedge_i g1) \wedge_i \Diamond(\lhd\ f \wedge_i g2) \equiv_i \Diamond(\lhd\ f \wedge_i g1 \wedge_i g2)$
proof −
have *1*: $\vdash true_i;(\lhd\ f \wedge_i g1) \wedge_i true_i;(\lhd\ f \wedge_i g2) \equiv_i (true_i \wedge_i true_i\ );(\lhd\ f \wedge_i g1 \wedge_i g2)$
  using *LstAndDistr* **by** *blast*
have *2*: $\vdash (true_i \wedge_i true_i\ );(\lhd\ f \wedge_i g1 \wedge_i g2) \equiv_i \Diamond(\lhd\ f \wedge_i g1 \wedge_i g2)$
  by (*simp add*: *sometimes-d-def*)
have *3*: $\vdash true_i;(\lhd\ f \wedge_i g1) \equiv_i \Diamond(\lhd\ f \wedge_i g1)$
  by (*simp add*: *sometimes-d-def*)
have *4*: $\vdash true_i;(\lhd\ f \wedge_i g2) \equiv_i \Diamond(\lhd\ f \wedge_i g2)$
  by (*simp add*: *sometimes-d-def*)
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *NotFstChop*:
$\vdash \neg_i(\rhd f\ ;g) \equiv_i \neg_i(di\ (\rhd f)) \vee_i (\rhd f;\neg_i g)$
proof −
have *1*: $\vdash g \supset_i true_i$ **by** *auto*
hence *2*: $\vdash \rhd f;g \supset_i \rhd f;true_i$ **using** *RightChopImpChop* **by** *blast*
hence *3*: $\vdash \rhd f;g \supset_i di(\rhd f)$ **by** (*simp add*:*di-d-def*)
hence *4*: $\vdash \neg_i(di(\rhd f)) \supset_i \neg_i(\rhd f;g)$ **by** *auto*
have *5*: $\vdash (\rhd f;\ \neg_i g \supset_i \neg_i(\rhd f;g)) \equiv_i ((\rhd f;\ \neg_i g) \wedge_i (\rhd f;g) \supset_i false_i\ )$ **by** *auto*
have *6*: $\vdash (\rhd f;\ \neg_i g) \wedge_i (\rhd f;g) \equiv_i \rhd f;(\neg_i g \wedge_i g)$ **using** *LFstAndDistrC* **by** *blast*
have *7*: $\vdash \neg_i(\rhd f;(\neg_i g \wedge_i g))$ **by** *auto*
have *8*: $\vdash \rhd f;\ \neg_i g \supset_i \neg_i(\rhd f;g)$ **using** *5 6 7* **by** *auto*
have *9*: $\vdash \neg_i(di(\rhd f)) \vee_i (\rhd f;\ \neg_i g) \supset_i \neg_i(\rhd f;g)$ **using** *4 8* **by** *auto*
have *10*: $\vdash di(\rhd f) \vee_i \neg_i(di(\rhd f))$ **by** *auto*
hence *11*: $\vdash (\rhd f;true_i) \vee_i \neg_i(di(\rhd f))$ **by** (*simp add*: *di-d-def*)
hence *12*: $\vdash (\rhd f;(g \vee_i \neg_i g)) \vee_i \neg_i(di(\rhd f))$ **by** *auto*
have *13*: $\vdash (\rhd f;(g \vee_i \neg_i g)) \equiv_i (\rhd f;g) \vee_i (\rhd f;\neg_i g)$ **using** *ChopOrEqv* **by** *auto*
have *14*: $\vdash ((\rhd f;g) \vee_i (\rhd f;\neg_i g)) \vee_i \neg_i(di(\rhd f))$ **using** *12 13* **by** *auto*
hence *15*: $\vdash \neg_i(\rhd f;g) \supset_i \neg_i(di(\rhd f)) \vee_i (\rhd f;\ \neg_i g)$ **by** *auto*
from *9 15* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *BsNotFstChop*:
$\vdash bs(\neg_i(\rhd f;g)) \equiv_i empty \vee_i \neg_i(di(\rhd f)) \vee_i (\rhd f;bs\neg_i g)$
proof −
have *1*: $\vdash bs(\neg_i(\rhd f;g)) \equiv_i empty \vee_i bi\neg_i(\rhd f;g);skip$

by (*simp add*:*bs-d-def*)
have *2*: $\vdash empty \lor_i bi\neg_i(\triangleright f;g);skip \equiv_i empty \lor_i \neg_i(di(\triangleright f;g));skip$
by *auto*
have *3*: $\vdash empty \lor_i \neg_i(di(\triangleright f;g));skip \equiv_i empty \lor_i \neg_i((\triangleright f;g);true_i);skip$
by *auto*
have *4*: $\vdash \neg_i((\triangleright f;g);true_i);skip \equiv_i \neg_i(\triangleright f;(g;true_i));skip$
using *ChopAssocB* **using** *LeftChopEqvChop itl-prop*(*33*) **by** *blast*
hence *5*: $\vdash empty \lor_i \neg_i((\triangleright f;g);true_i);skip \equiv_i empty \lor_i \neg_i(\triangleright f;(g;true_i));skip$
by *auto*
have *6*: $\vdash empty \lor_i \neg_i(\triangleright f;(g;true_i));skip \equiv_i empty \lor_i \neg_i(\triangleright f;di(g));skip$
by (*simp add*: *di-d-def*)
have *7*: $\vdash empty \lor_i \neg_i(\triangleright f;di(g));skip \equiv_i empty \lor_i \neg_i(\neg_i(\neg_i(\triangleright f;di(g));skip))$
by *auto*
have *8*: $\vdash \neg_i(\neg_i(\neg_i(\triangleright f;di(g));skip)) \equiv_i \neg_i(empty \lor_i (\triangleright f;di(g));skip)$
using *NotNotChopSkip* **using** *prop01* **by** *blast*
hence *9*: $\vdash empty \lor_i \neg_i(\neg_i(\neg_i(\triangleright f;di(g));skip)) \equiv_i empty \lor_i \neg_i(empty \lor_i (\triangleright f;di(g));skip)$
by *auto*
have *10*: $\vdash empty \lor_i \neg_i(empty \lor_i (\triangleright f;di(g));skip) \equiv_i empty \lor_i (more \land_i \neg_i((\triangleright f;di(g));skip))$
by *auto*
have *11*: $\vdash empty \lor_i (more \land_i \neg_i((\triangleright f;di(g));skip)) \equiv_i empty \lor_i \neg_i((\triangleright f;di(g));skip)$
by *auto*
have *12*: $\vdash empty \lor_i \neg_i((\triangleright f;di(g));skip) \equiv_i empty \lor_i \neg_i(\triangleright f;(di(g);skip))$
using *ChopAssocB 11 itl-prop*(*30*) *prop01 prop03 prop28* **by** *blast*
have *13*: $\vdash \neg_i(\triangleright f;(di(g);skip)) \equiv_i \neg_i(\triangleright f;(ds(g)))$
using *DsDi* **using** *RightChopEqvChop itl-prop*(*30*) *itl-prop*(*33*) **by** *blast*
hence *14*: $\vdash empty \lor_i \neg_i(\triangleright f;(di(g);skip)) \equiv_i empty \lor_i \neg_i(\triangleright f;(ds(g)))$
by *auto*
have *15*: $\vdash empty \lor_i \neg_i(\triangleright f;(ds(g))) \equiv_i empty \lor_i \neg_i(di\ (\triangleright f)) \lor_i (\triangleright f;\neg_i(ds\ g))$
using *NotFstChop* **by** *auto*
have *16*: $\vdash (\triangleright f;\neg_i(ds\ g)) \equiv_i (\triangleright f;(bs\ \neg_i g))$
using *NotDsEqvBsNot RightChopEqvChop* **by** *blast*
hence *17*: $\vdash (empty \lor_i \neg_i(di\ (\triangleright f))) \lor_i (\triangleright f;\neg_i(ds\ g)) \equiv_i (empty \lor_i \neg_i(di\ (\triangleright f))) \lor_i (\triangleright f;(bs\ \neg_i g))$
by *auto*
from *1 2 3 5 6 7 9 10 11 12 14 15 17* **show** *?thesis* **by** *auto*
qed

lemma *FstFstChopEqvFstChopFst*:
$\vdash \triangleright(\triangleright f;g) \equiv_i \triangleright f;\triangleright g$
proof −
have *1*: $\vdash \triangleright(\triangleright f;g) \equiv_i (\triangleright f;g) \land_i bs\ \neg_i(\triangleright f;g)$
by (*simp add*: *first-d-def*)
have *2*: $\vdash bs\ \neg_i(\triangleright f;g) \equiv_i empty \lor_i \neg_i(di(\triangleright f)) \lor_i (\triangleright f;bs\neg_i g)$
using *BsNotFstChop* **by** *auto*
hence *3*: $\vdash (\triangleright f;g) \land_i bs\ \neg_i(\triangleright f;g) \equiv_i (\triangleright f;g) \land_i (empty \lor_i \neg_i(di(\triangleright f)) \lor_i (\triangleright f;bs\neg_i g))$
by *auto*
have *4*: $\vdash (\triangleright f;g) \land_i (empty \lor_i \neg_i(di(\triangleright f)) \lor_i (\triangleright f;bs\neg_i g)) \equiv_i$
$((\triangleright f;g) \land_i empty) \lor_i ((\triangleright f;g) \land_i \neg_i(di(\triangleright f))) \lor_i ((\triangleright f;g) \land_i (\triangleright f;bs\neg_i g))$
by *auto*
have *5*: $\vdash \neg_i((\triangleright f;g) \land_i \neg_i(di(\triangleright f)))$
by *auto*

hence *6*: $\vdash ((\triangleright f;g) \wedge_i \textit{empty}) \vee_i ((\triangleright f;g) \wedge_i \neg_i(di(\triangleright f))) \vee_i ((\triangleright f;g) \wedge_i (\triangleright f;bs\neg_i g)) \equiv_i$
$((\triangleright f;g) \wedge_i \textit{empty}) \vee_i ((\triangleright f;g) \wedge_i (\triangleright f;bs\neg_i g))$
by *auto*
have *7*: $\vdash ((\triangleright f;g) \wedge_i (\triangleright f;(bs\neg_i g))) \equiv_i ((\triangleright f;(g \wedge_i (bs\neg_i g))))$
using *LFstAndDistrC* **by** *blast*
hence *8*: $\vdash ((\triangleright f;g) \wedge_i \textit{empty}) \vee_i ((\triangleright f;g) \wedge_i (\triangleright f;(bs\neg_i g))) \equiv_i$
$((\triangleright f;g) \wedge_i \textit{empty}) \vee_i ((\triangleright f;(g \wedge_i (bs\neg_i g))))$
by *auto*
have *9*: $\vdash ((\triangleright f;g) \wedge_i \textit{empty}) \vee_i ((\triangleright f;(g \wedge_i (bs\neg_i g)))) \equiv_i ((\triangleright f;g) \wedge_i \textit{empty}) \vee_i \triangleright f;\triangleright g$
by (*simp add*: *first-d-def*)
have *10*: $\vdash ((\triangleright f;g) \wedge_i \textit{empty}) \equiv_i ((\triangleright f;\triangleright g) \wedge_i \textit{empty})$
using *FstChopEmptyEqvFstChopFstEmpty* **by** *blast*
hence *11*: $\vdash ((\triangleright f;g) \wedge_i \textit{empty}) \vee_i \triangleright f;\triangleright g \equiv_i ((\triangleright f;\triangleright g) \wedge_i \textit{empty}) \vee_i \triangleright f;\triangleright g$
by *auto*
have *12*: $\vdash ((\triangleright f;\triangleright g) \wedge_i \textit{empty}) \vee_i \triangleright f;\triangleright g \equiv_i \triangleright f;\triangleright g$
by *auto*
from *1 3 4 6 8 9 11 12* **show** *?thesis* **by** *auto*
qed

lemma *FstFixFst*:
$\vdash \triangleright(\triangleright f) \equiv_i \triangleright f$
proof −
have *1*: $\vdash \triangleright f \equiv_i (\triangleright f);\textit{empty}$ **using** *ChopEmpty* **using** *itl-prop*(*30*) **by** *blast*
hence *2*: $\vdash \triangleright(\triangleright f) \equiv_i \triangleright((\triangleright f);\textit{empty})$ **using** *FstEqvRule* **by** *blast*
have *3*: $\vdash \triangleright((\triangleright f);\textit{empty}) \equiv_i \triangleright f;\triangleright \textit{empty}$ **using** *FstFstChopEqvFstChopFst* **by** *auto*
have *4*: $\vdash \triangleright f;\triangleright \textit{empty} \equiv_i \triangleright f;\textit{empty}$ **using** *FstEmpty* **by** *auto*
have *5*: $\vdash \triangleright f;\textit{empty} \equiv_i \triangleright f$ **using** *ChopEmpty* **by** *blast*
from *2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *FstCSEqvEmpty*:
$\vdash \triangleright(f^\star) \equiv_i \textit{empty}$
proof −
have *1*: $\vdash \triangleright(f^\star) \equiv_i \triangleright(\textit{empty} \vee_i ((f \wedge_i \textit{more});f^\star))$ **using** *ChopstarEqv FstEqvRule* **by** *blast*
from *1* **show** *?thesis* **using** *FstEmptyOrEqvEmpty* **by** *auto*
qed

lemma *FstIterFixFst*:
$\vdash \textit{power}\ (\triangleright\ f)\ n \equiv_i \triangleright(\textit{power}\ (\triangleright\ f)\ n)$
proof
(*induct n*)
case *0*
then show *?case*
proof −
have *1*: $\vdash \textit{power}\ (\triangleright\ f)\ 0 \equiv_i \textit{empty}$ **by** *auto*
have *2*: $\vdash \textit{empty} \equiv_i \triangleright\ \textit{empty}$ **using** *FstEmpty* **by** *auto*
have *3*: $\vdash \triangleright\ \textit{empty} \equiv_i \triangleright(\textit{power}\ (\triangleright f)\ 0)$ **by** *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed
next

case (*Suc n*)
then show *?case*
proof −
have *4*: $\vdash$ (*power* ($\triangleright$ *f*) (*Suc n*)) $\equiv_i$ ($\triangleright$ *f*) ;(*power* ($\triangleright$*f*) *n*)
using *pow-Suc* **by** *simp*
have *5*: $\vdash$ ($\triangleright$ *f*) ;(*power* ($\triangleright$*f*) *n*) $\equiv_i$ ($\triangleright$ *f*) ; $\triangleright$ (*power* ($\triangleright$ *f*) *n*)
using *RightChopEqvChop Suc.hyps* **by** *blast*
have *6*: $\vdash$ ($\triangleright$ *f*) ; $\triangleright$ (*power* ($\triangleright$ *f*) *n*) $\equiv_i$ $\triangleright$($\triangleright$*f*;(*power* ($\triangleright$ *f*) *n*))
using *FstFstChopEqvFstChopFst* **by** *auto*
have *7*: $\vdash$ $\triangleright$($\triangleright$*f*;(*power* ($\triangleright$ *f*) *n*)) $\equiv_i$ $\triangleright$(*power* ($\triangleright$ *f*) (*Suc n*))
using *pow-Suc* **by** *simp*
from *4 5 6 7* **show** *?thesis* **by** *auto*
qed
qed

lemma *DsImpNotFst*:
$\vdash$ *ds f* $\supset_i$ ($\neg_i$($\triangleright$*f*))
proof −
have *1*: $\vdash$ *ds f* $\wedge_i$ $\triangleright$*f* $\equiv_i$ *ds f* $\wedge_i$ (*f* $\wedge_i$ *bs* $\neg_i$ *f*) **by** (*simp add*: *first-d-def*)
have *2*: $\vdash$ *ds f* $\wedge_i$ (*f* $\wedge_i$ *bs* $\neg_i$ *f*) $\equiv_i$ *ds f* $\wedge_i$ *f* $\wedge_i$ $\neg_i$(*ds f*) **using** *NotDsEqvBsNot* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *LFixedAndDistrB1*:
$\vdash$ *len*(*k*);*f* $\wedge_i$ *len*(*k*);*g* $\equiv_i$ *len*(*k*);(*f* $\wedge_i$ *g*)
proof −
have *1*: $\vdash$ *len*(*k*);*f* $\equiv_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));*f*
by *auto*
have *2*: $\vdash$ *len*(*k*);*g* $\equiv_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));*g*
by *auto*
have *3*: $\vdash$ *len*(*k*);*f* $\wedge_i$ *len*(*k*);*g* $\equiv_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));*f* $\wedge_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));*g*
using *1 2* **by** *auto*
have *4*: $\vdash$ (*true*$_i$ $\wedge_i$ *len*(*k*));*f* $\wedge_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));*g* $\equiv_i$ (*true*$_i$ $\wedge_i$ *len*(*k*));(*f* $\wedge_i$ *g*)
using *LFixedAndDistrB* **by** *blast*
have *5*: $\vdash$ (*true*$_i$ $\wedge_i$ *len*(*k*));(*f* $\wedge_i$ *g*) $\equiv_i$ (*len*(*k*));(*f* $\wedge_i$ *g*)
by *auto*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *FstLenAndEqvLenAnd*:
$\vdash$ $\triangleright$(*len*(*k*) $\wedge_i$ *f*) $\equiv_i$ *len*(*k*) $\wedge_i$ *f*
proof −
have *1*: $\vdash$ *len*(*k*) $\wedge_i$ *f* $\wedge_i$ *ds*(*len*(*k*)$\wedge_i$ *f*) $\supset_i$ *ds* (*len*(*k*))
using *DsAndImpElimL* **by** *auto*
hence *2*: $\vdash$ *len*(*k*) $\wedge_i$ *f* $\wedge_i$ *ds*(*len*(*k*)$\wedge_i$ *f*) $\supset_i$ (*di*(*len*(*k*)));*skip*
using *DsDi itl-prop*(*31*) *prop02* **by** *blast*
hence *3*: $\vdash$ *len*(*k*) $\wedge_i$ *f* $\wedge_i$ *ds*(*len*(*k*)$\wedge_i$ *f*) $\supset_i$ ((*len*(*k*);*true*$_i$));*skip*
by (*simp add*: *di-d-def*)
hence *4*: $\vdash$ *len*(*k*) $\wedge_i$ *f* $\wedge_i$ *ds*(*len*(*k*)$\wedge_i$ *f*) $\supset_i$ (*len*(*k*);(*true*$_i$;*skip*))
using *ChopAssocB itl-prop*(*31*) *prop02* **by** *blast*

hence *5*: $\vdash$ *len*(k) $\wedge_i$ f $\wedge_i$ *ds*(*len*(k)$\wedge_i$ f) $\supset_i$ (*len*(k);(*skip*;*true*$_i$))
  using *SkipTrueEqvTrueSkip TrueChopSkipEqvSkipChopTrue RightChopEqvChop itl-prop*(*31*)
    prop02 **by** *blast*
hence *6*: $\vdash$ *len*(k) $\wedge_i$ f $\wedge_i$ *ds*(*len*(k)$\wedge_i$ f) $\supset_i$ (*len*(k);(*skip*;*true*$_i$)) $\wedge_i$ *len*(k)
  by *auto*
hence *7*: $\vdash$ *len*(k) $\wedge_i$ f $\wedge_i$ *ds*(*len*(k)$\wedge_i$ f) $\supset_i$ (*len*(k);(*skip*;*true*$_i$)) $\wedge_i$ *len*(k);*empty*
  using *ChopEmpty* **by** (*metis itl-prop*(*31*) *itl-prop*(*32*) *prop02*)
hence *8*: $\vdash$ *len*(k) $\wedge_i$ f $\wedge_i$ *ds*(*len*(k)$\wedge_i$ f) $\supset_i$ (*len*(k);((*skip*;*true*$_i$) $\wedge_i$ *empty*))
  using *LFixedAndDistrB1* **using** *itl-prop*(*31*) *prop02* **by** *blast*
have *9*: $\vdash$ $\neg_i$(*len*(k);((*skip*;*true*$_i$) $\wedge_i$ *empty*))
  by *auto*
have *10*: $\vdash$ *len*(k) $\wedge_i$ f $\supset_i$ $\neg_i$(*ds*(*len*(k)$\wedge_i$ f))
  using *8 9* **by** *auto*
hence *11*: $\vdash$ *len*(k) $\wedge_i$ f $\supset_i$ *bs* $\neg_i$(*len*(k)$\wedge_i$ f)
  using *NotDsEqvBsNot* **by** *auto*
hence *12*: $\vdash$ *len*(k) $\wedge_i$ f $\supset_i$ *len*(k) $\wedge_i$ f $\wedge_i$ *bs* $\neg_i$(*len*(k)$\wedge_i$ f)
  by *auto*
hence *13*: $\vdash$ *len*(k) $\wedge_i$ f $\supset_i$ $\rhd$(*len*(k) $\wedge_i$ f)
  by (*simp add*: *first-d-def*)
have *14*: $\vdash$ $\rhd$(*len*(k) $\wedge_i$ f) $\supset_i$ *len*(k) $\wedge_i$ f
  by (*simp add*: *first-d-def*)
from *13 14* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *FstAndElimL*:
$\vdash$ $\rhd f$ $\supset_i$ f
by (*simp add*: *first-d-def*)

lemma *FstImpNotDiChopSkip*:
$\vdash$ $\rhd f$ $\supset_i$ $\neg_i$(*di* f;*skip*)
proof −
have *1*: $\vdash$ $\rhd f$ $\supset_i$ *bs* $\neg_i$ f **by** (*simp add*: *first-d-def*)
hence *2*: $\vdash$ $\rhd f$ $\supset_i$ $\neg_i$(*ds* f) **using** *NotDsEqvBsNot* **by** *auto*
have *3*: $\vdash$ *ds* f $\equiv_i$ *di* f ; *skip* **using** *DsDi* **by** *blast*
hence *4*: $\vdash$ $\neg_i$(*ds* f) $\equiv_i$ $\neg_i$(*di* f;*skip*) **by** *auto*
from *2 4* **show** *?thesis* **by** *auto*
qed

lemma *FstImpNotDiChopSkipB*:
$\vdash$ $\rhd f$ $\supset_i$ $\neg_i$(*di* (f;*skip*))
proof −
have *1*: $\vdash$ $\rhd f$ $\supset_i$ *bs* $\neg_i$ f
  by (*simp add*: *first-d-def*)
hence *2*: $\vdash$ $\rhd f$ $\supset_i$ $\neg_i$(*ds* f)
  using *NotDsEqvBsNot* **by** *auto*
have *3*: $\vdash$ *ds* f $\equiv_i$ *di* f ; *skip*
  using *DsDi* **by** *blast*
have *4*: $\vdash$ *di* f ; *skip* $\equiv_i$ (f;*true*$_i$);*skip*
  by (*simp add*: *di-d-def*)
have *5*: $\vdash$(f;*true*$_i$);*skip* $\equiv_i$ f;(*true*$_i$;*skip*)

using *ChopAssocB* **by** *blast*
have *6*: $\vdash f;(true_i;skip) \equiv_i f;(skip;true_i)$
using *SkipTrueEqvTrueSkip* **using** *TrueChopSkipEqvSkipChopTrue RightChopEqvChop* **by** *blast*
have *7*: $\vdash f;(skip;true_i) \equiv_i (f;skip);true_i$
using *ChopAssoc* **by** *blast*
have *8*: $\vdash (f;skip);true_i \equiv_i di(f;skip)$
by (*simp add*: *di-d-def*)
have *9*: $\vdash \neg_i(ds\ f) \equiv_i \neg_i(di(f;skip))$
using *3 4 5 6 7 8* **by** *auto*
from *2 9* **show** *?thesis* **by** *auto*
qed

lemma *FstImpDiEqv*:
$\vdash \triangleright f \supset_i (di\ f \equiv_i f)$
proof –
have *1*: $\vdash \triangleright f \supset_i \neg_i(di\ f;skip)$ **using** *FstImpNotDiChopSkip* **by** *blast*
have *2*: $\vdash di\ f \supset_i f \vee_i (di\ f;skip)$ **using** *DiEqvOrDiChopSkipB itl-prop(31)* **by** *blast*
have *3*: $\vdash \triangleright f \wedge_i di\ f \supset_i (f \vee_i (di\ f;skip)) \wedge_i \neg_i(di\ f;skip)$ **using** *1 2* **by** *auto*
have *4*: $\vdash (f \vee_i (di\ f;skip)) \wedge_i \neg_i(di\ f;skip) \equiv_i f \wedge_i \neg_i(di\ f;skip)$ **by** *auto*
have *5*: $\vdash \triangleright f \wedge_i di\ f \supset_i f \wedge_i \neg_i(di\ f;skip)$ **using** *3 4* **using** *itl-prop(31) prop02* **by** *blast*
hence *6*: $\vdash \triangleright f \wedge_i di\ f \supset_i f$ **using** *itl-prop(32)* **by** *blast*
hence *7*: $\vdash \triangleright f \supset_i (di\ f \supset_i f)$ **using** *FstAndElimL prop13 prop26 prop32* **by** *blast*
have *8*: $\vdash f \supset_i di\ f$ **using** *DiIntro* **by** *auto*
hence *9*: $\vdash \triangleright f \supset_i (f \supset_i (di\ f))$ **by** *auto*
from *7 9* **show** *?thesis* **by** *auto*
qed

lemma *FstAndDiFstAndEqvFstAnd*:
$\vdash \triangleright f \wedge_i di(\triangleright f \wedge_i g) \equiv_i \triangleright f \wedge_i g$
proof –
have *1*: $\vdash \triangleright f \wedge_i di(\triangleright f \wedge_i g) \supset_i \triangleright f$
by *auto*
have *2*: $\vdash \triangleright f \wedge_i di(\triangleright f \wedge_i g) \supset_i di(\triangleright f \wedge_i g)$
by *auto*
have *3*: $\vdash di(\triangleright f \wedge_i g) \equiv_i (\triangleright f \wedge_i g) \vee_i di((\triangleright f \wedge_i g);skip)$
using *DiEqvOrDiChopSkipA* **by** *blast*
have *4*: $\vdash di((\triangleright f \wedge_i g);skip) \equiv_i ((\triangleright f \wedge_i g);skip);true_i$
by (*simp add*: *di-d-def*)
have *5*: $\vdash \triangleright f \wedge_i di(\triangleright f \wedge_i g) \supset_i (\triangleright f \wedge_i g) \vee_i ((\triangleright f \wedge_i g);skip);true_i$
using *2 3 4* **by** *auto*
have *6*: $\vdash \triangleright f \wedge_i g \supset_i f$
using *FstAndElimL* **by** *auto*
hence *7*: $\vdash ((\triangleright f \wedge_i g);skip);true_i \supset_i (f;skip);true_i$
by *auto*
hence *8*: $\vdash ((\triangleright f \wedge_i g);skip);true_i \supset_i di(f;skip)$
by *auto*
have *9*: $\vdash \triangleright f \supset_i \neg_i(di(f;skip))$
using *FstImpNotDiChopSkipB* **by** *blast*
have *10*: $\vdash \triangleright f \wedge_i di(\triangleright f \wedge_i g) \supset_i ((\triangleright f \wedge_i g) \vee_i di(f;skip))$
using *5 8 prop35* **by** *blast*

have *11*: $\vdash \rhd f \wedge_i \; di(\rhd f \wedge_i g) \supset_i \neg_i(di(f;skip)) \wedge_i ((\rhd f \wedge_i g) \vee_i di(f;skip))$
using *9 10 1 itl-prop*(*32*) *prop02* **by** *blast*
have *12*: $\vdash \neg_i(di(f;skip)) \wedge_i ((\rhd f \wedge_i g) \vee_i di(f;skip)) \equiv_i \neg_i(di(f;skip)) \wedge_i ((\rhd f \wedge_i g))$
by *auto*
have *13*: $\vdash \rhd f \wedge_i \; di(\rhd f \wedge_i g) \supset_i (\rhd f \wedge_i g)$
using *11 12* **by** *auto*
have *14*: $\vdash (\rhd f \wedge_i g) \supset_i \rhd f$
by *auto*
hence *15*: $\vdash (\rhd f \wedge_i g) \supset_i di(\rhd f \wedge_i g \;)$
using *DiIntro* **by** *auto*
have *16*: $\vdash (\rhd f \wedge_i g) \supset_i \rhd f \wedge_i di(\rhd f \wedge_i g)$
using *14 15* **by** *auto*
from *13 16* **show** *?thesis* **using** *itl-prop*(*31*) **by** *blast*
qed

lemma *FstAndDiImpBsNotAndDi*:
$\vdash (\rhd f \wedge_i di\, g) \supset_i (bs\; \neg_i(di\, f \wedge_i g))$
proof −
have *1*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i ds(di\, f \wedge_i g)$
by (*simp add*: *ds-d-def*)
hence *2*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i ds(di\, f)$
using *DsAndImp* **by** *auto*
hence *3*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i di(di\, f);skip$
using *DsDi* **using** *itl-prop*(*31*) *prop02* **by** *blast*
hence *4*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i di\, f;skip$
using *DiEqvDiDi* **using** *LeftChopImpChop itl-prop*(*31*) *prop02* **by** *blast*
hence *5*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i ds\, f$
using *DsDi* **using** *itl-prop*(*31*) *prop02* **by** *blast*
hence *6*: $\vdash (\rhd f \wedge_i di\, g) \wedge_i \neg_i(bs\; \neg_i(di\, f \wedge_i g)) \supset_i \neg_i (\rhd f)$
using *DsImpNotFst* **using** *itl-prop*(*31*) *prop02* **by** *blast*
from *6* **show** *?thesis* **by** *auto*
qed

lemma *FstFstOrEqvFstOrL*:
$\vdash \rhd(\rhd f \vee_i g) \equiv_i \rhd(f \vee_i g)$
proof −
have *1*: $\vdash \rhd(f \vee_i g) \equiv_i (f \vee_i g) \wedge_i bs\neg_i(f \vee_i g)$
by (*simp add*: *first-d-def*)
have *2*: $\vdash \neg_i(f \vee_i g) \equiv_i (\neg_i f \wedge_i \neg_i g)$
by *auto*
hence *3*: $\vdash bs\neg_i(f \vee_i g) \equiv_i bs\, (\neg_i f \wedge_i \neg_i g)$
using *BsEqvRule* **by** *blast*
have *4*: $\vdash bs\, (\neg_i f \wedge_i \neg_i g) \equiv_i bs\, \neg_i f \wedge_i bs\, \neg_i g$
using *BsAndEqv itl-prop*(*30*) **by** *blast*
hence *5*: $\vdash (f \vee_i g) \wedge_i bs\neg_i(f \vee_i g) \equiv_i (f \vee_i g) \wedge_i bs\, \neg_i f \wedge_i bs\, \neg_i g$
using *4 3* **by** *simp*
have *6*: $\vdash (f \vee_i g) \wedge_i bs\, \neg_i f \wedge_i bs\, \neg_i g \equiv_i$
$((f \wedge_i bs\, \neg_i f) \vee_i (g \wedge_i bs\, \neg_i f)) \wedge_i bs\, \neg_i g$
by *auto*
have *7*: $\vdash ((f \wedge_i bs\, \neg_i f) \vee_i (g \wedge_i bs\, \neg_i f)) \wedge_i bs\, \neg_i g \equiv_i$

$(\triangleright f \vee_i (g \wedge_i bs \neg_i f)) \wedge_i bs \neg_i g$
by (*simp add*: *first-d-def*)
have *8*: $\vdash (\triangleright f \vee_i (g \wedge_i bs \neg_i f)) \wedge_i bs \neg_i g \equiv_i$
$((\triangleright f \vee_i g) \wedge_i (\triangleright f \vee_i bs\neg_i f)) \wedge_i bs \neg_i g$
by *auto*
have *9*: $\vdash ((\triangleright f \vee_i g) \wedge_i (\triangleright f \vee_i bs\neg_i f)) \wedge_i bs \neg_i g \equiv_i$
$((\triangleright f \vee_i g) \wedge_i ((f \wedge_i bs \neg_i f) \vee_i bs\neg_i f)) \wedge_i bs \neg_i g$
by (*simp add*: *first-d-def*)
have *10*: $\vdash ((\triangleright f \vee_i g) \wedge_i ((f \wedge_i bs \neg_i f) \vee_i bs\neg_i f)) \wedge_i bs \neg_i g \equiv_i$
$(\triangleright f \vee_i g) \wedge_i bs \neg_i f \wedge_i bs \neg_i g$
by *auto*
have *11*: $\vdash (\triangleright f \vee_i g) \wedge_i bs \neg_i f \wedge_i bs \neg_i g \equiv_i$
$(\triangleright f \vee_i g) \wedge_i bs(\neg_i(\triangleright f))\wedge_i bs \neg_i g$
using *BsNotFstEqvBsNot* **using** *itl-prop*(*30*) *prop05 prop06* **by** *blast*
have *12*: $\vdash (\triangleright f \vee_i g) \wedge_i bs(\neg_i(\triangleright f))\wedge_i bs \neg_i g \equiv_i$
$(\triangleright f \vee_i g) \wedge_i bs (\neg_i(\triangleright f) \wedge_i \neg_i g)$
using *BsAndEqv* **using** *prop05* **by** *blast*
have *13*: $\vdash (\neg_i(\triangleright f) \wedge_i \neg_i g) \equiv_i \neg_i(\triangleright f \vee_i g)$
by *auto*
hence *14*: $\vdash bs (\neg_i(\triangleright f) \wedge_i \neg_i g) \equiv_i bs \neg_i(\triangleright f \vee_i g)$
using *BsEqvRule* **by** *blast*
hence *15*: $\vdash (\triangleright f \vee_i g) \wedge_i bs (\neg_i(\triangleright f) \wedge_i \neg_i g) \equiv_i (\triangleright f \vee_i g) \wedge_i bs \neg_i(\triangleright f \vee_i g)$
by *auto*
have *16*: $\vdash (\triangleright f \vee_i g) \wedge_i bs \neg_i(\triangleright f \vee_i g) \equiv_i \triangleright(\triangleright f \vee_i g)$
by (*simp add*: *first-d-def*)
from *16 15 12 11 10 9 8 7 6 5 1* **show** *?thesis* **by** *auto*
qed

lemma *FstFstOrEqvFstOrR*:
$\vdash \triangleright(f \vee_i \triangleright g)\equiv_i \triangleright(f \vee_i g)$
proof −
have *1*: $\vdash (f \vee_i \triangleright g)\equiv_i (\triangleright g \vee_i f)$ **by** *auto*
hence *2*: $\vdash \triangleright (f \vee_i \triangleright g)\equiv_i \triangleright(\triangleright g \vee_i f)$ **using** *FstEqvRule* **by** *blast*
have *3*: $\vdash \triangleright(\triangleright g \vee_i f) \equiv_i \triangleright(g \vee_i f)$ **using** *FstFstOrEqvFstOrL* **by** *blast*
have *4*: $\vdash (g \vee_i f) \equiv_i (f \vee_i g)$ **by** *auto*
hence *5*: $\vdash \triangleright(g \vee_i f) \equiv_i \triangleright(f \vee_i g)$ **using** *FstEqvRule* **by** *blast*
from *2 3 5* **show** *?thesis* **by** *auto*
qed

lemma *FstFstOrEqvFstOr*:
$\vdash \triangleright(\triangleright f \vee_i \triangleright g) \equiv_i \triangleright(f \vee_i g)$
proof −
have *1*: $\vdash \triangleright(\triangleright f \vee_i \triangleright g) \equiv_i \triangleright(f \vee_i \triangleright g)$ **using** *FstFstOrEqvFstOrL* **by** *blast*
have *2*: $\vdash \triangleright(f \vee_i \triangleright g) \equiv_i \triangleright(f \vee_i g)$ **using** *FstFstOrEqvFstOrR* **by** *blast*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *FstLenEqvLen*:
$\vdash \triangleright(len(k)) \equiv_i len(k)$
proof −

have *1*: $\vdash \rhd($ *len*$(k) \wedge_i$ *true*$_i) \equiv_i$ *len*$(k) \wedge_i$ *true*$_i$ **using** *FstLenAndEqvLenAnd* **by** *blast*
have *2*: $\vdash$ *len*$(k) \wedge_i$ *true*$_i \equiv_i$ *len*(k) **by** *auto*
hence *3*: $\vdash \rhd($ *len*$(k) \wedge_i$ *true*$_i) \equiv_i \rhd($*len*$(k))$ **using** *FstEqvRule* **by** *blast*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *FstSkip*:
$\vdash \rhd$ *skip* $\equiv_i$ *skip*
proof –
have *1*: $\vdash$ *skip* $\equiv_i$ *len*(1) **using** *LenOneEqvSkip* **using** *itl-prop*(30) **by** *blast*
hence *2*: $\vdash \rhd$*skip* $\equiv_i \rhd$ $($*len*$(1))$ **using** *FstEqvRule* **by** *blast*
have *3*: $\vdash \rhd($ *len*$(1)) \equiv_i$ *len*(1) **using** *FstLenEqvLen* **by** *blast*
from *1 2 3* **show** *?thesis* **using** *LenOneEqvSkip prop03* **by** *blast*
qed

lemma *NotChopNotSkip*:
$\vdash \neg_i(f;$*skip*$) \equiv_i$ *empty* $\vee_i$ $((\neg_i f);$*skip*$)$
proof –
have *1*: $\vdash \neg_i(\neg_i(\neg_i f);$*skip*$) \equiv_i$ *empty* $\vee_i$ $((\neg_i f);$*skip*$)$ **using** *NotNotChopSkip* **by** *blast*
have *2*: $\vdash \neg_i(\neg_i(\neg_i f);$*skip*$) \equiv_i \neg_i(f;$*skip*$)$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *BiBoxNotEqvNotTrueChopChopTrue*:
$\vdash$ *bi*$(\Box \neg_i f) \equiv_i \neg_i(($*true*$_i;f);$*true*$_i$ $)$
by *auto*

lemma *BoxMoreStateEqvBsFinState*:
$\vdash \Box($*more* $\supset_i \neg_i$ $($*init w*$)) \equiv_i$ *bs*$(\neg_i($*fin*$($*init w*$)))$
proof –
have *1*: $\vdash \Box($*more* $\supset_i \neg_i$ $($*init w*$)) \equiv_i \neg_i(\Diamond(\neg_i($*more* $\supset_i \neg_i$ $($*init w*$))))$
  by *auto*
have *2*: $\vdash \neg_i(\Diamond(\neg_i($*more* $\supset_i \neg_i$ $($*init w*$)))) \equiv_i \neg_i($*true*$_i;($*init w* $\wedge_i$ *more*$))$
  by *auto*
have *3*: $\vdash$ *more* $\equiv_i$ *true*$_i$; *skip*
  using *MoreEqvSkipChopTrue SkipTrueEqvTrueSkip prop03* **by** *blast*
have *4*: $\vdash$ *init w* $\wedge_i$ *more* $\equiv_i$ *init w* $\wedge_i$ $($*true*$_i$; *skip*$)$
  using *3* **by** *auto*
have *5*: $\vdash$ *init w* $\wedge_i$ $($*true*$_i$; *skip*$)$ $\equiv_i$ $(($*init w* $\wedge_i$ *empty*$);($*true*$_i;$*skip*$))$
  using *StateAndEmptyChop itl-prop*(30) **by** *blast*
have *6*: $\vdash$ *init w* $\wedge_i$ *more* $\equiv_i$ $(($*init w* $\wedge_i$ *empty*$);($*true*$_i;$*skip*$))$
  using *4 5* **by** *auto*
have *7*: $\vdash$ $($*true*$_i;($*init w* $\wedge_i$ *more*$)) \equiv_i ($*true*$_i;(($*init w* $\wedge_i$ *empty*$);($*true*$_i;$*skip*$))$ $)$
  using *6 RightChopEqvChop* **by** *blast*
have *8*: $\vdash$ $($*true*$_i;(($*init w* $\wedge_i$ *empty*$);($*true*$_i;$*skip*$))$ $) \equiv_i$
    $((($*true*$_i;($*init w* $\wedge_i$ *empty*$));($*true*$_i;$*skip*$))$ $)$
  using *ChopAssoc* **by** *blast*
have *9*: $\vdash$ $((($*true*$_i;($*init w* $\wedge_i$ *empty*$));($*true*$_i;$*skip*$))$ $) \equiv_i$
    $(((($*true*$_i;($*init w* $\wedge_i$ *empty*$));$*true*$_i);$*skip*$)$ $)$
  using *ChopAssoc* **by** *blast*

have *10*: $\vdash (true_i;(init\ w \wedge_i\ more)) \equiv_i$
$((((true_i;(init\ w \wedge_i\ empty));true_i);skip)\ )$
using *7 8 9* **by** *auto*
hence *11*: $\vdash \neg_i(true_i;(init\ w \wedge_i\ more)) \equiv_i$
$\neg_i((((true_i;(init\ w \wedge_i\ empty));true_i);skip)\ )$
by *auto*
have *12*: $\vdash \neg_i((((true_i;(init\ w \wedge_i\ empty));true_i);skip)\ ) \equiv_i$
$empty \vee_i (\neg_i((true_i;(init\ w \wedge_i\ empty));true_i);skip)$
using *NotChopNotSkip* **by** *blast*
have *13*: $\vdash (\neg_i((true_i;(init\ w \wedge_i\ empty));true_i)) \equiv_i bi(\Box\ \neg_i(init\ w \wedge_i\ empty))$
using *BiBoxNotEqvNotTrueChopChopTrue itl-prop*(30) **by** *blast*
hence *14*: $\vdash (\neg_i((true_i;(init\ w \wedge_i\ empty));true_i));skip \equiv_i$
$(bi(\Box\ \neg_i(init\ w \wedge_i\ empty)));skip$
using *RightChopEqvChop* **by** *auto*
hence *15*: $\vdash empty \vee_i (\neg_i((true_i;(init\ w \wedge_i\ empty));true_i));skip \equiv_i$
$empty \vee_i(bi(\Box\ \neg_i(init\ w \wedge_i\ empty)));skip$
by *auto*
have *16*: $\vdash \neg_i((((true_i;(init\ w \wedge_i\ empty));true_i);skip)\ ) \equiv_i$
$empty \vee_i (bi(\Box\ \neg_i(init\ w \wedge_i\ empty));skip)$
using *12 15* **by** *auto*
have *17*: $\vdash empty \vee_i (bi(\Box\ \neg_i(init\ w \wedge_i\ empty));skip) \equiv_i$
$empty \vee_i (bi(\Box\ (\neg_i(init\ w) \vee_i \neg_i empty));skip)$
by *auto*
have *18*: $\vdash \Box\ (\neg_i(init\ w) \vee_i \neg_i empty) \equiv_i \Box\ (\neg_i empty \vee_i \neg_i(init\ w)\ )$
by *auto*
have *19*: $\vdash \Box\ (\neg_i empty \vee_i \neg_i(init\ w)\ ) \equiv_i \Box(empty \supset_i \neg_i(init\ w)\ )$
by *auto*
have *20*: $\vdash \Box(empty \supset_i \neg_i(init\ w)\ ) \equiv_i fin\ (\neg_i(init\ w)\ )$
by (*simp add*: *fin-d-def*)
have *21*: $\vdash fin\ (\neg_i(init\ w)\ ) \equiv_i \neg_i(fin\ (init\ w))$
using *FinEqvFin FinNotStateEqvNotFinState Initprop*(2) *prop03* **by** *blast*
have *22*: $\vdash bi(\Box\ (\neg_i(init\ w) \vee_i \neg_i empty)) \equiv_i bi\ (\neg_i(fin\ (init\ w)))$
using *18 19 20 21 BiEqvBi* **using** *prop03* **by** *blast*
hence *23*: $\vdash (bi(\Box\ (\neg_i(init\ w) \vee_i \neg_i empty)));skip \equiv_i (bi\ (\neg_i(fin\ (init\ w))));skip$
using *RightChopEqvChop* **by** *auto*
hence *24*: $\vdash empty \vee_i (bi(\Box\ (\neg_i(init\ w) \vee_i \neg_i empty)));skip \equiv_i$
$empty \vee_i (bi\ (\neg_i(fin\ (init\ w))));skip$
by *auto*
hence *25*: $\vdash empty \vee_i (bi\ (\neg_i(fin\ (init\ w))));skip \equiv_i bs(\neg_i(fin\ (init\ w)))$
by (*simp add*:*bs-d-def*)
from *1 2 11 16 17 24 25* **show** *?thesis* **by** *auto*
qed

lemma *HaltStateEqvFstFinState*:
$\vdash halt\ (init\ w) \equiv_i \rhd (fin\ (init\ w))$
proof –
have *1*: $\vdash halt\ (init\ w) \equiv_i \Box(empty \equiv_i (init\ w))$ **by** (*simp add*: *halt-d-def*)
have *2*: $\vdash \Box(empty \equiv_i (init\ w)) \equiv_i \Box((empty \supset_i (init\ w)) \wedge_i ((init\ w) \supset_i empty))$ **by** *auto*
have *3*: $\vdash \Box((empty \supset_i (init\ w)) \wedge_i ((init\ w) \supset_i empty)) \equiv_i$
$\Box((empty \supset_i (init\ w))) \wedge_i \Box((init\ w) \supset_i empty)$ **by** *auto*

have *4*: $\vdash ((init\ w) \supset_i empty) \equiv_i (more \supset_i \neg_i(init\ w))$ **by** *auto*
hence *5*: $\vdash \Box\ ((init\ w) \supset_i empty) \equiv_i \Box\ (more \supset_i \neg_i(init\ w))$ **using** *BoxEqvBox* **by** *blast*
have *6*: $\vdash \Box\ (more \supset_i \neg_i(init\ w)) \equiv_i bs(\neg_i(fin(init\ w)))$ **using** *BoxMoreStateEqvBsFinState* **by** *blast*
have *7*: $\vdash \Box((empty \supset_i (init\ w))) \equiv_i fin(init\ w)$ **by** (*simp add*: *fin-d-def*)
have *8*: $\vdash\ \Box((empty \supset_i (init\ w))) \wedge_i \Box((init\ w) \supset_i empty) \equiv_i$
$fin(init\ w) \wedge_i bs(\neg_i(fin(init\ w)))$ **using** *5 6 7* **by** *auto*
from *1 2 3 8* **show** *?thesis* **by** (*simp add*:*first-d-def*)
qed

lemma *FstLenEqvLenFst*:
$\vdash \rhd(len\ k\ ;\ f) \equiv_i len\ k\ ;\ \rhd f$
proof –
have *1*: $\vdash len\ k\ ;\ f \equiv_i \rhd(len\ k)\ ;\ f$ **using** *FstLenEqvLen LeftChopEqvChop* **by** *auto*
have *2*: $\vdash \rhd(len\ k\ ;\ f) \equiv_i \rhd\ (\rhd(len\ k)\ ;\ f)$ **using** *1 FstEqvRule* **by** *blast*
have *3*: $\vdash \rhd\ (\rhd(len\ k)\ ;\ f) \equiv_i \rhd(len\ k)\ ;\ \rhd f$ **using** *FstFstChopEqvFstChopFst* **by** *blast*
have *4*: $\vdash \rhd(len\ k)\ ;\ \rhd f \equiv_i len\ k\ ;\ \rhd f$ **using** *FstLenEqvLen LeftChopEqvChop* **by** *auto*
from *2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *FstNextEqvNextFst*:
$\vdash \rhd(\bigcirc\ f) \equiv_i \bigcirc(\ \rhd f)$
proof –
have *1*: $\vdash \rhd(\bigcirc\ f) \equiv_i \rhd(skip\ ;\ f)$ **using** *FstEqvRule* **by** (*simp add*: *next-d-def*)
have *2*: $\vdash skip\ ;\ f \equiv_i \rhd skip\ ;\ f$ **using** *FstSkip* **by** *auto*
have *3*: $\vdash \rhd(skip\ ;\ f) \equiv_i \rhd\ (\rhd skip\ ;\ f)$ **using** *2 FstEqvRule LeftChopEqvChop* **by** *blast*
have *4*: $\vdash \rhd\ (\rhd skip\ ;\ f) \equiv_i \rhd skip\ ;\ \rhd f$ **using** *3 FstFstChopEqvFstChopFst* **by** *blast*
have *5*: $\vdash \rhd skip\ ;\ \rhd f \equiv_i skip\ ;\ \rhd f$ **using** *4 FstSkip LeftChopEqvChop* **by** *blast*
have *6*: $\vdash skip\ ;\ \rhd f \equiv_i \bigcirc(\ \rhd f)$ **by** (*simp add*: *next-d-def*)
from *1 2 3 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *FstDiamondStateEqvHalt*:
$\vdash \rhd\ (\Diamond\ (init\ w)) \equiv_i halt\ (init\ w)$
proof –
have *1*: $\vdash \Diamond\ (init\ w) \equiv_i \Diamond\ ((init\ w) \wedge_i true_i)$ **by** *simp*
have *2*: $\vdash fin\ (init\ w)\ ;\ true_i \equiv_i \Diamond\ ((init\ w) \wedge_i true_i)$ **using** *1 FinChopEqvDiamond* **by** *blast*
have *3*: $\vdash fin\ (init\ w)\ ;\ true_i \equiv_i di\ (fin\ (init\ w))$ **using** *di-d-def* **by** *simp*
have *4*: $\vdash (\Diamond\ (init\ w)) \equiv_i (di\ (fin\ (init\ w)))$ **using** *1 2 3* **by** *auto*
have *5*: $\vdash \rhd\ (\Diamond\ (init\ w)) \equiv_i \rhd\ (di\ (fin\ (init\ w)))$ **using** *4 FstEqvRule* **by** *blast*
hence *6*: $\vdash \rhd\ (\Diamond\ (init\ w)) \equiv_i \rhd\ (fin\ (init\ w))$ **using** *FstDiEqvFst* **by** *auto*
hence *7*: $\vdash \rhd\ (\Diamond\ (init\ w)) \equiv_i halt\ (init\ w)$ **using** *HaltStateEqvFstFinState* **by** *auto*
from *7* **show** *?thesis* **by** *simp*
qed

lemma *FstBoxStateEqvStateAndEmpty*:
$\vdash \rhd\ (\Box\ (init\ w)) \equiv_i (init\ w) \wedge_i empty$
proof –
have *1*: $\vdash (init\ w) \wedge_i (\Box\ (init\ w))^\star \equiv_i \Box\ (init\ w)$
using *BoxCSEqvBox* **by** *blast*
have *2*: $\vdash \Box\ (init\ w) \equiv_i (init\ w) \wedge_i (\Box\ (init\ w))^\star$

using *1* **by** *simp*
hence *3*: $\vdash \Box\ (init\ w) \equiv_i (init\ w) \wedge_i (\Box\ (init\ w))^\star$
by *blast*
have *4*: $\vdash ((init\ w) \wedge_i empty)\ ;\ (\Box\ (init\ w))^\star \equiv_i (init\ w) \wedge_i (\Box\ (init\ w))^\star$
using *StateAndEmptyChop* **by** *blast*
have *5*: $\vdash (init\ w) \wedge_i (\Box\ (init\ w))^\star\ \equiv_i\ ((init\ w) \wedge_i empty)\ ;\ (\Box\ (init\ w))^\star$
using *4* **by** *simp*
have *6*: $\vdash \Box\ (init\ w) \equiv_i ((init\ w) \wedge_i empty)\ ;\ (\Box\ (init\ w))^\star$
using *3 5 prop03* **by** *blast*
have *7*: $\vdash ((init\ w) \wedge_i empty)\ ;\ (\Box\ (init\ w))^\star \equiv_i \rhd\ (init\ w)\ ;\ (\Box\ (init\ w))^\star$
using *FstState* **by** *auto*
have *8*: $\vdash \Box\ (init\ w) \equiv_i \rhd\ (init\ w)\ ;\ (\Box\ (init\ w))^\star$
using *6 7 prop03* **by** *blast*
have *9*: $\vdash \rhd\ (\Box\ (init\ w)) \equiv_i \rhd\ (\rhd\ (init\ w)\ ;\ (\Box\ (init\ w))^\star)$
using *8 FstEqvRule* **by** *blast*
have *10*: $\vdash \rhd\ (\rhd\ (init\ w)\ ;\ (\Box\ (init\ w))^\star) \equiv_i \rhd\ (init\ w)\ ;\ \ \rhd\ ((\Box\ (init\ w))^\star)$
using *FstFstChopEqvFstChopFst* **by** *blast*
have *11*: $\vdash \rhd\ (init\ w)\ ;\ \ \rhd\ ((\Box\ (init\ w))^\star) \equiv_i \rhd\ (init\ w)\ ;\ empty$
using *RightChopEqvChop FstCSEqvEmpty* **by** *blast*
have *12*: $\vdash \rhd\ (init\ w)\ ;\ empty \equiv_i \rhd\ (init\ w)$
using *RightChopEqvChop ChopEmpty* **by** *blast*
have *13*: $\vdash \rhd\ (init\ w) \equiv_i (init\ w) \wedge_i empty$
using *FstState* **by** *auto*
from *9 10 11 12 13* **show** *?thesis* **by** *auto*
qed

end

theory *Monitor*
imports *First*

begin

7 Monitors

The RV monitors language is introduced plus the algebraic properties of the monitor operators.

7.1 Syntax

datatype *'a monitor* =
mFIRST-d 'a pitl ((*FIRST* -))
| *mUPTO-d 'a monitor 'a monitor* ((- *UPTO* -) [*84*,*84*] *83*)
| *mTHRU-d 'a monitor 'a monitor* ((- *THRU* -) [*84*,*84*] *83*)
| *mTHEN-d 'a monitor 'a monitor* ((- *THEN* -) [*84*,*84*] *83*)
| *mWITH-d 'a monitor 'a pitl* ((- *WITH* -) [*84*,*84*] *83*)

7.2 Derived Monitors

definition *mHALT-d* ((*HALT* -) [*84*] *83*)
where
HALT w ≡ *FIRST* (*fin* (*init w*))

definition *mLEN-d* ((*LEN* -) [*84*] *83*)
where
LEN k ≡ *FIRST* (*len k*)

definition *mEMPTY-d* (*EMPTY*)
where
EMPTY ≡ *FIRST empty*

definition *mSKIP-d* (*SKIP*)
where
SKIP ≡ *FIRST skip*

definition *mGUARD-d* ((*GUARD* -) [*84*] *83*)
where
GUARD w ≡ *EMPTY WITH* ((*init w*))

definition *mFAIL-d* (*FAIL*)
where
FAIL ≡ *GUARD* $false_i$

primrec *mTIMES-d* :: *'a monitor* ⇒ *nat* ⇒ *'a monitor* ((- *TIMES* -) [*84*,*84*] *83*)
where
mTIMES-0 : *a TIMES 0* = *EMPTY*
| *mTIMES-Suc*: *a TIMES* (*Suc k*) = *a THEN* (*a TIMES k*)

definition *mALWAYS-d* ((- *ALWAYS* -) [*84*,*84*] *83*)
where
a ALWAYS (*w*) ≡ *a WITH* (*bi* (*fin* (*init w*)))

definition *mSOMETIME-d* ((- *SOMETIME* -) [*84*,*84*] *83*)
where
a SOMETIME (*w*) ≡ *a WITH* (*di* (*fin* (*init w*)))

definition *mlimit-d* ((*Limit* -) [*84*] *83*)
where
Limit f ≡ (*bs* ($\neg_i$ *f*))

definition *mWITHIN-d* ((- *WITHIN* -) [*84*,*84*] *83*)
where
a WITHIN (*f*) ≡ *a WITH* (*Limit f*)

definition *mUNTIL-d* ((- *UNTIL* -) [*84*,*84*] *83*)
where
w1 UNTIL w2 ≡ (*HALT w2*) *WITH* (*bm w1*)

7.3 Semantics

fun *semantics-monitor* :: $'a$ *monitor* $\Rightarrow$ $'a$ *pitl* (($\mathcal{M}$ -) [*80*] *80*)
where
($\mathcal{M}$ (*FIRST a*)) = $\triangleright$ *a*
| ($\mathcal{M}$ (*a UPTO b*)) = $\triangleright$(($\mathcal{M}$ *a*) $\vee_i$ ($\mathcal{M}$ *b*))
| ($\mathcal{M}$ (*a THRU b*)) = $\triangleright$(*di*($\mathcal{M}$ *a*) $\wedge_i$ *di*($\mathcal{M}$ *b*))
| ($\mathcal{M}$ (*a THEN b*)) = (($\mathcal{M}$ *a*);($\mathcal{M}$ *b*))
| ($\mathcal{M}$ (*a WITH f*)) = (($\mathcal{M}$ *a*) $\wedge_i$ *f*)

definition *mAND-d* ((- *AND* -) [*84*,*84*] *83*)
where
a AND b $\equiv$ *a WITH* ($\mathcal{M}$ *b*)

definition *mITERATE-d* ((- *ITERATE* -) [*84*,*84*] *83*)
where
a ITERATE b $\equiv$ *a WITH* ($\mathcal{M}$ *b*)*

definition *mSTAR-d* ((- *STAR* -) [*84*,*84*] *83*)
where
a STAR f $\equiv$ (*FIRST*($\diamond$ *f*)) *ITERATE* (*a*)

definition *mREPEAT-d* ((- *REPEATUNTIL* -) [*84*,*84*] *83*)
where
a REPEATUNTIL w $\equiv$ ((*HALT w*) *ITERATE* (*a WITH* (*keep*($\neg_i$ (*init w*))))) *THEN a*

7.4 Monitor Laws

lemma *MFixFst*:
$\vdash$ ($\mathcal{M}$ *a*) $\equiv_i$ $\triangleright$ ($\mathcal{M}$ *a*)
proof
(*induct a*)
case (*mFIRST-d x*)
then show *?case*
proof −
have *1*: $\vdash$ ($\mathcal{M}$ *FIRST x*) $\equiv_i$ $\triangleright$ *x* **by** *simp*
have *2*: $\vdash$ $\triangleright$ *x* $\equiv_i$ $\triangleright$ ($\triangleright$ *x*) **using** *FstFixFst* **by** *auto*
have *3*: $\vdash$ $\triangleright$ ($\triangleright$ *x*) $\equiv_i$ $\triangleright$($\mathcal{M}$ *FIRST x*) **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed
next
case (*mUPTO-d a1 a2*)
then show *?case*
proof −
have *1*: $\vdash$ ($\mathcal{M}$ (*a1 UPTO a2*)) $\equiv_i$ $\triangleright$(($\mathcal{M}$ *a1*) $\vee_i$ ($\mathcal{M}$ *a2*)) **by** *simp*
have *2*: $\vdash$ $\triangleright$(($\mathcal{M}$ *a1*) $\vee_i$ ($\mathcal{M}$ *a2*)) $\equiv_i$ $\triangleright$($\triangleright$(($\mathcal{M}$ *a1*) $\vee_i$ ($\mathcal{M}$ *a2*))) **using** *FstFixFst* **by** *auto*
have *3*: $\vdash$ $\triangleright$($\triangleright$(($\mathcal{M}$ *a1*) $\vee_i$ ($\mathcal{M}$ *a2*))) $\equiv_i$ $\triangleright$($\mathcal{M}$ (*a1 UPTO a2*)) **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed
next
case (*mTHRU-d a1 a2*)

then show *?case*
proof −
have *1*: $\vdash (\mathcal{M}\ (a1\ THRU\ a2)) \equiv_i \triangleright(\ di\ (\mathcal{M}\ a1) \wedge_i di(\mathcal{M}\ a2))$ **by** *simp*
have *2*: $\vdash \triangleright(\ di\ (\mathcal{M}\ a1) \wedge_i di(\mathcal{M}\ a2)) \equiv_i \triangleright(\triangleright(di(\mathcal{M}\ a1) \wedge_i di(\mathcal{M}\ a2)))$ **using** *FstFixFst* **by** *auto*
have *3*: $\vdash \triangleright(\triangleright(\ di\ (\mathcal{M}\ a1) \wedge_i di(\mathcal{M}\ a2))) \equiv_i \triangleright(\mathcal{M}\ (a1\ THRU\ a2))$ **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed
next
case (*mTHEN-d a1 a2*)
then show *?case*
proof −
have *1*: $\vdash (\mathcal{M}\ (a1\ THEN\ a2)) \equiv_i (\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2)$
by *simp*
have *2*: $\vdash (\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2) \equiv_i \triangleright(\mathcal{M}\ a1)\ ;\ \triangleright(\mathcal{M}\ a2)$
using *ChopEqvChop mTHEN-d.hyps(1) mTHEN-d.hyps(2)* **by** *blast*
have *3*: $\vdash \triangleright(\mathcal{M}\ a1)\ ;\ \triangleright(\mathcal{M}\ a2) \equiv_i \triangleright(\triangleright(\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2))$
using *FstFstChopEqvFstChopFst itl-prop(30)* **by** *blast*
have *4*: $\vdash \triangleright(\triangleright(\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2)) \equiv_i \triangleright((\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2))$
using *FstEqvRule LeftChopEqvChop itl-prop(30) mTHEN-d.hyps(1)* **by** *blast*
have *5*: $\vdash \triangleright((\mathcal{M}\ a1)\ ;\ (\mathcal{M}\ a2)) \equiv_i \triangleright(\mathcal{M}\ (a1\ THEN\ a2))$
by *simp*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed
next
case (*mWITH-d a x2*)
then show *?case*
proof −
have *1*: $\vdash (\mathcal{M}\ (a\ WITH\ x2)) \equiv_i (\mathcal{M}\ a) \wedge_i (\ x2)$
by *simp*
have *2*: $\vdash (\mathcal{M}\ a) \wedge_i (\ x2) \equiv_i \triangleright(\mathcal{M}\ a) \wedge_i (\ x2)$
using *mWITH-d.hyps* **by** *auto*
have *3*: $\vdash \triangleright(\mathcal{M}\ a) \wedge_i (\ x2) \equiv_i \triangleright(\triangleright(\mathcal{M}\ a) \wedge_i (\ x2))$
using *FstFstAndEqvFstAnd itl-prop(30)* **by** *blast*
have *4*: $\vdash \triangleright(\triangleright(\mathcal{M}\ a) \wedge_i (\ x2)) \equiv_i \triangleright((\mathcal{M}\ a) \wedge_i (\ x2))$
using *2 FstEqvRule itl-prop(30)* **by** *blast*
have *5*: $\vdash \triangleright((\mathcal{M}\ a) \wedge_i (\ x2)) \equiv_i \triangleright(\mathcal{M}\ (a\ WITH\ x2))$
by *simp*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed
qed

lemma *MGuardFalseEqvFalse*:
$\vdash \mathcal{M}\ (GUARD\ false_i) \equiv_i false_i$
proof −
have *1*: $\vdash \mathcal{M}(GUARD\ false_i) \equiv_i \mathcal{M}(EMPTY\ WITH\ (init\ false_i))$ **by** (*simp add*: *mGUARD-d-def*)
have *2*: $\vdash \mathcal{M}(EMPTY\ WITH\ (init\ false_i)) \equiv_i \mathcal{M}(EMPTY) \wedge_i (init\ false_i)$ **by** *simp*
have *3*: $\vdash false_i \equiv_i (init\ false_i)$ **by** *simp*
have *4*: $\vdash \mathcal{M}(EMPTY) \wedge_i (init\ false_i) \equiv_i \mathcal{M}(EMPTY) \wedge_i false_i$ **using** *3* **by** *simp*
have *5*: $\vdash \mathcal{M}(EMPTY) \wedge_i false_i \equiv_i false_i$ **by** *simp*
have *6*: $\vdash \mathcal{M}(EMPTY) \wedge_i (init\ false_i) \equiv_i false_i$ **using** *4 5* **by** *simp*

have *7*: $\vdash\mathcal{M}(EMPTY\ WITH\ (init\ false_i)) \equiv_i false_i$ **using** *2 6* **by** *simp*
have *8*: $\vdash\mathcal{M}(GUARD\ false_i) \equiv_i false_i$ **using** *1 7* **by** *simp*
from *8* **show** *?thesis* **by** *auto*
qed

lemma *MFirstFalseEqvFalse*:
$\vdash \mathcal{M}\ (FIRST\ false_i) \equiv_i false_i$
proof –
have *1*: $\vdash \mathcal{M}(FIRST\ false_i) \equiv_i \triangleright\ false_i$ **by** *simp*
have *2*: $\vdash \mathcal{M}(FIRST\ false_i) \equiv_i false_i$ **using** *FstFalse* **by** *simp*
from *2* **show** *?thesis* **by** *auto*
qed

lemma *MFailAlt*:
$\vdash \mathcal{M}\ FAIL \equiv_i false_i$
proof –
have *1*: $\vdash \mathcal{M}\ FAIL \equiv_i \mathcal{M}\ (GUARD\ (false_i))$ **by** (*simp add*: *mFAIL-d-def*)
have *2*: $\vdash \mathcal{M}\ (GUARD\ (false_i)) \equiv_i false_i$ **using** *MGuardFalseEqvFalse* **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MFailEqvFirstFalseWithinEmpty*:
$\vdash (\mathcal{M}\ FAIL) \equiv_i \mathcal{M}((FIRST\ false_i)\ WITHIN\ (\ empty\ ))$
proof –
have *1*: $\vdash \mathcal{M}(\ (FIRST\ false_i)\ WITHIN\ (\ empty\ )) \equiv_i \mathcal{M}\ ((FIRST\ false_i)\ WITH\ (Limit\ empty)\ )$
by (*simp add*: *mWITHIN-d-def*)
have *2*: $\vdash \mathcal{M}\ ((FIRST\ false_i)\ WITH\ (Limit\ empty)\ ) \equiv_i \mathcal{M}\ (FIRST\ false_i) \wedge_i (Limit\ empty\ )$
by *simp*
have *3*: $\vdash \mathcal{M}\ ((FIRST\ false_i)\ WITH\ (Limit\ empty)\ ) \equiv_i false_i$
using *MFirstFalseEqvFalse* **by** *auto*
have *4*: $\vdash \mathcal{M}(\ (FIRST\ false_i)\ WITHIN\ (\ empty\ )) \equiv_i false_i$
using *1 3* **by** *auto*
have *5*: $\vdash (\mathcal{M}\ FAIL) \equiv_i false_i$
using *MFailAlt* **by** *simp*
from *4 5* **show** *?thesis* **by** *auto*
qed

lemma *MEmptyAlt*:
$\vdash (\mathcal{M}\ EMPTY) \equiv_i empty$
proof –
have *1*: $\vdash (\mathcal{M}\ EMPTY) \equiv_i (\mathcal{M}\ (FIRST\ empty))$ **by** (*simp add*: *mEMPTY-d-def*)
have *2*: $\vdash (\mathcal{M}\ (FIRST\ empty)) \equiv_i \triangleright\ empty$ **by** *simp*
have *3*: $\vdash \triangleright\ empty \equiv_i empty$ **using** *FstEmpty* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MSkipAlt*:
$\vdash \mathcal{M}\ SKIP \equiv_i skip$
proof –
have *1*: $\vdash \mathcal{M}\ SKIP \equiv_i \mathcal{M}(FIRST\ skip)$ **by** (*simp add*: *mSKIP-d-def*)

have *2*: $\vdash \mathcal{M}(\mathit{FIRST\ skip}) \equiv_i \rhd\ \mathit{skip}$ **by** *simp*
have *3*: $\vdash \rhd\ \mathit{skip} \equiv_i \mathit{skip}$ **using** *FstSkip* **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MGuardAlt*:
$\vdash \mathcal{M}\ (\mathit{GUARD}(w)) \equiv_i \mathit{empty} \wedge_i \mathit{init}\ w$
proof −
have *1*: $\vdash \mathcal{M}\ (\mathit{GUARD}(w)) \equiv_i \mathcal{M}(\mathit{EMPTY\ WITH}\ (\ (\mathit{init}\ w)))$ **by** (*simp add*:*mGUARD-d-def*)
have *2*: $\vdash \mathcal{M}(\mathit{EMPTY\ WITH}\ (\ (\mathit{init}\ w))) \equiv_i (\mathcal{M}\ \mathit{EMPTY}) \wedge_i (\ \mathit{init}\ w)$ **by** *simp*
have *3*: $\vdash (\mathcal{M}\ \mathit{EMPTY}) \wedge_i (\ \mathit{init}\ w) \equiv_i \mathit{empty} \wedge_i (\ \mathit{init}\ w)$ **using** *MEmptyAlt prop06* **by** *blast*
have *4*: $\vdash\ \ \mathit{empty} \wedge_i (\ \mathit{init}\ w) \equiv_i \mathit{empty} \wedge_i \mathit{init}\ w$ **by** *simp*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MLengthAlt*:
$\vdash\mathcal{M}(\mathit{LEN}(k)) \equiv_i \mathit{len}(k)$
proof −
have *1*: $\vdash \mathcal{M}(\mathit{LEN}(k)) \equiv_i \mathcal{M}(\mathit{FIRST}(\mathit{len}(k)))$ **by** (*simp add*:*mLEN-d-def*)
have *2*: $\vdash \mathcal{M}(\mathit{FIRST}(\mathit{len}(k))) \equiv_i \rhd(\mathit{len}(k))$ **by** *simp*
have *3*: $\vdash \rhd(\mathit{len}(k)) \equiv_i \mathit{len}(k)$ **using** *FstLenEqvLen* **by** *blast*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *BoxStateEqvBiFinState*:
$\vdash \Box\ (\mathit{init}\ w) \equiv_i \mathit{bi}\ (\mathit{fin}\ (\mathit{init}\ w))$
proof −
have *1*: $\vdash \Diamond\ (\neg_i\ (\mathit{init}\ w)) \equiv_i \mathit{true}_i\ ;\ \neg_i(\mathit{init}\ w)$
by *simp*
have *2*: $\vdash \Diamond\ (\mathit{init}(\neg_i\ w)) \equiv_i \mathit{true}_i\ ;\ \mathit{init}\ (\neg_i\ w)$
by *simp*
have *3*: $\vdash \mathit{di}\ (\mathit{true}_i \wedge_i \mathit{fin}\ (\mathit{init}\ (\neg_i\ w))) \equiv_i \mathit{true}_i\ ;\ \mathit{init}\ (\neg_i\ w)$
using *DiAndFinEqvChopState* **by** *blast*
have *4*: $\vdash \Diamond\ (\mathit{init}(\neg_i\ w)) \equiv_i \mathit{di}\ (\mathit{true}_i \wedge_i \mathit{fin}\ (\mathit{init}\ (\neg_i\ w)))$
using *1 2 3* **by** *simp*
have *5*: $\vdash \neg_i\ (\Diamond\ (\mathit{init}(\neg_i\ w))) \equiv_i \neg_i\ (\mathit{di}\ (\mathit{true}_i \wedge_i \mathit{fin}\ (\mathit{init}\ (\neg_i\ w))))$
using *4* **by** *simp*
have *6*: $\vdash \Box\ (\mathit{init}\ w) \equiv_i \neg_i\ (\mathit{di}\ (\mathit{true}_i \wedge_i \mathit{fin}\ (\mathit{init}\ (\neg_i\ w))))$
using *5* **by** *auto*
have *7*: $\vdash \Box\ (\mathit{init}\ w) \equiv_i \mathit{bi}\ (\neg_i\ (\mathit{fin}\ (\mathit{init}\ (\neg_i\ w))))$
using *6* **by** *auto*
have *8*: $\vdash \mathit{init}\ (\neg_i\ w) \equiv_i \neg_i\ (\mathit{init}\ w)$
by *simp*
have *9*: $\vdash \mathit{fin}\ (\mathit{init}\ (\neg_i\ w)) \equiv_i \mathit{fin}\ (\neg_i\ (\mathit{init}\ w))$
using *8 FinEqvFin* **by** *blast*
have *10*: $\vdash \mathit{fin}\ (\mathit{init}\ (\neg_i\ w)) \equiv_i \neg_i\ (\mathit{fin}\ (\mathit{init}\ w))$
using *8 FinNotStateEqvNotFinState FinEqvFin* **by** *blast*
have *11*: $\vdash \neg_i\ (\mathit{fin}\ (\mathit{init}\ (\neg_i\ w))) \equiv_i (\mathit{fin}\ (\mathit{init}\ w))$
using *10* **by** *simp*
have *12*: $\vdash \mathit{bi}\ (\neg_i\ (\mathit{fin}\ (\mathit{init}\ (\neg_i\ w)))) \equiv_i \mathit{bi}\ (\mathit{fin}\ (\mathit{init}\ w))$

    using *11* **by** *simp*
  have *13*: $\vdash \Box$ (*init w*) $\equiv_i$ *bi* (*fin* (*init w*))
    using *7 12* **by** *simp*
  from *13* **show** *?thesis* **by** *simp*
qed

lemma *MAlwaysAlt*:
 $\vdash \mathcal{M}$(*a ALWAYS w*) $\equiv_i \mathcal{M}$(*a*) $\wedge_i \Box$ (*init w*)
proof −
  have *1*: $\vdash \mathcal{M}$(*a ALWAYS w*) $\equiv_i \mathcal{M}$(*a WITH* (*bi* (*fin* (*init w*))))
    by (*simp add*: *mALWAYS-d-def*)
  have *2*: $\vdash \mathcal{M}$(*a WITH* (*bi* (*fin* (*init w*)))) $\equiv_i \mathcal{M}$(*a*) $\wedge_i$ (*bi* (*fin* (*init w*)))
    by *simp*
  have *3*: $\vdash \mathcal{M}$(*a*) $\wedge_i$ (*bi* (*fin* (*init w*))) $\equiv_i \mathcal{M}$(*a*) $\wedge_i \Box$ (*init w*)
    using *BoxStateEqvBiFinState* **by** *auto*
  from *1 2 3* **show** *?thesis* **by** *simp*
qed

lemma *DiamondStateEqvDiFinState*:
 $\vdash \Diamond$ (*init w*) $\equiv_i$ *di* (*fin* (*init w*))
proof −
  have *1*: $\vdash \Box$ (*init* ($\neg_i$ *w*)) $\equiv_i$ *bi* (*fin* (*init* ($\neg_i$ *w*))) **using** *BoxStateEqvBiFinState* **by** *blast*
  have *2*: $\vdash \neg_i$ ($\Box$ (*init* ($\neg_i$ *w*))) $\equiv_i \neg_i$ (*bi* (*fin* (*init* ($\neg_i$ *w*)))) **using** *1* **by** *auto*
  have *3*: $\vdash \Diamond$ ($\neg_i$ (*init* ($\neg_i$ *w*))) $\equiv_i$ *di* ($\neg_i$ (*fin* (*init* ($\neg_i$ *w*)))) **using** *2* **by** *auto*
  have *4*: $\vdash \Diamond$ (*init w*) $\equiv_i$ *di* ($\neg_i$ (*fin* (*init* ($\neg_i$ *w*)))) **using** *3* **by** *auto*
  have *5*: $\vdash \Diamond$ (*init w*) $\equiv_i$ *di* (*fin* (*init w*)) **using** *4 FinNotStateEqvNotFinState* **by** *auto*
  from *1 2 3 4 5* **show** *?thesis* **by** *simp*
qed

lemma *MSometimeAlt*:
 $\vdash \mathcal{M}$(*a SOMETIME w*) $\equiv_i \mathcal{M}$(*a*) $\wedge_i \Diamond$ (*init w*)
proof −
  have *1*: $\vdash \mathcal{M}$(*a SOMETIME w*) $\equiv_i \mathcal{M}$(*a WITH* (*di* (*fin* (*init w*))))
    by (*simp add*: *mSOMETIME-d-def*)
  have *2*: $\vdash\mathcal{M}$(*a WITH* (*di* (*fin* (*init w*)))) $\equiv_i \mathcal{M}$(*a*) $\wedge_i$ (*di* (*fin* (*init w*)))
    by *simp*
  have *3*: $\vdash\mathcal{M}$(*a WITH* (*di* (*fin* (*init w*)))) $\equiv_i \mathcal{M}$(*a*) $\wedge_i$ $\Diamond$ (*init w*)
    using *DiamondStateEqvDiFinState* **by** *auto*
  from *1 2 3* **show** *?thesis* **by** *simp*
qed

lemma *MWithinAlt*:
 $\vdash \mathcal{M}$(*a WITHIN f*) $\equiv_i \mathcal{M}$(*a*) $\wedge_i$ (*bs* ($\neg_i$ *f*))
proof −
  have *1*: $\vdash \mathcal{M}$(*a WITHIN f*) $\equiv_i \mathcal{M}$(*a WITH* (*bs* ($\neg_i$ *f*)))
    by (*simp add*: *mWITHIN-d-def mlimit-d-def*)
  have *2*: $\vdash \mathcal{M}$(*a WITH* (*bs* ($\neg_i$ *f*))) $\equiv_i \mathcal{M}$(*a*) $\wedge_i$ (*bs* ($\neg_i$ *f*))
    by *simp*
  from *1 2* **show** *?thesis* **by** *simp*
qed

lemma *MTimesAlt*:
$\vdash \mathcal{M}(a\ TIMES\ k) \equiv_i power\ (\mathcal{M}\ a)\ k$
proof
 (*induct k*)
 case *0*
 then show *?case*
  proof $-$
   have *1*: $\vdash \mathcal{M}\ \ a\ TIMES\ 0 \equiv_i \mathcal{M}\ EMPTY$ **by** *simp*
   have *2*: $\vdash \mathcal{M}\ EMPTY \equiv_i empty$ **using** *MEmptyAlt* **by** *simp*
   have *3*: $\vdash empty \equiv_i power\ (\mathcal{M}\ a)\ 0$ **by** *simp*
   from *1 2 3* **show** *?thesis* **by** *auto*
  qed
 next
 case (*Suc k*)
 then show *?case*
  proof $-$
   have *1*: $\vdash \mathcal{M}\ \ a\ TIMES\ Suc\ k \equiv_i \mathcal{M}(a\ THEN\ (a\ TIMES\ k)\ )$
     by *simp*
   have *2*: $\vdash \mathcal{M}(a\ THEN\ (a\ TIMES\ k)\ ) \equiv_i (\mathcal{M}\ a);(\mathcal{M}(a\ TIMES\ k))$
     by *simp*
   have *3*: $\vdash (\mathcal{M}\ a);(\mathcal{M}(a\ TIMES\ k)) \equiv_i (\mathcal{M}\ a);(power\ (\mathcal{M}\ a)\ k)$
     using *RightChopEqvChop Suc.hyps* **by** *blast*
   have *4*: $\vdash (\mathcal{M}\ a);(power\ (\mathcal{M}\ a)\ k) \equiv_i power\ (\mathcal{M}\ a)\ (Suc\ k)$
     by *simp*
   from *1 2 3 4* **show** *?thesis* **by** *auto*
  qed
 qed

lemma *MUptoAlt*:
$\vdash \mathcal{M}(a\ UPTO\ b) \equiv_i (\ (\mathcal{M}\ a) \wedge_i bi\ \neg_i(\mathcal{M}\ b)) \vee_i ((\mathcal{M}\ b) \wedge_i bi\ \neg_i(\mathcal{M}\ a)) \vee_i ((\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b))$
proof $-$
  have *1*: $\vdash \mathcal{M}(a\ UPTO\ b) \equiv_i \rhd((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))$
    by *simp*
  have *2*: $\vdash \rhd((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)) \equiv_i (\rhd(\mathcal{M}\ a) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ b))) \vee_i (\rhd(\mathcal{M}\ b) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ a)))$
    using *FstWithOrEqv* **by** *blast*
  have *3*: $\vdash (\rhd(\mathcal{M}\ a) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ b))) \vee_i (\rhd(\mathcal{M}\ b) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ a))) \equiv_i$
      $((\mathcal{M}\ a) \wedge_i ((\mathcal{M}\ b) \vee_i \neg_i(\mathcal{M}\ b)) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ b))) \vee_i$
      $((\mathcal{M}\ b) \wedge_i ((\mathcal{M}\ a) \vee_i \neg_i(\mathcal{M}\ a)) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ a)))$
    using *MFixFst* **by** *auto*
  have *4*: $\vdash ((\mathcal{M}\ a) \wedge_i ((\mathcal{M}\ b) \vee_i \neg_i(\mathcal{M}\ b)) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ b))) \vee_i$
      $((\mathcal{M}\ b) \wedge_i ((\mathcal{M}\ a) \vee_i \neg_i(\mathcal{M}\ a)) \wedge_i\ (bs\ \neg_i(\mathcal{M}\ a))) \equiv_i$
      $((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)) \vee_i (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)))\ ) \vee_i$
      $((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)) \vee_i (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)))\ )$
    by *auto*
  have *5*: $\vdash ((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)) \vee_i (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)))\ ) \vee_i$
      $((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)) \vee_i (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)))\ ) \equiv_i$
      $((\mathcal{M}\ a) \wedge_i (\ (\rhd(\mathcal{M}\ b)) \vee_i (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)))\ ) \vee_i$
      $((\mathcal{M}\ b) \wedge_i (\ (\rhd(\mathcal{M}\ a)) \vee_i (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)))\ )$
    by (*simp add*: *first-d-def*)

have *6*: $\vdash ((\mathcal{M}\ a) \wedge_i (\ (\rhd(\mathcal{M}\ b)) \vee_i (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)))\ ) \vee_i$
$((\mathcal{M}\ b) \wedge_i (\ (\rhd(\mathcal{M}\ a)) \vee_i (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)))\ ) \equiv_i$
$((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b)) \vee_i (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)))\ ) \vee_i$
$((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a)) \vee_i (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)))\ )$
using *MFixFst* **by** *auto*
have *7*: $\vdash (\neg_i(\mathcal{M}\ b) \wedge_i bs\neg_i(\mathcal{M}\ b)) \equiv_i bi(\neg_i(\mathcal{M}\ b))$
using *AndBsEqvBi* **by** *blast*
have *8*: $\vdash (\neg_i(\mathcal{M}\ a) \wedge_i bs\neg_i(\mathcal{M}\ a)) \equiv_i bi(\neg_i(\mathcal{M}\ a))$
using *AndBsEqvBi* **by** *blast*
have *9*: $\vdash ((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b)) \vee_i ((\neg_i(\mathcal{M}\ b)) \wedge_i bs(\neg_i(\mathcal{M}\ b))))\ ) \vee_i$
$((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a)) \vee_i ((\neg_i(\mathcal{M}\ a)) \wedge_i bs(\neg_i(\mathcal{M}\ a))))\ ) \equiv_i$
$((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b)) \vee_i (\ bi(\neg_i(\mathcal{M}\ b))))\ ) \vee_i$
$((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a)) \vee_i (\ bi(\neg_i(\mathcal{M}\ a))))\ )$
using *7 8* **by** *auto*
have *10*: $\vdash((\mathcal{M}\ a) \wedge_i (\ ((\mathcal{M}\ b)) \vee_i (\ bi(\neg_i(\mathcal{M}\ b))))\ ) \vee_i$
$((\mathcal{M}\ b) \wedge_i (\ ((\mathcal{M}\ a)) \vee_i (\ bi(\neg_i(\mathcal{M}\ a))))\ ) \equiv_i$
$((\ (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b)) \vee_i (\ (\mathcal{M}\ a) \wedge_i bi(\neg_i(\mathcal{M}\ b)))) \vee_i$
$((\ (\mathcal{M}\ b) \wedge_i (\mathcal{M}\ a)) \vee_i (\ (\mathcal{M}\ b) \wedge_i bi(\neg_i(\mathcal{M}\ a))))$
by *auto*
have *11*: $\vdash ((\ (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b)) \vee_i (\ (\mathcal{M}\ a) \wedge_i bi(\neg_i(\mathcal{M}\ b)))) \vee_i$
$((\ (\mathcal{M}\ b) \wedge_i (\mathcal{M}\ a)) \vee_i (\ (\mathcal{M}\ b) \wedge_i bi(\neg_i(\mathcal{M}\ a)))) \equiv_i$
$(\ (\mathcal{M}\ a) \wedge_i bi\ \neg_i(\mathcal{M}\ b)) \vee_i ((\mathcal{M}\ b) \wedge_i bi\ \neg_i(\mathcal{M}\ a)) \vee_i ((\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b))$
by *auto*
from *1 2 3 4 5 6 9 10 11* **show** *?thesis* **by** *auto*
qed

lemma *MThruAlt*:
$\vdash \mathcal{M}(a\ THRU\ b) \equiv_i ((\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i ((\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ a))$
proof −
have *1*: $\vdash \mathcal{M}(a\ THRU\ b) \equiv_i \rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))$
by *simp*
have *2*: $\vdash \rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \equiv_i (\rhd(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (\rhd(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ a))$
using *FstDiAndDiEqv* **by** *auto*
have *3*: $\vdash (\rhd(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (\rhd(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ a)) \equiv_i$
$((\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i ((\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ a))$
using *MFixFst* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MHaltAlt*:
$\vdash \mathcal{M}(HALT\ w) \equiv_i halt(init\ w)$
proof −
have *1*: $\vdash \mathcal{M}(HALT\ w) \equiv_i \mathcal{M}(FIRST\ (fin\ (init\ \ w)))$ **by** (*simp add*: *mHALT-d-def*)
have *2*: $\vdash \mathcal{M}(FIRST\ (fin\ (init\ \ w))) \equiv_i \rhd\ (fin\ (init\ \ w))$ **by** *simp*
have *3*: $\vdash \rhd\ (fin\ (init\ \ w)) \equiv_i halt(init\ w)$ **using** *HaltStateEqvFstFinState* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *simp*
qed

lemma *MFailUpto*:
$\vdash (\mathcal{M}\ (FAIL\ UPTO\ a)) \equiv_i (\mathcal{M}\ a)$

proof −
have *1*: $\vdash (\mathcal{M}\ (FAIL\ UPTO\ a)) \equiv_i \triangleright((\mathcal{M}\ FAIL) \vee_i (\mathcal{M}\ a))$ **by** *simp*
have *2*: $\vdash \mathcal{M}\ FAIL \vee_i \mathcal{M}\ a \equiv_i false_i \vee_i \mathcal{M}\ a$ **using** *MFailAlt* **by** *auto*
have *3*: $\vdash \triangleright((\mathcal{M}\ FAIL) \vee_i (\mathcal{M}\ a)) \equiv_i \triangleright(false_i \vee_i (\mathcal{M}\ a))$ **using** *2 FstEqvRule* **by** *blast*
have *4*: $\vdash false_i \vee_i (\mathcal{M}\ a) \equiv_i \mathcal{M}\ a$ **by** *simp*
have *5*: $\vdash \triangleright(false_i \vee_i (\mathcal{M}\ a)) \equiv_i \triangleright(\mathcal{M}\ a)$ **using** *4 FstEqvRule* **by** *blast*
have *6*: $\vdash \triangleright(\mathcal{M}\ a) \equiv_i \mathcal{M}\ a$ **using** *MFixFst* **by** *auto*
from *1 2 3 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *MFailThru*:
$\vdash \mathcal{M}(FAIL\ THRU\ a) \equiv_i \mathcal{M}\ FAIL$
proof −
have *1*: $\vdash \mathcal{M}(FAIL\ THRU\ a) \equiv_i \triangleright(di\ (\mathcal{M}\ FAIL) \wedge_i di(\mathcal{M}\ a))$
by *simp*
have *2*: $\vdash \triangleright(di\ (\mathcal{M}\ FAIL) \wedge_i di(\mathcal{M}\ a)) \equiv_i \triangleright(di\ (false_i) \wedge_i di(\mathcal{M}\ a))$
using *MFailAlt DiEqvDi FstEqvRule prop06* **by** *blast*
have *3*: $\vdash di\ false_i \equiv_i false_i$
by *simp*
hence *4*: $\vdash \triangleright(di\ (false_i) \wedge_i di(\mathcal{M}\ a)) \equiv_i \triangleright(\ (false_i) \wedge_i di(\mathcal{M}\ a))$
using *FstEqvRule prop06* **by** *blast*
have *5*: $\vdash \triangleright(\ (false_i) \wedge_i di(\mathcal{M}\ a)) \equiv_i \triangleright false_i$
using *FstEqvRule itl-prop*(*19*) **by** *blast*
have *6*: $\vdash \triangleright false_i \equiv_i false_i$ **using** *FstFalse*
by *auto*
have *7*: $\vdash false_i \equiv_i \mathcal{M}\ FAIL$
using *MFailAlt* **by** *auto*
from *1 2 4 5 6 7* **show** *?thesis* **by** *auto*
qed

lemma *MFailAnd*:
$\vdash \mathcal{M}\ (FAIL\ AND\ a) \equiv_i \mathcal{M}\ FAIL$
proof −
have *1*: $\vdash \mathcal{M}\ (FAIL\ AND\ a) \equiv_i (\mathcal{M}\ FAIL) \wedge_i (\mathcal{M}\ a)$ **by** (*simp add*: *mAND-d-def*)
have *2*: $\vdash (\mathcal{M}\ FAIL) \wedge_i (\mathcal{M}\ a) \equiv_i false_i \wedge_i (\mathcal{M}\ a)$ **using** *MFailAlt* **by** *auto*
have *3*: $\vdash false_i \wedge_i (\mathcal{M}\ a) \equiv_i false_i$ **by** *auto*
have *4*: $\vdash \mathcal{M}\ (FAIL\ AND\ a) \equiv_i false_i$ **using** *1 2 3* **by** *auto*
have *5*: $\vdash false_i \equiv_i \mathcal{M}\ FAIL$ **using** *MFailAlt* **by** *auto*
from *1 2 3 4 5* **show** *?thesis* **by** *auto*
qed

lemma *MThenFail*:
$\vdash \mathcal{M}\ (a\ THEN\ FAIL) \equiv_i \mathcal{M}\ FAIL$
proof −
have *1*: $\vdash \mathcal{M}\ (a\ THEN\ FAIL) \equiv_i (\mathcal{M}\ a);(\mathcal{M}\ FAIL)$ **by** *simp*
have *2*: $\vdash (\mathcal{M}\ a);(\mathcal{M}\ FAIL) \equiv_i (\mathcal{M}\ a);false_i$ **using** *MFailAlt* **by** *auto*
have *3*: $\vdash (\mathcal{M}\ a);false_i \equiv_i false_i$ **by** *auto*
have *4*: $\vdash false_i \equiv_i \mathcal{M}\ FAIL$ **using** *MFailAlt* **by** *auto*
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MFailThen*:
$\vdash \mathcal{M}$ (*FAIL THEN a*) $\equiv_i$ $\mathcal{M}$ *FAIL*
proof −
  have *1*: $\vdash \mathcal{M}$ (*FAIL THEN a*) $\equiv_i$ ($\mathcal{M}$ *FAIL*);($\mathcal{M}$ *a*) **by** *simp*
  have *2*: $\vdash$ ($\mathcal{M}$ *FAIL*);($\mathcal{M}$ *a*) $\equiv_i$ *false*$_i$;($\mathcal{M}$ *a*) **using** *MFailAlt* **by** *auto*
  have *3*: $\vdash$ *false*$_i$;($\mathcal{M}$ *a*) $\equiv_i$ *false*$_i$ **by** *auto*
  have *4*: $\vdash$ *false*$_i$ $\equiv_i$ $\mathcal{M}$ *FAIL* **using** *MFailAlt* **by** *auto*
 from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MFailWith*:
$\vdash \mathcal{M}$ (*FAIL WITH f*) $\equiv_i$ $\mathcal{M}$ *FAIL*
proof −
  have *1*: $\vdash \mathcal{M}$ (*FAIL WITH f*) $\equiv_i$ ($\mathcal{M}$ *FAIL*) $\wedge_i$ *f* **by** *simp*
  have *2*: $\vdash$ ($\mathcal{M}$ *FAIL*) $\wedge_i$ *f* $\equiv_i$ *false*$_i$ $\wedge_i$ *f* **using** *MFailAlt* **by** *auto*
  have *3*: $\vdash$ *false*$_i$ $\wedge_i$ *f* $\equiv_i$ *false*$_i$ **by** *simp*
  have *4*: $\vdash$ *false*$_i$ $\equiv_i$ $\mathcal{M}$ *FAIL* **using** *MFailAlt* **by** *auto*
 from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MEmptyUpto*:
$\vdash \mathcal{M}$ (*EMPTY UPTO a*) $\equiv_i$ $\mathcal{M}$ *EMPTY*
proof −
  have *1*: $\vdash \mathcal{M}$ (*EMPTY UPTO a*) $\equiv_i$ $\triangleright$(($\mathcal{M}$ *EMPTY*) $\vee_i$ ($\mathcal{M}$ *a*)) **by** *simp*
  have *2*: $\vdash$ ($\mathcal{M}$ *EMPTY*) $\equiv_i$ *empty* **using** *MEmptyAlt* **by** *auto*
 hence *3*: $\vdash$ ($\mathcal{M}$ *EMPTY*) $\vee_i$ ($\mathcal{M}$ *a*) $\equiv_i$ *empty* $\vee_i$ ($\mathcal{M}$ *a*) **by** *auto*
 hence *4*: $\vdash$ $\triangleright$(($\mathcal{M}$ *EMPTY*) $\vee_i$ ($\mathcal{M}$ *a*)) $\equiv_i$ $\triangleright$(*empty* $\vee_i$ ($\mathcal{M}$ *a*)) **using** *FstEqvRule* **by** *blast*
  have *5*: $\vdash$ $\triangleright$(*empty* $\vee_i$ ($\mathcal{M}$ *a*)) $\equiv_i$ *empty* **using** *FstEmptyOrEqvEmpty* **by** *blast*
  have *6*: $\vdash$ *empty* $\equiv_i$ ($\mathcal{M}$ *EMPTY*) **using** *MEmptyAlt* **by** *auto*
 from *1 4 5 6* **show** *?thesis* **by** *auto*
qed

lemma *MEmptyThru*:
$\vdash \mathcal{M}$ (*EMPTY THRU a*) $\equiv_i$ ($\mathcal{M}$ *a*)
proof −
  have *1*: $\vdash \mathcal{M}$ (*EMPTY THRU a*) $\equiv_i$ $\triangleright$(*di*($\mathcal{M}$ *EMPTY*) $\wedge_i$ *di*($\mathcal{M}$ *a*)) **by** *simp*
  have *2*: $\vdash$ *di*($\mathcal{M}$ *EMPTY*) $\equiv_i$ *di empty* **using** *MEmptyAlt DiEqvDi* **by** *blast*
 hence *3*: $\vdash$ *di*($\mathcal{M}$ *EMPTY*) $\wedge_i$ *di*($\mathcal{M}$ *a*) $\equiv_i$ *di empty* $\wedge_i$ *di*($\mathcal{M}$ *a*) **by** *auto*
 hence *4*: $\vdash$ *di empty* $\wedge_i$ *di*($\mathcal{M}$ *a*) $\equiv_i$ *di*($\mathcal{M}$ *a*) **by** *auto*
  have *5*: $\vdash$ *di*($\mathcal{M}$ *EMPTY*) $\wedge_i$ *di*($\mathcal{M}$ *a*) $\equiv_i$ *di*($\mathcal{M}$ *a*) **using** *3 4* **by** *auto*
 hence *6*: $\vdash$ $\triangleright$(*di*($\mathcal{M}$ *EMPTY*) $\wedge_i$ *di*($\mathcal{M}$ *a*)) $\equiv_i$ $\triangleright$(*di*($\mathcal{M}$ *a*)) **using** *FstEqvRule* **by** *blast*
  have *7*: $\vdash$ $\triangleright$(*di*($\mathcal{M}$ *a*)) $\equiv_i$ $\triangleright$($\mathcal{M}$ *a*) **using** *FstDiEqvFst* **by** *blast*
  have *8*: $\vdash$ $\triangleright$($\mathcal{M}$ *a*) $\equiv_i$ ($\mathcal{M}$ *a*) **using** *MFixFst* **by** *auto*
 from *1 6 7 8* **show** *?thesis* **by** *auto*
qed

lemma *MThenEmpty*:
$\vdash \mathcal{M}$ (*a THEN EMPTY*) $\equiv_i$ ($\mathcal{M}$ *a*)
proof −

have *1*: $\vdash \mathcal{M}$ (*a THEN EMPTY*) $\equiv_i$ ($\mathcal{M}$ *a*);($\mathcal{M}$ *EMPTY*) **by** *simp*
have *2*: $\vdash$ ($\mathcal{M}$ *a*);($\mathcal{M}$ *EMPTY*) $\equiv_i$ ($\mathcal{M}$ *a*); *empty* **using** *MEmptyAlt* **by** *auto*
have *3*: $\vdash$ ($\mathcal{M}$ *a*); *empty* $\equiv_i$ ($\mathcal{M}$ *a*) **using** *ChopEmpty* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MEmptyThen*:
$\vdash \mathcal{M}$ (*EMPTY THEN a*) $\equiv_i$ ($\mathcal{M}$ *a*)
proof $-$
have *1*: $\vdash \mathcal{M}$ (*EMPTY THEN a*) $\equiv_i$ ($\mathcal{M}$ *EMPTY*);($\mathcal{M}$ *a*) **by** *simp*
have *2*: $\vdash$ ($\mathcal{M}$ *EMPTY*);($\mathcal{M}$ *a*) $\equiv_i$ *empty*; ($\mathcal{M}$ *a*) **using** *MEmptyAlt* **by** *auto*
have *3*: $\vdash$ *empty*;($\mathcal{M}$ *a*) $\equiv_i$ ($\mathcal{M}$ *a*) **using** *ChopEmpty* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MEmptyIterate*:
$\vdash \mathcal{M}$ (*EMPTY ITERATE b*) $\equiv_i$ ($\mathcal{M}$ *EMPTY*)
proof $-$
have *1*: $\vdash \mathcal{M}$ (*EMPTY ITERATE b*) $\equiv_i$ $\mathcal{M}$ (*EMPTY WITH* ($\mathcal{M}$ *b*)*)
by (*simp add*: *mITERATE-d-def*)
have *2*: $\vdash \mathcal{M}$ (*EMPTY WITH* ($\mathcal{M}$ *b*)*) $\equiv_i$ $\mathcal{M}$ *EMPTY* $\wedge_i$ ($\mathcal{M}$ *b*)*
by *simp*
have *3*: $\vdash \mathcal{M}$ *EMPTY* $\wedge_i$ ($\mathcal{M}$ *b*)* $\equiv_i$ *empty* $\wedge_i$ ($\mathcal{M}$ *b*)*
using *MEmptyAlt* **by** *auto*
have *4*: $\vdash$ *empty* $\wedge_i$ ($\mathcal{M}$ *b*)* $\equiv_i$ *empty* $\wedge_i$ (*empty* $\vee_i$ ((($\mathcal{M}$ *b*) $\wedge_i$ *more*);($\mathcal{M}$ *b*)*))
using *ChopstarEqv* **by** *auto*
have *5*: $\vdash$ *empty* $\wedge_i$ (*empty* $\vee_i$ ((($\mathcal{M}$ *b*) $\wedge_i$ *more*);($\mathcal{M}$ *b*)*)) $\equiv_i$ *empty*
by *auto*
have *6*: $\vdash \mathcal{M}$ (*EMPTY ITERATE b*) $\equiv_i$ $\mathcal{M}$ *EMPTY*
using *1 2 3 4 5 MEmptyAlt* **by** *auto*
from *6* **show** *?thesis* **by** *simp*
qed

lemma *FstAndFstStarEqvFst*:
$\vdash \rhd f \wedge_i (\rhd f)^\star \equiv_i \rhd f$
proof $-$
have *1*: $\vdash (\rhd f)^\star \equiv_i$ *empty* $\vee_i (\rhd f);(\rhd f)^\star$ **using** *CSEqvOrChopCS* **by** *simp*
have *2*: $\vdash (\rhd f)^\star \wedge_i \rhd f \equiv_i$ (*empty* $\vee_i (\rhd f);(\rhd f)^\star) \wedge_i \rhd f$ **using** *1 prop06* **by** *blast*
have *3*: $\vdash$ (*empty* $\vee_i (\rhd f);(\rhd f)^\star) \wedge_i \rhd f \equiv_i$ (*empty* $\wedge_i \rhd f) \vee_i ((\rhd f);(\rhd f)^\star \wedge_i \rhd f)$ **by** *auto*
have *4*: $\vdash (\rhd f)^\star \wedge_i \rhd f \equiv_i$ (*empty* $\wedge_i \rhd f) \vee_i ((\rhd f);(\rhd f)^\star \wedge_i \rhd f)$ **using** *2 3* **by** *simp*
have *5*: $\vdash (\rhd f);(\rhd f)^\star \wedge_i \rhd f \equiv_i (\rhd f);(\rhd f)^\star \wedge_i \rhd f$;*empty* **using** *ChopEmpty* **by** *auto*
have *6*: $\vdash (\rhd f);(\rhd f)^\star \wedge_i \rhd f$;*empty* $\equiv_i (\rhd f);((\rhd f)^\star \wedge_i$ *empty*) **using** *LFstAndDistrC* **by** *blast*
have *7*: $\vdash (\rhd f)^\star \wedge_i$ *empty* $\equiv_i$ *empty* **using** *EmptyImpCS* **by** *auto*
have *8*: $\vdash (\rhd f);((\rhd f)^\star \wedge_i$ *empty*) $\equiv_i \rhd f$ **using** *7 RightChopEqvChop ChopEmpty* **by** *auto*
have *9*: $\vdash (\rhd f);(\rhd f)^\star \wedge_i \rhd f \equiv_i \rhd f$ **using** *5 6 8* **by** *simp*
have *10*: $\vdash (\rhd f)^\star \wedge_i \rhd f \equiv_i$ (*empty* $\wedge_i \rhd f) \vee_i \rhd f$ **using** *4 9* **by** *simp*
have *11*: $\vdash$ (*empty* $\wedge_i \rhd f) \vee_i \rhd f \equiv_i \rhd f$ **by** *auto*
have *12*: $\vdash (\rhd f)^\star \wedge_i \rhd f \equiv_i \rhd f$ **using** *10 11* **by** *simp*
from *12* **show** *?thesis* **by** *auto*
qed

lemma *MIterateIdemp*:
$\vdash \mathcal{M}$ (*a ITERATE a*) $\equiv_i \mathcal{M}$ *a*
proof −
have *1*: $\vdash \mathcal{M}$ (*a ITERATE a*) $\equiv_i \mathcal{M}$ (*a WITH* $(\mathcal{M}\ a)^\star$) **by** (*simp add*: *mITERATE-d-def*)
have *2*: $\vdash \mathcal{M}$ (*a WITH* $(\mathcal{M}\ a)^\star$) $\equiv_i \mathcal{M}\ a \wedge_i (\mathcal{M}\ a)^\star$ **by** *simp*
have *3*: $\vdash \mathcal{M}\ a \wedge_i (\mathcal{M}\ a)^\star \equiv_i \triangleright(\mathcal{M}\ a) \wedge_i (\triangleright(\mathcal{M}\ a))^\star$ **using** *MFixFst* **by** *auto*
have *4*: $\vdash \triangleright(\mathcal{M}\ a) \wedge_i (\triangleright(\mathcal{M}\ a))^\star \equiv_i \triangleright(\mathcal{M}\ a)$ **using** *FstAndFstStarEqvFst* **by** *simp*
have *5*: $\vdash \triangleright(\mathcal{M}\ a) \equiv_i \mathcal{M}\ a$ **using** *MFixFst* **by** *auto*
from *1 2 3 4 5* **show** *?thesis* **using** *prop03* **by** *blast*
qed

lemma *MUptoIdemp*:
$\vdash \mathcal{M}$ (*a UPTO a*) $\equiv_i \mathcal{M}$ *a*
proof −
have *1*: $\vdash \mathcal{M}$ (*a UPTO a*) $\equiv_i \triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ a))$ **by** *simp*
have *2*: $\vdash \triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ a)) \equiv_i \triangleright(\mathcal{M}\ a)$ **using** *FstEqvRule itl-prop*(*27*) **by** *blast*
have *3*: $\vdash \triangleright(\mathcal{M}\ a) \equiv_i (\mathcal{M}\ a)$ **using** *MFixFst* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *simp*
qed

lemma *MThruIdemp*:
$\vdash \mathcal{M}$ (*a THRU a*) $\equiv_i (\mathcal{M}\ a)$
proof −
have *1*: $\vdash \mathcal{M}$ (*a THRU a*) $\equiv_i \triangleright($ *di*$(\mathcal{M}\ a) \wedge_i$ *di*$(\mathcal{M}\ a))$ **by** *simp*
have *2*: $\vdash \triangleright($ *di*$(\mathcal{M}\ a) \wedge_i$ *di*$(\mathcal{M}\ a)) \equiv_i \triangleright($*di* $(\mathcal{M}\ a))$ **using** *FstEqvRule itl-prop*(*20*) **by** *blast*
have *3*: $\vdash \triangleright($*di* $(\mathcal{M}\ a)) \equiv_i \triangleright(\mathcal{M}\ a)$ **using** *FstDiEqvFst* **by** *blast*
have *4*: $\vdash \triangleright(\mathcal{M}\ a) \equiv_i (\mathcal{M}\ a)$ **using** *MFixFst* **by** *auto*
from *1 2 3 4* **show** *?thesis* **by** *simp*
qed

lemma *MAndIdemp*:
$\vdash \mathcal{M}$(*a AND a*) $\equiv_i (\mathcal{M}\ a)$
proof −
have *1*: $\vdash \mathcal{M}$(*a AND a*) $\equiv_i (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ a)$ **by** (*simp add*: *mAND-d-def*)
have *2*: $\vdash (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ a) \equiv_i (\mathcal{M}\ a)$ **by** *auto*
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MWithIdemp*:
$\vdash \mathcal{M}$ ((*a WITH f*) *WITH f*) $\equiv_i \mathcal{M}$(*a WITH f*)
proof −
have *1*: $\vdash \mathcal{M}$ ((*a WITH f*) *WITH f*) $\equiv_i ((\mathcal{M}\ a) \wedge_i (\ f)) \wedge_i (\ f)$ **by** *simp*
have *2*: $\vdash ((\mathcal{M}\ a) \wedge_i (\ f)) \wedge_i (\ f) \equiv_i (\mathcal{M}\ a) \wedge_i (\ f)$ **by** *auto*
have *3*: $\vdash (\mathcal{M}\ a) \wedge_i (\ f) \equiv_i \mathcal{M}$(*a WITH f*) **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MUptoCommut*:
$\vdash \mathcal{M}$(*a UPTO b*) $\equiv_i \mathcal{M}$(*b UPTO a*)

proof −
  have *1*: $\vdash \mathcal{M}(a\ UPTO\ b) \equiv_i \rhd((\mathcal{M}\ a) \lor_i (\mathcal{M}\ b))$ **by** *simp*
  have *2*: $\vdash ((\mathcal{M}\ a) \lor_i (\mathcal{M}\ b)) \equiv_i ((\mathcal{M}\ b) \lor_i (\mathcal{M}\ a))$ **by** *auto*
 hence *3*: $\vdash \rhd((\mathcal{M}\ a) \lor_i (\mathcal{M}\ b)) \equiv_i \rhd((\mathcal{M}\ b) \lor_i (\mathcal{M}\ a))$ **using** *FstEqvRule* **by** *blast*
  have *4*: $\vdash \rhd((\mathcal{M}\ b) \lor_i (\mathcal{M}\ a)) \equiv_i \mathcal{M}(b\ UPTO\ a)$ **by** *simp*
 from *1 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MThruCommut*:
$\vdash \mathcal{M}(a\ THRU\ b) \equiv_i \mathcal{M}(b\ THRU\ a)$
proof −
  have *1*: $\vdash \mathcal{M}(a\ THRU\ b) \equiv_i \rhd(di(\mathcal{M}\ a) \land_i di(\mathcal{M}\ b))$ **by** *simp*
  have *2*: $\vdash (di(\mathcal{M}\ a) \land_i di(\mathcal{M}\ b)) \equiv_i (di(\mathcal{M}\ b) \land_i di(\mathcal{M}\ a))$ **by** *auto*
 hence *3*: $\vdash \rhd(di(\mathcal{M}\ a) \land_i di(\mathcal{M}\ b)) \equiv_i \rhd(di(\mathcal{M}\ b) \land_i di(\mathcal{M}\ a))$ **using** *FstEqvRule* **by** *blast*
  have *4*: $\vdash \rhd(di(\mathcal{M}\ b) \land_i di(\mathcal{M}\ a)) \equiv_i \mathcal{M}(b\ THRU\ a)$ **by** *simp*
 from *1 3 4* **show** *?thesis* **by** *auto*
qed

lemma *MAndCommut*:
$\vdash \mathcal{M}(a\ AND\ b) \equiv_i \mathcal{M}(b\ AND\ a)$
proof −
  have *1*: $\vdash \mathcal{M}(a\ AND\ b) \equiv_i (\mathcal{M}\ a) \land_i (\mathcal{M}\ b)$ **by** (*simp add*: *mAND-d-def*)
  have *2*: $\vdash (\mathcal{M}\ a) \land_i (\mathcal{M}\ b) \equiv_i (\mathcal{M}\ b) \land_i (\mathcal{M}\ a)$ **by** *auto*
  have *3*: $\vdash (\mathcal{M}\ b) \land_i (\mathcal{M}\ a) \equiv_i \mathcal{M}(b\ AND\ a)$ **by** (*simp add*: *mAND-d-def*)
 from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MWithCommut*:
$\vdash \mathcal{M}((a\ WITH\ f)\ WITH\ g) \equiv_i \mathcal{M}((a\ WITH\ g)\ WITH\ f)$
proof −
  have *1*: $\vdash \mathcal{M}((a\ WITH\ f)\ WITH\ g) \equiv_i (\mathcal{M}\ a) \land_i (f) \land_i (g)$ **by** *simp*
  have *2*: $\vdash (\mathcal{M}\ a) \land_i (f) \land_i (g) \equiv_i \mathcal{M}((a\ WITH\ g)\ WITH\ f)$ **by** *auto*
 from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MWithAbsorp*:
$\vdash \mathcal{M}((a\ WITH\ f)\ WITH\ g) \equiv_i \mathcal{M}(a\ WITH\ (f \land_i g))$
proof −
  have *1*: $\vdash \mathcal{M}((a\ WITH\ f)\ WITH\ g) \equiv_i (\mathcal{M}\ a) \land_i (f) \land_i (g)$ **by** *simp*
  have *2*: $\vdash (\mathcal{M}\ a) \land_i (f) \land_i (g) \equiv_i (\mathcal{M}\ a) \land_i (f \land_i g)$ **by** *auto*
 from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MUptoAssoc*:
$\vdash \mathcal{M}((a\ UPTO\ b)\ UPTO\ c) \equiv_i \mathcal{M}(a\ UPTO\ (b\ UPTO\ c))$
proof −
  have *1*: $\vdash \mathcal{M}((a\ UPTO\ b)\ UPTO\ c) \equiv_i \rhd(\mathcal{M}(a\ UPTO\ b) \lor_i (\mathcal{M}\ c))$
      by *simp*
  have *2*: $\vdash \rhd(\mathcal{M}(a\ UPTO\ b) \lor_i (\mathcal{M}\ c)) \equiv_i \rhd(\rhd((\mathcal{M}\ a) \lor_i (\mathcal{M}\ b)) \lor_i (\mathcal{M}\ c))$
      by *simp*

have 3: $\vdash \rhd(\rhd((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)) \vee_i (\mathcal{M}\ c)) \equiv_i \rhd(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)) \vee_i (\mathcal{M}\ c))$
using *FstFstOrEqvFstOrL* **by** *blast*
have 4: $\vdash (((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)) \vee_i (\mathcal{M}\ c)) \equiv_i ((\mathcal{M}\ a) \vee_i ((\mathcal{M}\ b) \vee_i (\mathcal{M}\ c)))$
by *auto*
hence 5: $\vdash \rhd(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)) \vee_i (\mathcal{M}\ c)) \equiv_i \rhd((\mathcal{M}\ a) \vee_i ((\mathcal{M}\ b) \vee_i (\mathcal{M}\ c)))$
using *FstEqvRule* **by** *blast*
have 6: $\vdash \rhd((\mathcal{M}\ a) \vee_i ((\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))) \equiv_i \rhd((\mathcal{M}\ a) \vee_i \rhd((\mathcal{M}\ b) \vee_i (\mathcal{M}\ c)))$
using *FstFstOrEqvFstOrR itl-prop(30)* **by** *blast*
have 7: $\vdash \rhd((\mathcal{M}\ a) \vee_i \rhd((\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))) \equiv_i \rhd((\mathcal{M}\ a) \vee_i \mathcal{M}(b\ UPTO\ c))$
by *simp*
have 8: $\vdash \rhd((\mathcal{M}\ a) \vee_i \mathcal{M}(b\ UPTO\ c)) \equiv_i \mathcal{M}(a\ UPTO\ (b\ UPTO\ c))$
by *simp*
from *1 2 3 5 6 7 8* **show** *?thesis* **by** *auto*
qed

lemma *MThruAssoc*:
$\vdash \mathcal{M}((a\ THRU\ b)\ THRU\ c) \equiv_i \mathcal{M}(a\ THRU\ (b\ THRU\ c))$
proof $-$
have 1: $\vdash \mathcal{M}((a\ THRU\ b)\ THRU\ c) \equiv_i \rhd(di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \wedge_i di(\mathcal{M}\ c))$
by *simp*
have 2: $\vdash di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i di((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *DiEqvDiFst itl-prop(30)* **by** *blast*
have 3: $\vdash di((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)$
using *DiDiAndEqvDi* **by** *blast*
have 4: $\vdash di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)$
using *2 3* **by** *auto*
hence 5: $\vdash di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \wedge_i di(\mathcal{M}\ c) \equiv_i di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)$
by *auto*
have 6: $\vdash di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c) \equiv_i di\ (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))$
using *DiDiAndEqvDi itl-prop(30)* **by** *blast*
have 7: $\vdash di\ (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)) \equiv_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))$
using *DiEqvDiFst* **by** *blast*
have 8: $\vdash di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c) \equiv_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))$
using *6 7* **using** *prop03* **by** *blast*
hence 9: $\vdash di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c) \equiv_i di(\mathcal{M}\ a) \wedge_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))$
by *auto*
have 10: $\vdash di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \wedge_i di(\mathcal{M}\ c) \equiv_i$
$di(\mathcal{M}\ a) \wedge_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))$
using *5 9* **by** *auto*
hence 11: $\vdash \rhd(di(\rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \wedge_i di(\mathcal{M}\ c)) \equiv_i$
$\rhd(di(\mathcal{M}\ a) \wedge_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))))$
using *FstEqvRule* **by** *blast*
have 12: $\vdash \rhd(di(\mathcal{M}\ a) \wedge_i di\ (\rhd(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))) \equiv_i \mathcal{M}(a\ THRU\ (b\ THRU\ c))$
by *simp*
from *1 11 12* **show** *?thesis* **by** *auto*
qed

lemma *MAndAssoc*:
$\vdash \mathcal{M}((a\ AND\ b)\ AND\ c) \equiv_i \mathcal{M}(a\ AND\ (b\ AND\ c))$
proof $-$

have *1*: $\vdash \mathcal{M}((a\ AND\ b)\ AND\ c) \equiv_i (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b) \wedge_i (\mathcal{M}\ c)$ **by** (*simp add*: *mAND-d-def*)
have *2*: $\vdash (\mathcal{M}\ a) \wedge_i (\mathcal{M}\ b) \wedge_i (\mathcal{M}\ c) \equiv_i \mathcal{M}(a\ AND\ (b\ AND\ c))$ **by** (*simp add*: *mAND-d-def*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *MThenAssoc*:
$\vdash \mathcal{M}((a\ THEN\ b)\ THEN\ c) \equiv_i \mathcal{M}(a\ THEN\ (b\ THEN\ c))$
proof –
have *1*: $\vdash \mathcal{M}((a\ THEN\ b)\ THEN\ c) \equiv_i ((\mathcal{M}\ a);(\mathcal{M}\ b));(\mathcal{M}\ c)$ **by** *simp*
have *2*: $\vdash ((\mathcal{M}\ a);(\mathcal{M}\ b));(\mathcal{M}\ c) \equiv_i (\mathcal{M}\ a);((\mathcal{M}\ b);(\mathcal{M}\ c))$ **using** *ChopAssocB* **by** *blast*
have *3*: $\vdash (\mathcal{M}\ a);((\mathcal{M}\ b);(\mathcal{M}\ c)) \equiv_i \mathcal{M}(a\ THEN\ (b\ THEN\ c))$ **by** *simp*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *OrDiEqvDi*:
$\vdash f \vee_i di\ f \equiv_i di\ f$
proof –
have *1*: $\vdash f \supset_i di\ f$ **using** *DiIntro* **by** *blast*
from *1* **show** *?thesis* **by** *auto*
qed

lemma *AndDiEqv*:
$\vdash f \wedge_i di\ f \equiv_i f$
proof –
have *1*: $\vdash f \supset_i di\ f$ **using** *DiIntro* **by** *blast*
from *1* **show** *?thesis* **by** *auto*
qed

lemma *MUptoThruAbsorp*:
$\vdash \mathcal{M}(a\ UPTO\ (a\ THRU\ b)) \equiv_i \mathcal{M}\ a$
proof –
have *1*: $\vdash \mathcal{M}(a\ UPTO\ (a\ THRU\ b)) \equiv_i \rhd((\mathcal{M}\ a) \vee_i \rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ ))$
by *simp*
have *2*: $\vdash \rhd((\mathcal{M}\ a) \vee_i \rhd(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ )) \equiv_i$
$\rhd((\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ ))$
using *FstFstOrEqvFstOrR* **by** *auto*
have *3*: $\vdash ((\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ )) \equiv_i$
$(((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
by *auto*
have *4*: $\vdash (((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
using *OrDiEqvDi* **by** *auto*
have *5*: $\vdash ((\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ )) \equiv_i$
$((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
using *3 4* **by** *auto*
hence *6*: $\vdash \rhd\ ((\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)\ )) \equiv_i$
$\rhd\ ((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
using *FstEqvRule* **by** *blast*
have *7*: $\vdash \rhd\ ((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$(di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$

$bs\ \neg_i(\ (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
by (*simp add*: *first-d-def*)
have *8*: $\vdash (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \equiv_i$
$(di(\mathcal{M}\ a) \wedge_i (\mathcal{M}\ a)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))$
by *auto*
hence *9*: $\vdash \neg_i((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$\neg_i((di(\mathcal{M}\ a) \wedge_i (\mathcal{M}\ a)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *prop01* **by** *blast*
have *10*: $\vdash \neg_i((di(\mathcal{M}\ a) \wedge_i (\mathcal{M}\ a)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$\neg_i(((\mathcal{M}\ a)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *AndDiEqv* **by** *auto*
have *11*: $\vdash \neg_i(((\mathcal{M}\ a)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$\neg_i(\mathcal{M}\ a) \wedge_i \neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))$
by *auto*
have *12*: $\vdash \neg_i((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$\neg_i(\mathcal{M}\ a) \wedge_i \neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))$
using *9 10 11* **by** *auto*
hence *13*: $\vdash bs\ \neg_i((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$bs\ (\neg_i(\mathcal{M}\ a) \wedge_i \neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *BsEqvRule* **by** *blast*
have *14*: $\vdash bs\ ((\neg_i(\mathcal{M}\ a)) \wedge_i \neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *BsAndEqv* **using** *itl-prop*(*30*) **by** *blast*
have *141*: $\vdash bs\ \neg_i((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *13 14* **by** *auto*
hence *15*: $\vdash (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs\ \neg_i((di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$(di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
by *auto*
have *16*: $\vdash (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$(bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
by *auto*
have *17*: $\vdash (bs\ ((\neg_i(\mathcal{M}\ a))) \wedge_i di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$(\triangleright(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *FstEqvBsNotAndDi itl-prop*(*30*) *prop06* **by** *blast*
have *18*: $\vdash (\triangleright(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$((\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *MFixFst itl-prop*(*30*) *prop06* **by** *blast*
have *19*: $\vdash ((\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$((\mathcal{M}\ a)) \wedge_i\ \ bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
by *auto*

have *20*: $\vdash\ (\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i (\neg_i(di(\mathcal{M}\ a)) \vee_i \neg_i(di(\mathcal{M}\ b)))$
by *auto*
have *21*: $\vdash (\neg_i(di(\mathcal{M}\ a)) \vee_i \neg_i(di(\mathcal{M}\ b))) \equiv_i ((bi\ \neg_i(\mathcal{M}\ a)) \vee_i (bi \neg_i(\mathcal{M}\ b)))$
by *auto*
have *22*: $\vdash (\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i ((bi\ \neg_i(\mathcal{M}\ a)) \vee_i (bi \neg_i(\mathcal{M}\ b)))$
using *20 21* **by** *auto*
hence *23*: $\vdash bs\ (\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i bs\ ((bi\ \neg_i(\mathcal{M}\ a)) \vee_i (bi \neg_i(\mathcal{M}\ b)))$
using *BsEqvRule* **by** *blast*
have *24*: $\vdash bs\ ((bi\ \neg_i(\mathcal{M}\ a)) \vee_i (bi \neg_i(\mathcal{M}\ b))) \equiv_i bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b))$
using *BsOrBsEqvBsBiOrBi itl-prop*(*30*) **by** *blast*
have *25*: $\vdash bs\ (\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b))$
using *23 24* **by** *auto*
hence *26*: $\vdash (\mathcal{M}\ a) \wedge_i bs\ (\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i$
$(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b)))$
by *auto*
have *27*: $\vdash(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b))) \equiv_i$
$\rhd(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b)))$
using *MFixFst prop06* **by** *blast*
have *28*: $\vdash \rhd(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b))) \equiv_i$
$(\mathcal{M}\ a) \wedge_i bs\ \neg_i(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b)))$
by (*simp add*: *first-d-def*)
have *29*: $\vdash (\mathcal{M}\ a) \wedge_i bs\ \neg_i(\mathcal{M}\ a) \wedge_i (bs\ (\neg_i(\mathcal{M}\ a)) \vee_i bs\ (\neg_i(\mathcal{M}\ b))) \equiv_i$
$(\mathcal{M}\ a) \wedge_i bs\ \neg_i(\mathcal{M}\ a)$
by *auto*
have *30*: $\vdash (\mathcal{M}\ a) \wedge_i bs\ \neg_i(\mathcal{M}\ a) \equiv_i \rhd(\mathcal{M}\ a)$
by (*simp add*: *first-d-def*)
have *31*: $\vdash \rhd(\mathcal{M}\ a) \equiv_i (\mathcal{M}\ a)$
using *MFixFst* **by** *auto*
have *32*: $\vdash \mathcal{M}(a\ UPTO\ (a\ THRU\ b)) \equiv_i$
$(di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs\ \neg_i(\ (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
using *1 2 6 7* **by** *auto*
have *33*: $\vdash (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i$
$bs\ \neg_i(\ (di(\mathcal{M}\ a)) \wedge_i ((\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i$
$((\mathcal{M}\ a)) \wedge_i\ bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)))$
using *15 16 17 18 19* **by** *auto*
have *34*: $\vdash ((\mathcal{M}\ a)) \wedge_i\ bs(\neg_i(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b))) \equiv_i (\mathcal{M}\ a)$
using *26 27 28 29 30 31* **using** *prop03* **by** *blast*
from *32 33 34* **show** *?thesis* **by** *auto*
qed

lemma *MThruUptoAbsorp*:
$\vdash \mathcal{M}(a\ THRU\ (a\ UPTO\ b)) \equiv_i (\mathcal{M}\ a)$
proof –
have *1*: $\vdash \mathcal{M}(a\ THRU\ (a\ UPTO\ b)) \equiv_i \rhd(di(\mathcal{M}\ a) \wedge_i di(\rhd((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))))$
by *simp*
have *2*: $\vdash\ \rhd(di(\mathcal{M}\ a) \wedge_i di(\rhd((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)))) \equiv_i$
$\rhd(di(\mathcal{M}\ a) \wedge_i di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))))$
using *DiEqvDiFst* **using** *FstEqvRule itl-prop*(*30*) *prop05* **by** *blast*
have *3*: $\vdash \rhd(di(\mathcal{M}\ a) \wedge_i di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b)))) \equiv_i$

$\triangleright(di(\mathcal{M}\ a) \wedge_i (di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)))$
using *DiOrEqv* **using** *FstEqvRule prop05* **by** *blast*
have *4*: $\vdash (di(\mathcal{M}\ a) \wedge_i (di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i (di(\mathcal{M}\ a))$
by *auto*
hence *5*: $\vdash \triangleright(di(\mathcal{M}\ a) \wedge_i (di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b))) \equiv_i \triangleright(di(\mathcal{M}\ a))$
using *FstEqvRule* **by** *blast*
have *6*: $\vdash \triangleright(di(\mathcal{M}\ a)) \equiv_i \triangleright(\mathcal{M}\ a)$
using *FstDiEqvFst* **by** *blast*
have *7*: $\vdash \triangleright(\mathcal{M}\ a) \equiv_i (\mathcal{M}\ a)$
using *MFixFst* **by** *auto*
from *1 2 3 5 6 7* **show** *?thesis* **by** *auto*
qed

lemma *MUptoThruDistrib*:
$\vdash \mathcal{M}(a\ UPTO\ (b\ THRU\ c)) \equiv_i \mathcal{M}((a\ UPTO\ b)\ THRU\ (a\ UPTO\ c))$
proof $-$
have *1*: $\vdash \mathcal{M}((a\ UPTO\ b)\ THRU\ (a\ UPTO\ c)) \equiv_i$
$\triangleright(\ di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ )$
by *simp*
have *2*: $\vdash (\ di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ ) \equiv_i$
$(\ di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ )$
using *DiEqvDiFst* **by** (*metis itl-prop*(*31*) *prop22*)
have *3*: $\vdash (\ di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ ) \equiv_i$
$(\ di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i (di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ c))$
using *DiOrEqv* **by** *auto*
have *4*: $\vdash (\ di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ b)) \wedge_i (di(\mathcal{M}\ a) \vee_i di(\mathcal{M}\ c)) \equiv_i$
$di(\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))$
by *auto*
have *5*: $\vdash (\ di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ ) \equiv_i$
$di(\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))$
using *2 3 4* **by** *auto*
hence *6*: $\vdash \triangleright(\ di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ b))) \wedge_i di(\triangleright((\mathcal{M}\ a) \vee_i (\mathcal{M}\ c)))\ ) \equiv_i$
$\triangleright(\ di(\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))\ )$
using *FstEqvRule* **by** *blast*
have *7*: $\vdash \triangleright(\ di(\mathcal{M}\ a) \vee_i (di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))\ ) \equiv_i$
$\triangleright(\ \triangleright(di(\mathcal{M}\ a)) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))\ )$
using *FstFstOrEqvFstOr* **by** *auto*
have *8*: $\vdash \triangleright(di(\mathcal{M}\ a)) \equiv_i \triangleright((\mathcal{M}\ a))$
using *FstDiEqvFst* **by** *blast*
have *9*: $\vdash \triangleright((\mathcal{M}\ a)) \equiv_i (\mathcal{M}\ a)$
using *MFixFst* **by** *auto*
have *10*: $\vdash \triangleright(di(\mathcal{M}\ a)) \equiv_i (\mathcal{M}\ a)$
using *8 9* **by** *auto*
hence *11*: $\vdash \triangleright(di(\mathcal{M}\ a)) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)) \equiv_i$
$(\mathcal{M}\ a) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))$
by *auto*
hence *12*: $\vdash \triangleright(\triangleright(di(\mathcal{M}\ a)) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))) \equiv_i$
$\triangleright((\mathcal{M}\ a) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c)))$
using *FstEqvRule* **by** *blast*
have *13*: $\vdash\triangleright((\mathcal{M}\ a) \vee_i \triangleright(di(\mathcal{M}\ b) \wedge_i di(\mathcal{M}\ c))) \equiv_i \mathcal{M}(a\ UPTO\ (b\ THRU\ c))$

by *simp*
from *1 6 7 12 13* **show** *?thesis* **by** *auto*
qed

lemma *MThruUptoDistrib*:
$\vdash \mathcal{M}(a\ THRU\ (b\ UPTO\ c)) \equiv_i \mathcal{M}((a\ THRU\ b)\ UPTO\ (a\ THRU\ c))$
proof −
have *1*: $\vdash \mathcal{M}((a\ THRU\ b)\ UPTO\ (a\ THRU\ c)) \equiv_i$
$\triangleright(\triangleright(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i \triangleright(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c)))$ **by** *simp*
have *2*: $\vdash \triangleright(\triangleright(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i \triangleright(di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c))) \equiv_i$
$\triangleright((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c)))$ **using** *FstFstOrEqvFstOr* **by** *auto*
have *3*: $\vdash ((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c))) \equiv_i$
$(di(\mathcal{M}\ a) \wedge_i (di(\mathcal{M}\ b) \vee_i di(\mathcal{M}\ c)))$ **by** *auto*
have *4*: $\vdash (di(\mathcal{M}\ a) \wedge_i (di(\mathcal{M}\ b) \vee_i di(\mathcal{M}\ c))) \equiv_i$
$(di(\mathcal{M}\ a) \wedge_i di(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c)))$ **using** *DiOrEqv* **by** *auto*
have *5*: $\vdash (di(\mathcal{M}\ a) \wedge_i di(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))) \equiv_i$
$(di(\mathcal{M}\ a) \wedge_i di(\triangleright(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))))$ **using** *DiEqvDiFst prop05* **by** *blast*
have *6*: $\vdash ((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c))) \equiv_i$
$(di(\mathcal{M}\ a) \wedge_i di(\triangleright(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))))$ **using** *3 4 5* **by** *auto*
hence *7*: $\vdash \triangleright((di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ b)) \vee_i (di(\mathcal{M}\ a) \wedge_i di(\mathcal{M}\ c))) \equiv_i$
$\triangleright(di(\mathcal{M}\ a) \wedge_i di(\triangleright(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c))))$ **using** *FstEqvRule* **by** *blast*
have *8*: $\vdash \triangleright(di(\mathcal{M}\ a) \wedge_i di(\triangleright(\ (\mathcal{M}\ b) \vee_i (\mathcal{M}\ c)))) \equiv_i$
$\mathcal{M}(a\ THRU\ (b\ UPTO\ c))$ **by** *simp*
from *1 2 7 8* **show** *?thesis* **by** *auto*
qed

lemma *MWithAndDistrib*:
$\vdash \mathcal{M}((a\ AND\ b)\ WITH\ f) \equiv_i \mathcal{M}((a\ WITH\ f)\ AND\ (b\ WITH\ f))$
proof −
have *1*: $\vdash \mathcal{M}((a\ AND\ b)\ WITH\ f) \equiv_i \mathcal{M}(a\ AND\ b) \wedge_i f$
by *simp*
have *2*: $\vdash \mathcal{M}(a\ AND\ b) \equiv_i \mathcal{M}(a\ WITH\ (\mathcal{M}\ b))$
by (*simp add*: *mAND-d-def*)
have *3*: $\vdash \mathcal{M}(a\ AND\ b) \wedge_i f \equiv_i \mathcal{M}(a\ WITH\ (\mathcal{M}\ b)) \wedge_i f$
using *2 prop06* **by** *simp*
have *4*: $\vdash \mathcal{M}(a\ WITH\ (\mathcal{M}\ b)) \wedge_i f \equiv_i \mathcal{M}(a) \wedge_i \mathcal{M}(b) \wedge_i f$
by *simp*
have *5*: $\vdash \mathcal{M}(a) \wedge_i \mathcal{M}(b) \wedge_i f \equiv_i (\mathcal{M}(a) \wedge_i f) \wedge_i (\mathcal{M}(b) \wedge_i f)$
by *auto*
have *6*: $\vdash (\mathcal{M}(a) \wedge_i f) \wedge_i (\mathcal{M}(b) \wedge_i f) \equiv_i \mathcal{M}(a\ WITH\ f) \wedge_i \mathcal{M}(b\ WITH\ f)$
by *simp*
have *7*: $\vdash \mathcal{M}(a\ WITH\ f) \wedge_i \mathcal{M}(b\ WITH\ f) \equiv_i \mathcal{M}((a\ WITH\ f)\ WITH\ (\mathcal{M}(b\ WITH\ f)))$
by *simp*
have *8*: $\vdash \mathcal{M}((a\ WITH\ f)\ WITH\ (\mathcal{M}(b\ WITH\ f))) \equiv_i \mathcal{M}((a\ WITH\ f)\ AND\ (b\ WITH\ f))$
by (*simp add*: *mAND-d-def*)
from *1 2 3 4 5 6 7 8* **show** *?thesis* **by** *simp*
qed

lemma *MHaltWithAndDistrib*:

$\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ AND\ ((HALT\ w)\ WITH\ g)) \equiv_i \mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g))$
proof −
have *1*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ AND\ ((HALT\ w)\ WITH\ g)) \equiv_i$
$\mathcal{M}(((HALT\ w)\ WITH\ f)\ WITH\ (\mathcal{M}((HALT\ w)\ WITH\ g)))$
by (*simp add*: *mAND-d-def*)
have *2*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ WITH\ (\mathcal{M}((HALT\ w)\ WITH\ g))) \equiv_i$
$\mathcal{M}(HALT\ w) \wedge_i f \wedge_i \mathcal{M}(HALT\ w) \wedge_i g$
by (*simp add*: *mHALT-d-def*)
have *3*: $\vdash \mathcal{M}(HALT\ w) \wedge_i f \wedge_i \mathcal{M}(HALT\ w) \wedge_i g \equiv_i \mathcal{M}(HALT\ w) \wedge_i f \wedge_i g$
by *auto*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *MHaltWithUptoHaltWithEqvHaltWithOr*:
$\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ UPTO\ ((HALT\ w)\ WITH\ g)) \equiv_i \mathcal{M}((HALT\ w)\ WITH\ (f \vee_i g))$
proof −
have *1*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ UPTO\ ((HALT\ w)\ WITH\ g)) \equiv_i$
$\triangleright(\mathcal{M}((HALT\ w)\ WITH\ f) \vee_i \mathcal{M}((HALT\ w)\ WITH\ g))$
by *simp*
have *2*: $\vdash \triangleright(\mathcal{M}((HALT\ w)\ WITH\ f) \vee_i \mathcal{M}((HALT\ w)\ WITH\ g)) \equiv_i$
$\triangleright((\mathcal{M}(HALT\ w) \wedge_i f) \vee_i (\mathcal{M}(HALT\ w) \wedge_i g))$
by *simp*
have *3*: $\vdash (\mathcal{M}(HALT\ w) \wedge_i f) \vee_i (\mathcal{M}(HALT\ w) \wedge_i g) \equiv_i (\mathcal{M}(HALT\ w) \wedge_i (f \vee_i g))$
by *auto*
have *4*: $\vdash \triangleright((\mathcal{M}(HALT\ w) \wedge_i f) \vee_i (\mathcal{M}(HALT\ w) \wedge_i g)) \equiv_i \triangleright(\mathcal{M}(HALT\ w) \wedge_i (f \vee_i g))$
using *3 FstEqvRule* **by** *blast*
have *5*: $\vdash \triangleright(\mathcal{M}(HALT\ w) \wedge_i (f \vee_i g)) \equiv_i \triangleright(\mathcal{M}((HALT\ w)\ WITH\ (f \vee_i g)))$
by *simp*
have *6*: $\vdash (\mathcal{M}((HALT\ w)\ WITH\ (f \vee_i g))) \equiv_i \triangleright(\mathcal{M}((HALT\ w)\ WITH\ (f \vee_i g)))$
using *MFixFst* **by** *blast*
from *1 2 3 4 5 6* **show** *?thesis* **by** *simp*
qed

lemma *DiHaltAndDiHaltAndEqvDiHaltAndAnd*:
$\vdash di(halt\ (init\ w) \wedge_i f) \wedge_i di(halt\ (init\ w) \wedge_i g) \equiv_i di(halt\ (init\ w) \wedge_i f \wedge_i g)$
proof −
have *1*: $\vdash di(halt\ (init\ w) \wedge_i f) \wedge_i di(halt\ (init\ w) \wedge_i g) \equiv_i$
$di(\triangleright(fin\ (init\ w)) \wedge_i f) \wedge_i di\ (\triangleright(fin\ (init\ w)) \wedge_i g)$
using *HaltStateEqvFstFinState* **by** *auto*
have *2*: $\vdash di(\triangleright(fin\ (init\ w)) \wedge_i f) \wedge_i di(\triangleright(fin\ (init\ w)) \wedge_i g) \equiv_i$
$di(\triangleright(fin\ (init\ w)) \wedge_i f \wedge_i g)$
using *LFstAndDistrD* **by** *simp*
have *3*: $\vdash di(\triangleright(fin\ (init\ w)) \wedge_i f \wedge_i g) \ \equiv_i di(halt\ (init\ w) \wedge_i f \wedge_i g)$
using *HaltStateEqvFstFinState* **by** *auto*
from *1 2 3* **show** *?thesis* **by** *simp*
qed

lemma *MHaltWithThruHaltWithEqvHaltWithAndHaltWith*:
$\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ THRU\ ((HALT\ w)\ WITH\ g)) \equiv_i \mathcal{M}(((HALT\ w)\ WITH\ f)\ AND\ ((HALT\ w)\ WITH\ g))$

proof −
have *1*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ THRU\ ((HALT\ w)\ WITH\ g)) \equiv_i$
$\triangleright(\ di(\mathcal{M}(HALT\ w) \wedge_i f) \wedge_i di(\mathcal{M}(HALT\ w) \wedge_i g)\ )$
by *simp*
have *2*: $\vdash di(\mathcal{M}(HALT\ w) \wedge_i f) \wedge_i di(\mathcal{M}(HALT\ w) \wedge_i g) \equiv_i$
$di(halt(init\ w) \wedge_i f) \wedge_i di(halt(init\ w) \wedge_i g)$
using *MHaltAlt DiEqvDi* **by** *auto*
have *3*: $\vdash di(halt(init\ w) \wedge_i f) \wedge_i di(halt(init\ w) \wedge_i g) \equiv_i$
$di(halt(init\ w) \wedge_i f \wedge_i g)$
using *DiHaltAndDiHaltAndEqvDiHaltAndAnd* **by** *simp*
have *4*: $\vdash di(\ halt(init\ w) \wedge_i f \wedge_i g) \equiv_i di(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g)$
using *MHaltAlt* **by** *auto*
have *5*: $\vdash di(\mathcal{M}(HALT\ w) \wedge_i f) \wedge_i di(\mathcal{M}(HALT\ w) \wedge_i g) \equiv_i di(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g)$
using *2 3 4* **by** *simp*
have *6*: $\vdash \triangleright(di(\mathcal{M}(HALT\ w) \wedge_i f) \wedge_i di(\mathcal{M}(HALT\ w) \wedge_i g)) \equiv_i \triangleright(di(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g))$
using *5 FstEqvRule* **by** *blast*
have *7*: $\vdash \triangleright(di(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g)) \equiv_i \triangleright(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g)$
using *FstDiEqvFst* **by** *simp*
have *8*: $\vdash \triangleright(\mathcal{M}(HALT\ w) \wedge_i f \wedge_i g) \equiv_i \triangleright(\mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g)))$
by *simp*
have *9*: $\vdash \mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g)) \equiv_i \triangleright(\mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g)))$
using *MFixFst* **by** *blast*
have *10*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ THRU\ ((HALT\ w)\ WITH\ g)) \equiv_i \mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g))$
using *1 2 3 4 5 6 7 8 9* **by** *simp*
have *11*: $\vdash \mathcal{M}(((HALT\ w)\ WITH\ f)\ AND\ ((HALT\ w)\ WITH\ g)) \equiv_i \mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g))$
using *MHaltWithAndDistrib* **by** *simp*
have *12*: $\vdash \mathcal{M}((HALT\ w)\ WITH\ (f \wedge_i g)) \equiv_i \mathcal{M}(((HALT\ w)\ WITH\ f)\ AND\ ((HALT\ w)\ WITH\ g))$
using *11* **by** *simp*
from *10 12* **show** *?thesis* **by** *simp*
qed

end

theory *ITA*
imports *ITL*

begin
sledgehammer-params [*minimize=true*,*preplay-timeout=10*,*timeout=60*,*verbose=true*,
provers=cvc4 z3 e spass vampire]

8 Interval Temporal Algebra

8.1 Definition of fuse operator

The *fuse* operation corresponds to the chop operation of ITL at semantic level. Although *fuse* is not needed to define the semantics of the ITL fusion/chop operation, it is introduced to link with the work of [1]. The ITL proof system is derived from a collection of algebraic laws. This work is a continuation of

[2].

primrec *fuse* :: $'a$ *interval* $\Rightarrow$ $'a$ *interval* $\Rightarrow$ $'a$ *interval* **where**
  fuse-St : *fuse* $(St\ x)\ ys = ys$
| *fuse-Cons* : *fuse* $(x \odot xs)\ ys = x \odot (fuse\ xs\ ys)$

8.1.1 Fuse lemmas

lemma *interval-fuse-leftneutral* :
  $fuse\ (St\ (intfirst\ xs))\ xs = xs$
by *simp*

lemma *interval-fuse-rightneutral* :
  $fuse\ xs\ (St\ (intlast\ xs)) = xs$
by (*induct xs*) *simp-all*

lemma *interval-intfirst-fuse* :
 assumes $intlast\ xs = intfirst\ ys$
 shows $intfirst\ (fuse\ xs\ ys) = intfirst\ xs$
using *assms* **by** (*induct xs*) *simp-all*

lemma *interval-intlast-fuse* :
 assumes $intlast\ xs = intfirst\ ys$
 shows $intlast\ (fuse\ xs\ ys) = intlast\ ys$
using *assms* **by** (*induct xs*) *simp-all*

lemma *interval-FusionAssoc* :
 assumes $(intlast\ xs) = (intfirst\ ys) \wedge (intlast\ ys) = (intfirst\ zs)$
 shows $(fuse\ xs\ (fuse\ ys\ zs)) = (fuse\ (fuse\ xs\ ys)\ zs)$
using *assms* **by** (*induct xs*) *simp-all*

lemma *interval-fuse-intlen* :
 assumes $intlast\ xs = intfirst\ ys$
 shows $intlen\ (fuse\ xs\ ys) = (intlen\ xs) + (intlen\ ys)$
using *assms* **by** (*induct xs*) *simp-all*

lemma *interval-intlast-intfirst*:
  $(intlast\ (prefix\ i\ xs)) = (intfirst\ (suffix\ i\ xs))$
by (*induct xs arbitrary*: *i*, *simp*, *simp add*: *Nitpick.case-nat-unfold*)

lemma *interval-fuse-pref-suf* :
  $(\ fuse\ (prefix\ i\ xs)\ (suffix\ i\ xs)) = xs$
by (*induct xs arbitrary*: *i*, *simp*, *simp add*: *Nitpick.case-nat-unfold*)

lemma *interval-prefix-fuse* :
 assumes $intlast\ xs = intfirst\ ys$
 shows $(prefix\ (intlen\ xs)\ (fuse\ xs\ ys)) = xs$
using *assms* **by** (*induct xs arbitrary*: *ys*, *simp*, *simp*)

lemma *interval-suffix-fuse* :
 assumes $intlast\ xs = intfirst\ ys$

shows $(suffix\ (intlen\ xs)\ (fuse\ xs\ ys)) = ys$
using *assms* **by** (*induct xs arbitrary*: *ys*, *simp*, *simp*)

lemma *chop-fuse-1* :
$(\exists\ \sigma 1\ \sigma 2.\ \sigma = fuse\ \sigma 1\ \sigma 2\ \wedge$
$(\sigma 1\ \models f) \wedge (\sigma 2\ \models g) \wedge$
$(intlast\ \sigma 1 = intfirst\ \sigma 2)) \longleftrightarrow$
$(\exists\ i.\ 0 \leq i \wedge i \leq intlen\ \sigma \wedge (\ prefix\ i\ \sigma\ \models f) \wedge (suffix\ i\ \sigma\ \models g))$
by (*metis interval-fuse-intlen interval-fuse-pref-suf interval-intlast-intfirst interval-intlen-gr-zero interval-prefix-fuse interval-suffix-fuse le-add-same-cancel1*)

lemma *chop-fuse-2* :
$(\exists\ \sigma 1\ \sigma 2.\ \sigma = fuse\ \sigma 1\ \sigma 2\ \wedge$
$(\sigma 1 \in X) \wedge (\sigma 2 \in Y) \wedge$
$(intlast\ \sigma 1 = intfirst\ \sigma 2)) \longleftrightarrow$
$(\exists\ i \leq intlen\ \sigma.\ (prefix\ i\ \sigma) \in X \wedge (suffix\ i\ \sigma) \in Y)$
by (*metis interval-fuse-intlen interval-fuse-pref-suf interval-intlast-intfirst interval-prefix-fuse interval-suffix-fuse le-add1*)

lemma *chop-fuse*:
$(\exists\ \sigma 1\ \sigma 2.\ \sigma = fuse\ \sigma 1\ \sigma 2\ \wedge$
$(\sigma 1\ \models f) \wedge (\sigma 2\ \models g) \wedge$
$(intlast\ \sigma 1 = intfirst\ \sigma 2)) \longleftrightarrow$
$(\sigma\ \models f;g)$
using *chop-fuse-1* **by** (*simp add*: *chop-fuse-1*)

8.2 Definition of Set of intervals and Operations on them

type-synonym *'a intervals* = *'a interval set*

definition *lan*:: *'a pitl* ⇒ *'a intervals*
where $lan\ f = \{\ \sigma\ .\ (\sigma \models f)\ \}$

definition *fusion* :: *'a intervals* ⇒ *'a intervals* ⇒ *'a intervals* (**infixl** · *70*)
where $X \cdot Y = \{fuse\ \sigma 1\ \sigma 2 |\ \sigma 1\ \sigma 2.\ \sigma 1 \in X \wedge \sigma 2 \in Y \wedge intlast\ \sigma 1 = intfirst\ \sigma 2\}$

definition *reverse* :: *'a intervals* ⇒ *'a intervals* ((*SRev* -) [*85*] *85*)
where $(SRev\ X) = \{intrev\ \sigma\ |\sigma.\ \sigma \in X\ \}$

definition *sempty* :: *'a intervals* (*SEmpty*)
where
SEmpty ≡ *range St*

definition *smore* :: *'a intervals* (*SMore*)
where
SMore ≡ − *SEmpty*

definition *sskip* :: *'a intervals* (*SSkip*)
where
SSkip ≡ −(*SEmpty* ∪ (*SMore*·*SMore*))

definition *sfalse* :: *'a intervals* (*SFalse*)
where
 SFalse $\equiv$ {}

definition *strue* :: *'a intervals* (*STrue*)
where
 STrue $\equiv -\{\}$

definition *sinit* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SInit* -) [*85*] *85*)
where
 SInit X $\equiv$ (*X* $\cap$ *SEmpty*)$\cdot$*STrue*

definition *ssometime* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SSometime* -) [*85*] *85*)
where
 SSometime X $\equiv$ *STrue*$\cdot$*X*

definition *salways* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SAlways* -) [*85*] *85*)
where
 SAlways X $\equiv -$(*SSometime* ($-X$))

definition *sdi* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SDi* -) [*85*] *85*)
where
 SDi X $\equiv$ *X*$\cdot$*STrue*

definition *sbi* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SBi* -) [*85*] *85*)
where
 SBi X $\equiv -$(*SDi* ($-X$))

definition *sda* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SDa* -) [*85*] *85*)
where
 SDa X $\equiv$ *STrue*$\cdot$*X*$\cdot$*STrue*

definition *sba* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SBa* -) [*85*] *85*)
where
 SBa X $\equiv -$(*SDa* ($-X$))

definition *snext* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SNext* -) [*85*] *85*)
where
 SNext X $\equiv$ *SSkip*$\cdot$*X*

definition *swnext* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SWnext* -) [*85*] *85*)
where
 SWnext X $\equiv$ ($-$(*SSkip*$\cdot$($-X$)))

definition *sprev* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SPrev* -) [*85*] *85*)
where
 SPrev X $\equiv$ *X*$\cdot$*SSkip*

definition *swprev* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SWprev* -) [*85*] *85*)

where
  SWprev X $\equiv (-((-X) \cdot SSkip))$

primrec *spower* :: *'a intervals* $\Rightarrow$ *nat* $\Rightarrow$ *'a intervals* ((*SPower* - -) [*88*,*88*] *87*)
where
  pwr-0 : *SPower X 0* = *SEmpty*
| *pwr-Suc*: *SPower X* (*Suc n*) = ((*X* $\cap$ *SMore*)$\cdot$(*SPower X n*))

definition *sstar* :: *'a intervals* $\Rightarrow$ *'a intervals* ((*SStar* -) [*85*] *85*)
where
  SStar X $\equiv$ ($\bigcup$ *n*. *SPower X n*)

8.3 Simplification Lemmas

lemma *snot-elim* :
 $x \in -X \longleftrightarrow x \notin X$
by *simp*

lemma *sor-elim* :
 $x \in (X \cup Y) \longleftrightarrow (x \in X \vee x \in Y)$
by *simp*

lemma *sand-elim* :
  $x \in (X \cap Y) \longleftrightarrow (x \in X \wedge x \in Y)$
by *simp*

lemma *sfalse-elim* :
  $\sigma \notin SFalse$
by (*simp add*: *sfalse-def*)

lemma *strue-elim* :
  $\sigma \in STrue$
by (*simp add*: *strue-def*)

lemma *sempty-elim* :
  $\sigma \in SEmpty \longleftrightarrow intlen\ \sigma = 0$
by (*simp add*: *image-iff interval-st-intlen sempty-def*)

lemma *smore-elim* :
  $\sigma \in SMore \longleftrightarrow intlen\ \sigma > 0$
by (*simp add*: *sempty-elim smore-def*)

lemma *fusion-iff*:
  $\sigma \in X \cdot Y \longleftrightarrow (\exists \sigma 1\ \sigma 2.\ \sigma = fuse\ \sigma 1\ \sigma 2 \wedge \sigma 1 \in X \wedge \sigma 2 \in Y \wedge intlast\ \sigma 1 = intfirst\ \sigma 2)$
by (*unfold fusion-def*) *auto*

lemma *fusion-iff-1*:
  $\sigma \in X \cdot Y \longleftrightarrow (\exists\ i \leq intlen\ \sigma.\ (prefix\ i\ \sigma) \in X \wedge (suffix\ i\ \sigma) \in Y\ )$
by (*simp add*: *chop-fuse-2 fusion-iff*)

lemma *smore-fusion-smore* :
$\sigma \in (SMore \cdot SMore) \longleftrightarrow intlen\ \sigma > 1$
using *fusion-iff-1*
by (*metis interval-prefix-length-good interval-suffix-length-good less-one not-less not-less-iff-gr-or-eq smore-elim zero-less-diff*)

lemma *sskip-elim* :
$\sigma \in SSkip \longleftrightarrow intlen\ \sigma = 1$
using *sskip-def smore-fusion-smore*
by (*metis One-nat-def Suc-lessI Un-iff less-numeral-extra*(*4*) *sempty-elim smore-def smore-elim snot-elim zero-neq-one*)

lemma *spower-elim-zero* :
$\sigma \in SPower\ X\ 0 \longleftrightarrow \sigma \in SEmpty$
by *simp*

lemma *spower-elim-suc* :
$\sigma \in SPower\ X\ (Suc\ n) \longleftrightarrow \sigma \in (X \cap SMore) \cdot (SPower\ X\ n)$
by *simp*

lemma *spower-elim-suc-1* :
$\sigma \in (X \cap SMore) \cdot (SPower\ X\ n) \longleftrightarrow$
$(\exists \sigma 1\ \sigma 2.\ \sigma = fuse\ \sigma 1\ \sigma 2 \wedge \sigma 1 \in X \wedge intlen\ \sigma 1 > 0 \wedge \sigma 2 \in (SPower\ X\ n) \wedge$
$intlast\ \sigma 1 = intfirst\ \sigma 2)$
by (*meson IntD1 IntD2 IntI smore-elim fusion-iff*)

lemma *sstar-elim* :
$\sigma \in SStar\ X \longleftrightarrow (\exists\ n.\ \sigma \in SPower\ X\ n)$
by (*simp add*: *sstar-def*)

lemma *sstar-elim-1* :
$(\exists\ n\ .\ \sigma \in SPower\ X\ n) \longleftrightarrow$
$(\ \sigma \in SPower\ X\ 0 \vee (\exists\ n.\ \sigma \in SPower\ X\ (Suc\ n)))$
by (*metis not0-implies-Suc*)

lemma *spower-suc* :
$(\exists\ n.\ \sigma \in SPower\ X\ (Suc\ n)) \longleftrightarrow$
$(\exists\ n.\ \sigma \in (X \cap SMore) \cdot (SPower\ X\ n))$
by *simp*

lemma *spower-suc-1* :
$(\exists\ n.\ \sigma \in (X \cap SMore) \cdot (SPower\ X\ n)) \longleftrightarrow$
$\sigma \in (X \cap SMore) \cdot (SStar\ X)$
by (*metis fusion-iff sstar-elim*)

lemma *sstar-eqv* :
$\sigma \in SStar\ X \longleftrightarrow$
$(\ \sigma \in SEmpty \vee \sigma \in (X \cap SMore) \cdot (SStar\ X))$
by (*metis spower.simps*(*1*) *spower-elim-suc spower-suc-1 sstar-elim sstar-elim-1*)

lemma *spower-sskip-elim* :
$(\sigma \in SPower\ SSkip\ n) \longleftrightarrow intlen\ \sigma = n$
by (*induct n arbitrary*: σ, *simp add*: *sempty-elim*, *smt chop-fuse diff-Suc-1 interval-fuse-intlen more-d-def more-defs next-d-def plus-1-eq-Suc skip-defs spower-elim-suc spower-elim-suc-1 sskip-elim zero-less-Suc zero-less-one*)

lemma *srev-elim*:
$\sigma \in (SRev\ X) \longleftrightarrow intrev\ \sigma \in X$
by (*smt interval-rev-rev-ident mem-Collect-eq reverse-def*)

8.4 Algebraic Laws

8.4.1 Commutative Additive Monoid

lemma *UnionCommute*:
$(X{::}'a\ intervals) \cup Y = Y \cup X$
by (*simp add*: *Un-commute*)

lemma *UnionSFalse*:
$X \cup SFalse = X$
by (*simp add*: *sfalse-def*)

lemma *UnionAssoc*:
$(X{::}'a\ intervals) \cup (Y \cup Z) = (X \cup Y) \cup Z$
by (*simp add*: *sup-assoc*)

8.4.2 Boolean algebra

lemma *Huntington*:
$(X{::}'a\ intervals) = -(-X \cup -Y) \cup -(-X \cup Y)$
by *auto*

lemma *Morgan*:
$(X{::}'a\ intervals) \cap Y = -(-X \cup -Y)$
by *auto*

— identities

lemma *STrueTop*:
$STrue = X \cup -X$
by (*simp add*: *strue-def*)

lemma *SFalseBottom*:
$SFalse = X \cap -X$
by (*simp add*: *sfalse-def*)

8.4.3 multiplicative monoid

lemma *FusionSEmptyL* [*simp*]:
$SEmpty \cdot X = X$
using *fusion-iff-1 set-eqI*[*of SEmpty·X X*]

by (*metis interval-intlen-gr-zero interval-prefix-length-good interval-suffix-zero sempty-elim*)

lemma *FusionSEmptyR* [*simp*]:
$X \cdot SEmpty\ = X$
using *fusion-iff-1 set-eqI*[*of* $X{\cdot}SEmpty$ X]
by (*metis add-cancel-right-right fusion-iff interval-fuse-intlen interval-fuse-pref-suf interval-intlast-intfirst interval-prefix-fuse interval-prefix-intlen sempty-elim*)

lemma *FusionAssoc* [*simp*]:
$X{\cdot}(Y{\cdot}Z) = (X{\cdot}Y){\cdot}Z$
using *set-eqI*[*of* $X{\cdot}(Y{\cdot}Z)$ $(X{\cdot}Y){\cdot}Z$]
by (*smt fusion-iff interval-intfirst-fuse interval-FusionAssoc interval-intlast-fuse*)

— left and right distributivity

lemma *FusionUnionDistL*:
$(X \cup Y){\cdot}Z = X{\cdot}Z \cup Y{\cdot}Z$
using *fusion-iff set-eqI*[*of* $(X \cup Y){\cdot}Z$ $X{\cdot}Z \cup Y{\cdot}Z$]
by (*metis* (*no-types*, *lifting*) *sor-elim*)

lemma *FusionUnionDistR*:
$X{\cdot}(Y \cup Z) = X{\cdot}Y \cup X{\cdot}Z$
using *fusion-iff set-eqI*[*of* $X{\cdot}(Y \cup Z)$ $X{\cdot}Y \cup X{\cdot}Z$]
by (*metis* (*no-types*, *lifting*) *sor-elim*)

— left and right annihilation
lemma *SFalseFusion*:
$SFalse \cdot X = SFalse$
by (*simp add*: *fusion-def sfalse-def*)

lemma *FusionSFalse*:
$X{\cdot}SFalse = SFalse$
by (*simp add*: *fusion-def sfalse-def*)

— idempotency
lemma *UnionIdem*:
$(X{::}'a\ intervals) \cup X = X$
by *simp*

8.4.4 Subsumption order

lemma *Subsumption*:
$((X :: 'a\ intervals) \subseteq Y) = (X \cup Y = Y)$
by *auto*

8.4.5 Helper lemmas

lemma *FusionRuleR*:
assumes $X \subseteq Y$
shows $Z \cdot X \subseteq Z \cdot Y$
using *assms FusionUnionDistR* **by** (*metis Subsumption*)

lemma *FusionRuleL*:
 assumes $X \subseteq Y$
 shows $X \cdot Z \subseteq Y \cdot Z$
using *assms* **by** (*metis FusionUnionDistL subset-Un-eq*)

lemma *spower-commutes*:
 $(X \cap SMore)\cdot(SPower\ X\ n) = (SPower\ X\ n)\cdot(X \cap SMore)$
by (*induct n*, *simp*, *simp*)

lemma *fusion-inductl*:
 assumes $Y \cup X\cdot Z \subseteq Z$
 shows $(SPower\ X\ n)\cdot Y \subseteq Z$
using *assms*
by (*induct n*, *auto*, *smt IntD1 UnI2 FusionAssoc fusion-iff pwr-Suc subset-eq*)

lemma *fusion-inductr*:
assumes $Y \cup Z\cdot X \subseteq Z$
 shows $Y\cdot(SPower\ X\ n) \subseteq Z$
using *assms*
by (*induct n*, *simp*, *smt abel-semigroup.commute FusionAssoc FusionUnionDistL FusionUnionDistR le-iff-sup pwr-Suc spower-commutes sup.abel-semigroup-axioms sup-assoc sup-inf-absorb*)

lemma *sstar-contl*:
 $Y \cdot (SStar\ X) = (\bigcup n.\ Y \cdot (SPower\ X\ n))$
using *set-eqI*[*of* $Y \cdot (SStar\ X)\ (\bigcup n.\ Y \cdot (SPower\ X\ n))$]
by (*smt UN-iff fusion-iff sstar-def*)

lemma *sstar-contr*:
 $(SStar\ X)\cdot Y = (\bigcup n.\ (SPower\ X\ n)\cdot Y)$
using *set-eqI*[*of* $(SStar\ X)\cdot Y\ (\bigcup n.\ (SPower\ X\ n)\cdot Y)$]
by (*smt UN-iff fusion-iff sstar-def*)

8.4.6 Kleene Algebra

— left unfold
lemma *UnfoldL*:
 $SEmpty \cup X\cdot(SStar\ X) = (SStar\ X)$
proof −
 have *1*: $(SStar\ X) = SEmpty \cup (X \cap SMore)\cdot(SStar\ X)$
 by (*meson Un-iff set-eqI sstar-eqv*)
 have *2*: $(X \cap SMore)\cdot(SStar\ X) \subseteq X\cdot(SStar\ X)$
 by (*simp add*: *FusionRuleL*)
 have *3*: $(SStar\ X) \subseteq SEmpty \cup X\cdot(SStar\ X)$
 using *1 2* **by** *blast*
 have *4*: $SEmpty \subseteq (SStar\ X)$
 using *1* **by** *auto*
 have *5*: $X \subseteq SEmpty \cup (X \cap SMore)$
 by (*simp add*: *Un-Int-distrib smore-def*)
 have *6*: $X\cdot(SStar\ X) \subseteq (SStar\ X) \cup (X \cap SMore)\cdot(SStar\ X)$

using *5* **by** (*metis FusionRuleL FusionUnionDistL FusionSEmptyL*)
have *7*: (*SStar X*) $\subseteq$ *SEmpty* $\cup$ (*X* $\cap$ *SMore*)·(*SStar X*)
using *1* **by** *auto*
have *8*: *X*·(*SStar X*) $\subseteq$ *SEmpty* $\cup$ (*X* $\cap$ *SMore*)·(*SStar X*)
using *6 7* **by** *blast*
hence *9*: *X*·(*SStar X*) $\subseteq$ (*SStar X*)
using *1* **by** *auto*
have *10*: *SEmpty* $\cup$ *X*·(*SStar X*) $\subseteq$ (*SStar X*)
using *9 4* **by** *simp*
from *3 10* **show** *?thesis* **by** *auto*
qed

— Left induction
lemma *SStarInductL*:
assumes *Y* $\cup$ *X*·*Z* $\subseteq$ *Z*
shows (*SStar X*)·*Y* $\subseteq$ *Z*
by (*metis UN-least assms fusion-inductl sstar-contr*)

— Right induction
lemma *SStarInductR*:
assumes *Y* $\cup$ *Z*·*X* $\subseteq$ *Z*
shows *Y*·(*SStar X*) $\subseteq$ *Z*
using *sstar-contl assms fusion-inductr* **by** *blast*

8.4.7 Time reversal

lemma *SRevSEmpty*:
(*SRev SEmpty*) = *SEmpty*
using *set-eqI*[*of* (*SRev SEmpty*) *SEmpty*]
by (*simp add*: *sempty-elim srev-elim*)

lemma *SRevSNot*:
(*SRev* (− *X*)) = (− (*SRev X*))
using *set-eqI*[*of* (*SRev* (− *X*)) (− (*SRev X*))]
by (*simp add*: *srev-elim*)

lemma *SRevFusion*:
(*SRev* (*X*·*Y*)) = (*SRev Y*)·(*SRev X*)
using *set-eqI*[*of* (*SRev* (*X*·*Y*)) (*SRev Y*)·(*SRev X*)]
using *fusion-iff-1*
by (*smt interval-intrev-intlen interval-intrev-prefix interval-intrev-suffix*
interval-prefix-length interval-suffix-length-good order-refl srev-elim)

lemma *SRevUnion*:
(*SRev* (*X* $\cup$ *Y*)) = (*SRev X*) $\cup$ (*SRev Y*)
using *set-eqI*[*of* (*SRev* (*X* $\cup$ *Y*)) (*SRev X*) $\cup$ (*SRev Y*)]
using *srev-elim* **by** *auto*

lemma *SRevSPower*:

(*SRev* (*SPower X n*)) = (*SPower* (*SRev X*) *n*)
proof
(*induct n*)
case *0*
then show *?case* **by** (*simp add*: *SRevSEmpty*)
next
case (*Suc n*)
then show *?case*
by (*smt Morgan SRevFusion SRevSEmpty SRevSNot SRevUnion pwr-Suc smore-def spower-commutes*)
qed

lemma *SRevSStar*:
(*SRev* (*SStar X*)) = (*SStar* (*SRev X*))
proof −
have *1*: (*SRev* (*SStar X*)) = (*SRev* ($\bigcup$ *n*. *SPower X n*)) **by** (*simp add*: *sstar-def*)
have *2*: (*SRev* ($\bigcup$ *n*. *SPower X n*)) = ($\bigcup$ *n*. *SPower* (*SRev X*) *n*)
using *set-eqI*[*of* (*SRev* ($\bigcup$ *n*. *SPower X n*)) ($\bigcup$ *n*. *SPower* (*SRev X*) *n*)]
by (*metis* (*mono-tags*, *lifting*) *SRevSPower UN-iff srev-elim*)
have *3*: ($\bigcup$ *n*. *SPower* (*SRev X*) *n*) = (*SStar* (*SRev X*)) **by** (*simp add*: *sstar-def*)
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *SRevSRev*:
(*SRev* (*SRev X*)) = *X*
using *set-eqI*[*of* (*SRev* (*SRev X*)) *X*]
by (*simp add*: *srev-elim*)

8.4.8 ITL specific Laws

lemma *PwrFusionInterL*:
((((*SPower SSkip n*) ∩ *X*)·*V*) ∩ (((*SPower SSkip n*) ∩ *Y*)·*W*)) =
(((*SPower SSkip n*) ∩ *X* ∩ *Y*)·(*V* ∩ *W*))
using *set-eqI*[*of* ((((*SPower SSkip n*) ∩ *X*)·*V*) ∩ (((*SPower SSkip n*) ∩ *Y*)·*W*))
(((*SPower SSkip n*) ∩ *X* ∩ *Y*)·(*V* ∩ *W*))]
using *fusion-iff*
by (*smt interval-prefix-fuse interval-suffix-fuse sand-elim spower-sskip-elim*)

lemma *PwrFusionInterR*:
((*V*·((*SPower SSkip n*) ∩ *X*)) ∩ ((*W*·((*SPower SSkip n*) ∩ *Y*)))) =
((*V* ∩ *W*)·((*SPower SSkip n*) ∩ *X* ∩ *Y*))
using *set-eqI*[*of* ((*V*·((*SPower SSkip n*) ∩ *X*)) ∩ ((*W*·((*SPower SSkip n*) ∩ *Y*))))
((*V* ∩ *W*)·((*SPower SSkip n*) ∩ *X* ∩ *Y*))]
using *fusion-iff*
by (*smt add-right-imp-eq interval-fuse-intlen interval-prefix-fuse*
interval-suffix-fuse sand-elim spower-sskip-elim)

lemma *SSkipFusionImpSMore*:
SSkip·*STrue* ⊆ *SMore*
by (*metis fusion-iff gr0I interval-fuse-intlen nat*.*distinct*(*1*) *plus-1-eq-Suc*
smore-elim sskip-elim subsetI)

lemma *SStarSkip*:
$(SStar\ SSkip) = STrue$
using *set-eqI*[*of* (*SStar SSkip*) *STrue*]
by (*simp add*: *strue-def spower-sskip-elim sstar-elim*)

8.5 Derived Laws

8.5.1 Helper Lemmas

lemma *B01*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y$
shows $-Y \subseteq -X$
using *assms* **by** *auto*

lemma *B04*:
$((X{::}\ 'a\ intervals) = Y\ ) \longleftrightarrow (X \subseteq Y) \wedge (Y \subseteq X)$
by *auto*

lemma *B09*:
assumes $-X \cup Y = STrue$
shows $(X{::}\ 'a\ intervals) \subseteq Y$
using *assms* **using** *strue-def* **by** *auto*

lemma *B20*:
$(X{::}\ 'a\ intervals) \subseteq Y \cup Z \longleftrightarrow X \cap -Y \subseteq Z$
by *auto*

lemma *B28*:
$((X{::}\ 'a\ intervals) \cap Y) \cup (X \cap Z) = X \cap (Y \cup Z)$
by *auto*

lemma *CH01*:
$STrue{\cdot}STrue = STrue$
by (*metis FusionSEmptyR FusionUnionDistR Int-commute SStarSkip STrueTop UnfoldL inf-sup-absorb*)

lemma *CH07*:
$(((SSkip \cap X){\cdot}V) \cap ((SSkip \cap Y){\cdot}W)) = ((SSkip \cap X \cap Y){\cdot}(V \cap W))$
using *PwrFusionInterL*[*of 1 X V Y W*]
by (*simp add*: *inf-commute smore-def sskip-def*)

lemma *CH08*:
$((V{\cdot}(SSkip \cap X)) \cap ((W{\cdot}(SSkip \cap Y)))) = ((V \cap W){\cdot}(SSkip \cap X \cap Y))$
using *PwrFusionInterR*[*of V 1 X W Y*]
by (*simp add*: *inf-commute smore-def sskip-def*)

lemma *CH09*:
$(((X \cap SEmpty){\cdot}V) \cap ((Y \cap SEmpty){\cdot}W)) = (((X \cap Y) \cap SEmpty){\cdot}(V \cap W))$
using *PwrFusionInterL*[*of 0 X V Y W*]
by (*metis* (*no-types*, *lifting*) *inf-assoc inf-commute pwr-0*)

lemma *CH10*:
$((V \cdot (X \cap SEmpty)) \cap ((W \cdot (Y \cap SEmpty)))) = ((V \cap W) \cdot ((X \cap Y) \cap SEmpty))$
using *PwrFusionInterR*[*of V 0 X W Y*]
by (*metis* (*no-types*, *lifting*) *inf-assoc inf-commute pwr-0*)

lemma *ST13*:
$((X \cap SEmpty) \cdot Z) \cap ((Y \cap SEmpty) \cdot Z) = ((X \cap Y) \cap SEmpty) \cdot Z$
by (*simp add*: *CH09*)

lemma *ST15*:
$(SStar\ (X \cap SEmpty)) = SEmpty$
by (*metis FusionSEmptyL inf .right-idem inf-le2 UnfoldL*
SStarInductR sup.orderE sup-inf-absorb)

lemma *ST21*:
$((-X) \cap SEmpty) \cup (X \cap SEmpty) = SEmpty$
by *blast*

lemma *ST24*:
$(SInit\ X) \cap (SInit\ Y) = (SInit\ (X \cap Y))$
by (*simp add*: *ST13 sinit-def*)

lemma *ST25*:
$(SInit\ STrue) = STrue$
by (*simp add*: *sinit-def strue-def*)

lemma *ST26*:
$(SInit\ (-X)) \cup (SInit\ X) = STrue$
by (*metis Compl-disjoint2 ST21 ST25 Un-Int-distrib compl-bot-eq FusionUnionDistL*
sinit-def strue-def sup-bot.right-neutral sup-top-right)

lemma *ST28*:
$(SDi\ (SInit\ X)) = (SInit\ X)$
by (*metis compl-bot-eq FusionAssoc FusionUnionDistR FusionSEmptyR sdi-def*
sinit-def strue-def sup-top-right UnionCommute)

lemma *ST33*:
$(STrue \cap SEmpty) \cdot SEmpty = SEmpty$
by (*simp add*: *strue-def*)

lemma *ST36*:
$(SInit\ (-X)) \subseteq -(SInit\ X)$
by (*metis Compl-disjoint ST24 compl-bot-eq disjoint-eq-subset-Compl double-complement*
inf .coboundedI2 inf .orderE sfalse-def SFalseFusion sinit-def strue-def)

lemma *ST37*:
$-(SInit\ X) \subseteq (SInit\ (-X))$
using *B09 ST26* **by** *auto*

lemma *ST38*:

$-(SInit\ X) = (SInit\ (-X))$
using *ST37 ST36* **by** *auto*

lemma *ST47*:
$X \cup Y \cdot X = (SEmpty \cup Y) \cdot X$
by (*simp add*: *FusionUnionDistL*)

lemma *SStar01*:
assumes $X \cdot (SStar\ Y) \cup SEmpty \subseteq (SStar\ Y)$
shows $(SStar\ X) \subseteq (SStar\ Y)$
using *assms*
by (*metis Un-commute FusionSEmptyR SStarInductL*)

lemma *SStar03*:
$(SStar\ X) \cdot (SStar\ X) \subseteq (SStar\ X)$
by (*metis SStarInductL Un-absorb UnfoldL sup.absorb-iff1 sup.right-idem*)

lemma *SStar05*:
assumes $((SStar\ X) \cdot (SStar\ X)) \cup SEmpty \subseteq (SStar\ X)$
shows $(SStar\ (SStar\ X)) \subseteq (SStar\ X)$
using *assms*
by (*simp add*: *SStar01*)

lemma *SStar12*:
$(SEmpty \cup (X \cdot (SStar\ X))) \subseteq (SStar\ X)$
using *UnfoldL* **by** *blast*

lemma *SStar06*:
$((SStar\ X) \cdot (SStar\ X)) \cup SEmpty \subseteq (SStar\ X)$
using *SStar03 SStar12* **by** *force*

lemma *SStar07*:
$(SStar\ X) \subseteq (SStar\ (SStar\ X))$
by (*metis FusionUnionDistR FusionSEmptyR Subsumption Un-commute UnfoldL ST47 sup.right-idem*)

lemma *SStar08*:
$(SStar\ X) = (SStar\ (SStar\ X))$
by (*meson B04 SStar05 SStar06 SStar07*)

lemma *SStar15*:
$SEmpty \subseteq (SStar\ SSkip)$
by (*simp add*: *SStarSkip strue-def*)

lemma *SStar16*:
$SSkip \subseteq (SStar\ SSkip)$
by (*simp add*: *SStarSkip strue-def*)

lemma *SStar22*:
$(SEmpty \cap X) \cdot (SStar\ (SEmpty \cap X)) = (SEmpty \cap X)$
by (*metis ST15 FusionSEmptyR inf-commute*)

lemma *SStar23*:
 (*SStar* (*SEmpty* ∩ *X*)) = *SEmpty*
using *SStar22 UnfoldL* **by** *auto*

lemma *SStar25*:
 (*SStar STrue*) = *STrue*
by (*metis SStar08 SStarSkip*)

lemma *SStar28*:
 (*SStar X*)·*X* ⊆ *X*·(*SStar X*)
by (*metis B04 FusionUnionDistR FusionSEmptyR UnfoldL SStarInductL*)

lemma *SStar29*:
 X·(*SStar X*) ⊆ (*SStar X*)·*X*
by (*metis B04 SStar28 SStarInductR UnfoldL FusionRuleL ST47 sup.mono*)

lemma *SStar17*:
 (*SStar SSkip*)·*SSkip* ⊆ *SSkip*·(*SStar SSkip*)
by (*simp add*: *SStar28*)

lemma *SStar18*:
 SSkip·(*SStar SSkip*) ⊆ (*SStar SSkip*)·*SSkip*
by (*simp add*: *SStar29*)

lemma *SStar19*:
 (*SStar SSkip*)·*SSkip* = *SSkip*·(*SStar SSkip*)
using *SStar17 SStar18* **by** *auto*

lemma *SStar30*:
 X·(*SStar X*) = (*SStar X*)·*X*
using *SStar28 SStar29* **by** *auto*

lemma *SStar34*:
 assumes *SEmpty* ∪ (*X* ∪ *Y*)·((*SStar X*)·(*SStar* (*Y*·(*SStar X*)))) ⊆ (*SStar X*)·(*SStar* (*Y*·(*SStar X*)))
 shows (*SStar* (*X* ∪ *Y*)) ⊆ (*SStar X*)·(*SStar* (*Y*·(*SStar X*)))
by (*metis assms FusionSEmptyR SStarInductL*)

lemma *SStar35*:
 SEmpty ∪ (*X* ∪ *Y*)·((*SStar X*)·(*SStar* (*Y*·(*SStar X*)))) ⊆ (*SStar X*)·(*SStar* (*Y*·(*SStar X*)))
by (*smt B04 FusionAssoc FusionUnionDistL FusionSEmptyL UnfoldL UnionAssoc UnionCommute*)

lemma *SStar36*:
 (*SStar* (*X* ∪ *Y*)) ⊆ (*SStar X*)·(*SStar* (*Y*·(*SStar X*)))
using *SStar34 SStar35* **by** *blast*

lemma *SStar46*:
 (*SStar X*)·(*SStar* (*Y*·(*SStar X*))) ⊆ (*SStar* (*X* ∪ *Y*))
proof −
 have (*SEmpty* ∪ *SStar* (*X* ∪ *Y*) · *Y*) · *SStar X* ⊆ *SStar* (*X* ∪ *Y*)

by (*metis* (*no-types*) *FusionUnionDistR SStar12 SStar30 SStarInductR sup.bounded-iff*)
then show *?thesis* **by** (*simp add*: *SStarInductR ST47*)
qed

lemma *SStar47*:
$(SStar\ Z) = (SStar\ (Z \cap SMore))$
proof −
have *1*: $(SStar\ Z) = (SStar\ ((SEmpty \cap Z) \cup (SMore \cap Z)))$
by (*metis Int-Un-distrib2 compl-bot-eq inf-top.left-neutral smore-def strue-def STrueTop*)
have *2*: $(SStar\ ((SEmpty \cap Z) \cup (SMore \cap Z))) =$
$(SStar\ (SEmpty \cap Z)) \cdot (SStar\ ((SMore \cap Z) \cdot (SStar\ (SEmpty \cap Z))))$
by (*simp add*: *SStar36 SStar46 subset-antisym*)
have *3*: $(SStar\ (SEmpty \cap Z)) \cdot (SStar\ ((SMore \cap Z) \cdot (SStar\ (SEmpty \cap Z)))) =$
$(SStar\ (Z \cap SMore))$ **by** (*simp add*: *SStar23 inf-commute*)
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *SStar48*:
$(SStar\ SMore) = STrue$
by (*metis Compl-Un Compl-disjoint2 SStar25 SStar47 ST21 ST33 FusionSEmptyR inf.right-idem smore-def strue-def*)

lemma *SStar50*:
assumes $SSkip \cdot ((-X) \cup ((SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X))))) \cup (-X)$
$\subseteq (-X) \cup (SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X)))$
shows $((SStar\ SSkip) \cdot (-X)) \subseteq ((-X) \cup ((SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X)))))$
using *SStarInductL assms* **by** *blast*

lemma *SStar51*:
$SSkip \cdot ((-X) \cup ((SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X))))) \cup (-X)$
$\subseteq (-X) \cup (SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X)))$
by (*smt B20 double-compl FusionAssoc FusionUnionDistR inf-commute le-sup-iff UnfoldL ST47 Subsumption sup-ge2*)

lemma *SStar52*:
$(SStar\ X) \subseteq SEmpty \cup (X \cap SMore) \cdot (SStar\ X)$
by (*metis B04 SStar47 UnfoldL*)

lemma *SStar53*:
$SEmpty \cup (X \cap SMore) \cdot (SStar\ X) \subseteq (SStar\ X)$
by (*metis SStar12 SStar47*)

lemma *BD45*:
$(SBi\ ((-X) \cup X1)) \cap (X \cdot Y) \subseteq X1 \cdot Y$
proof −
have *1*: $(SBi\ ((-X) \cup X1)) = -(-((-X) \cup X1) \cdot (Y \cup (-Y)))$
by (*metis sbi-def sdi-def STrueTop*)
have *2*: $-(-((-X) \cup X1) \cdot (Y \cup (-Y))) \cap (X \cdot Y) \subseteq$
$-(-((-X) \cup X1) \cdot (Y)) \cap (X \cdot Y)$
by (*smt B01 B28 FusionUnionDistR inf-commute sup.absorb-iff1 sup-ge1*)

have *3*: $-(-((-X) \cup X1) \cdot (Y)) \cap (X \cdot Y) \subseteq (((-X) \cup X1) \cap X) \cdot Y$
by (*smt B09 Compl-disjoint2 FusionUnionDistL Huntington Morgan STrueTop UnionAssoc UnionCommute compl-inf sup-bot.left-neutral*)
have *4*: $(((-X) \cup X1) \cap X) \cdot Y \subseteq X1 \cdot Y$
by (*metis B20 double-compl FusionRuleL inf.right-idem inf-le1*)
from *1 2 3 4* **show** *?thesis* **by** *blast*
qed

lemma *BD46*:
$(SAlways\ ((-Y) \cup Y1)) \cap (X1 \cdot Y) \subseteq (X1 \cdot Y1)$
proof $-$
have *1*: $(SAlways\ ((-Y) \cup Y1)) = -((X1 \cup (-X1)) \cdot (-((-Y) \cup Y1)))$
by (*metis salways-def ssometime-def STrueTop*)
have *2*: $-((X1 \cup (-X1)) \cdot (-((-Y) \cup Y1))) \cap (X1 \cdot Y) \subseteq$
$-(X1 \cdot (-((-Y) \cup Y1))) \cap (X1 \cdot Y)$
by (*smt B01 B28 FusionUnionDistL inf-commute sup.absorb-iff2 sup-ge1*)
have *3*: $-(X1 \cdot (-((-Y) \cup Y1))) \cap (X1 \cdot Y) \subseteq X1 \cdot (((-Y) \cup Y1) \cap Y)$
by (*metis* (*no-types*, *lifting*) *B20 B04 compl-inf FusionUnionDistR Huntington Morgan Subsumption sup-ge1 UnionCommute*)
have *4*: $X1 \cdot (((-Y) \cup Y1) \cap Y) \subseteq (X1 \cdot Y1)$
by (*metis B20 double-compl FusionRuleR inf.right-idem inf-le1*)
from *1 2 3 4* **show** *?thesis* **by** *blast*
qed

8.5.2 ITL Axioms derived

lemma *SBoxGen*:
assumes $X = STrue$
shows $(SAlways\ X) = STrue$
using *assms*
by (*metis double-compl FusionSFalse salways-def sfalse-def ssometime-def strue-def*)

lemma *SBiGen*:
assumes $X = STrue$
shows $(SBi\ X) = STrue$
using *assms*
by (*metis double-compl sbi-def sdi-def sfalse-def SFalseFusion strue-def*)

lemma *SMP*:
assumes $X \subseteq Y$
assumes $X = STrue$
shows $Y = STrue$
using *assms*(*1*) *assms*(*2*)
using *strue-def* **by** *blast*

lemma *SChopAssoc*:
$X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z$
by *simp*

lemma *SOrChopImp*:

$(X \cup Y) \cdot Z \subseteq (X \cdot Z) \cup (Y \cdot Z)$
by (*simp add*: *FusionUnionDistL*)

lemma *SChopOrImp*:
$X \cdot (Y \cup Z) \subseteq (X \cdot Y) \cup (X \cdot Z)$
by (*simp add*: *FusionUnionDistR*)

lemma *SEmptyChop*:
$SEmpty \cdot X = X$
by *simp*

lemma *SChopEmpty*:
$X \cdot SEmpty = X$
by *simp*

lemma *SStateImpBi*:
$(SInit\ X) \subseteq (SBi\ (SInit\ X))$
by (*simp add*: *ST28 ST38 sbi-def*)

lemma *SNextImpNotNextNot*:
$(SNext\ X) \subseteq -(SNext\ (-X))$
proof $-$
have *1*: $((SNext\ X) \subseteq -(SNext\ (-X))) = (((SNext\ X) \cap (SNext\ (-X))) \subseteq SFalse)$
by (*simp add*: *disjoint-eq-subset-Compl sfalse-def*)
have *2*: $((SNext\ X) \cap (SNext\ (-X))) = SSkip \cdot (X \cap (-X))$
by (*metis CH07 SStar16 inf .orderE snext-def*)
have *3*: $(SSkip) \cdot (X \cap (-X)) = SSkip \cdot SFalse$
by (*simp add*: *sfalse-def*)
have *4*: $SSkip \cdot SFalse = SFalse$ **by** (*simp add*: *FusionSFalse*)
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *SBiBoxChopImpChop*:
$(SBi\ ((-X) \cup X1)) \cap (SAlways\ ((-Y) \cup Y1)) \cap (X \cdot Y) \subseteq (X1 \cdot Y1)$
using *BD45 BD46* **by** *blast*

lemma *SBoxInduct*:
$(SAlways\ (-X \cup (SWnext\ X))) \cap X \subseteq (SAlways\ X)$
proof $-$
have *1*: $((SAlways\ (-X \cup (SWnext\ X))) \cap X \subseteq (SAlways\ X)) =$
$((SSometime\ (-X)) \subseteq ((-X) \cup (SSometime\ (X \cap (SNext\ (-X))))))$
by (*smt Compl-subset-Compl-iff compl-sup double-compl inf-commute*
salways-def snext-def swnext-def)
have *2*: $((SSometime\ (-X)) \subseteq ((-X) \cup (SSometime\ (X \cap (SNext\ (-X)))))) =$
$(((SStar\ SSkip) \cdot (-X)) \subseteq ((-X) \cup ((SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X))))))$
by (*simp add*: *SStarSkip snext-def ssometime-def*)
have *3*: $(((SStar\ SSkip) \cdot (-X)) \subseteq ((-X) \cup ((SStar\ SSkip) \cdot (X \cap (SSkip \cdot (-X))))))$
using *SStar51 SStar50* **by** *blast*
from *1 2 3* **show** *?thesis* **by** *auto*
qed

lemma *SChopstarEqv*:
$(SStar\ X) = SEmpty \cup (X \cap SMore) \cdot (SStar\ X)$
using *SStar52 SStar53* **by** *blast*

8.6 Extra Laws

8.6.1 Boolean Laws

lemma *B02*:
assumes $-Y \subseteq -X$
shows $(X{::}\ 'a\ intervals) \subseteq Y$
using *assms* **by** *auto*

lemma *B03*:
$((X{::}\ 'a\ intervals) = Y\ ) \longleftrightarrow (-X = -Y)$
by *auto*

lemma *B05*:
assumes $(X{::}\ 'a\ intervals) \cup Y \subseteq Z$
shows $X \subseteq Z \wedge Y \subseteq Z$
using *assms* **by** *auto*

lemma *B06*:
assumes $X \subseteq Z \wedge Y \subseteq Z$
shows $(X{::}\ 'a\ intervals) \cup Y \subseteq Z$
using *assms* **by** *auto*

lemma *B07*:
$(X{::}\ 'a\ intervals) \cup Y \subseteq Z \longleftrightarrow$
$X \subseteq Z \wedge Y \subseteq Z$
by *auto*

lemma *B08*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y$
shows $-X \cup Y = STrue$
using *assms*
using *strue-def* **by** *auto*

lemma *B10*:
$(X{::}\ 'a\ intervals) \subseteq Y \longleftrightarrow -X \cup Y = STrue$
using *strue-def* **by** *auto*

lemma *B11*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y$
shows $X \cap -Y = SFalse$
using *assms sfalse-def* **by** *auto*

lemma *B12*:
assumes $X \cap -Y = SFalse$
shows $(X{::}\ 'a\ intervals) \subseteq Y$

using *assms sfalse-def* **by** *auto*

lemma *B13*:
$(X{::}\ 'a\ intervals) \subseteq Y \longleftrightarrow X \cap -Y = SFalse$
using *sfalse-def* **by** *auto*

lemma *B14*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y$
shows $X \cap Y = X$
using *assms* **by** *auto*

lemma *B15*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y \cap Z$
shows $X \subseteq Y \wedge X \subseteq Z$
using *assms* **by** *auto*

lemma *B16*:
assumes $X \subseteq Y \wedge X \subseteq Z$
shows $(X{::}\ 'a\ intervals) \subseteq Y \cap Z$
using *assms* **by** *auto*

lemma *B17*:
$(X{::}\ 'a\ intervals) \subseteq Y \cap Z \longleftrightarrow X \subseteq Y \wedge X \subseteq Z$
by *auto*

lemma *B18*:
assumes $(X{::}\ 'a\ intervals) \subseteq Y \cup Z$
shows $X \cap -Y \subseteq Z$
using *assms* **by** *auto*

lemma *B19*:
assumes $X \cap -Y \subseteq Z$
shows $(X{::}\ 'a\ intervals) \subseteq Y \cup Z$
using *assms* **by** *auto*

lemma *B21*:
$(X{::}\ 'a\ intervals) \cup (Y \cap Z) \subseteq (X \cup Y) \cap (X \cup Z) \longleftrightarrow$
$X \cup (Y \cap Z) \subseteq (X \cup Y) \wedge X \cup (Y \cap Z) \subseteq (X \cup Z)$
by *auto*

lemma *B22*:
$(X{::}\ 'a\ intervals) \cup (Y \cap Z) \subseteq X \cup Y$
by *auto*

lemma *B23*:
$(X{::}\ 'a\ intervals) \cup (Y \cap Z) \subseteq X \cup Z$
by *auto*

lemma *B24*:
$((X{::}\ 'a\ intervals) \cup Y) \cap (X \cup Z) \subseteq X \cup (Y \cap Z) \longleftrightarrow$

$((X \cup Y) \cap (X \cup Z)) \cap -X \subseteq Y \cap Z$
by *auto*

lemma *B25*:
$(((X{::}\ 'a\ intervals) \cup Y) \cap (X \cup Z)) \cap -X \subseteq Y \cap Z \longleftrightarrow$
$((X \cup Y) \cap (X \cup Z)) \cap -X \subseteq Y \wedge$
$((X \cup Y) \cap (X \cup Z)) \cap -X \subseteq Z$
by *auto*

lemma *B26*:
$(((X{::}\ 'a\ intervals) \cup Y) \cap (X \cup Z)) \cap -X \subseteq Y$
by *auto*

lemma *B27*:
$(((X{::}\ 'a\ intervals) \cup Y) \cap (X \cup Z)) \cap -X \subseteq Z$
by *auto*

lemma *B29*:
$(X{::}\ 'a\ intervals) \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$
by *auto*

8.6.2 Chop

lemma *CH02*:
$X{\cdot}Y \cap -(X{\cdot}Z) \subseteq X{\cdot}(Y \cap -Z)$
by (*metis B20 FusionRuleR FusionUnionDistR inf .right-idem inf-le1*)

lemma *CH03*:
$X{\cdot}Y \cap -(Z{\cdot}Y) \subseteq (X \cap -Z){\cdot}Y$
by (*metis B20 FusionRuleL FusionUnionDistL inf .right-idem inf-le1*)

lemma *CH04*:
$X{\cdot}Y \cap -(X{\cdot}-Z) \subseteq X{\cdot}(Y \cap Z)$
using *CH02* **by** *fastforce*

lemma *CH05*:
$X{\cdot}Y \cap -(-Z{\cdot}Y) \subseteq (X \cap Z){\cdot}Y$
using *CH03* **by** *fastforce*

lemma *CH06*:
assumes $X \subseteq X1$
$Y \subseteq Y1$
shows $X{\cdot}Y \subseteq X1{\cdot}Y1$
using *assms*(*1*) *assms*(*2*)
by (*smt FusionUnionDistL FusionUnionDistR UnionAssoc le-iff-sup*)

8.6.3 Next

lemma *N01*:
(*SNext SEmpty*) = *SSkip*
by (*simp add*: *snext-def*)

lemma *N02*:
  (*SNext SFalse*) = *SFalse*
by (*simp add*: *FusionSFalse snext-def*)

lemma *N03*:
  (*SNext X*)·*Y* = (*SNext* (*X*·*Y*))
by (*simp add*: *snext-def*)

lemma *N04*:
  (*SNext* (*X* ∪ *Y*)) = (*SNext X*) ∪ (*SNext Y*)
by (*simp add*: *FusionUnionDistR snext-def*)

lemma *N05*:
  (*SNext* (*X* ∩ *Y*)) = (*SNext X*) ∩ (*SNext Y*)
by (*metis CH07 SStar16 inf.orderE snext-def*)

lemma *N06*:
 assumes *X* ⊆ *Y*
 shows (*SNext X*) ⊆ (*SNext Y*)
using *assms*
by (*metis FusionUnionDistR Subsumption snext-def*)

lemma *N07*:
  (*SNext* ((−*X*) ∪ *Y*)) = (*SNext* (−*X*)) ∪ (*SNext Y*)
by (*simp add*: *N04*)

lemma *N08*:
  SMore ⊆ *SSkip*·*STrue*
by (*smt B09 Morgan SStar15 SStarSkip UnfoldL compl-inf inf.absorb2 smore-def*)

lemma *N23*:
 (*SWprev X*) ⊆ (*SEmpty* ∪ (*SPrev X*))
by (*metis B09 FusionUnionDistL SStar30 SStarSkip STrueTop UnfoldL UnionAssoc*
    abel-semigroup.commute double-compl sprev-def sup.abel-semigroup-axioms swprev-def)

lemma *N24*:
  (*SEmpty*) ⊆ (*SWprev X*)
by (*metis B10 B02 FusionRuleL SSkipFusionImpSMore SStar30 SStarSkip UnfoldL*
    compl-bot-eq double-compl smore-def strue-def subset-antisym swprev-def top-greatest)

lemma *N25*:
  (*SPrev X*) ⊆ (*SWprev X*)
proof −
 have *1*: ((*SPrev X*) ⊆ (*SWprev X*)) = (((*SPrev X*) ∩ (*SPrev* (−*X*)))⊆ *SFalse*)
    by (*simp add*: *B10 sfalse-def sprev-def swprev-def*)
 have *2*: ((*SPrev X*) ∩ (*SPrev* (−*X*))) = (*X* ∩ (−*X*))·*SSkip*
    by (*metis CH08 SStar16 inf.orderE sprev-def*)
 have *3*: (*X* ∩ (−*X*))·*SSkip* = *SFalse*·*SSkip*
    by (*simp add*: *sfalse-def*)

have *4*: *SFalse·SSkip* = *SFalse*
    by (*simp add*: *SFalseFusion*)
from *1 2 3 4* **show** *?thesis* **by** *auto*
qed

lemma *N26*:
(*SWprev X*) = (*SEmpty* ∪ (*SPrev X*))
using *N23 N24 N25* **by** *blast*

lemma *N09*:
SSkip ∪ *SMore·SSkip* ⊆ *SMore*
proof −
have *1*: *SSkip* ⊆ *SMore* **by** (*simp add*: *smore-def sskip-def*)
have *2*: *SMore·SSkip* ⊆ *SMore*
by (*metis N26 UnionCommute compl-le-swap1 smore-def sup-ge2 swprev-def*)
from *1 2* **show** *?thesis* **by** *auto*
qed

lemma *N10*:
assumes *SSkip* ∪ *SMore·SSkip* ⊆ *SMore*
shows *SSkip·*(*SStar SSkip*) ⊆ *SMore*
using *assms*
using *SStarInductR N09* **by** *blast*

lemma *N11*:
SSkip · *STrue* ⊆ *SMore*
by (*metis SStarSkip N09 N10*)

lemma *N12*:
(*SNext X*) = −(*SWnext* (−*X*))
by (*simp add*: *snext-def swnext-def*)

lemma *N13*:
SMore·STrue = *SMore*
by (*metis FusionAssoc N11 N08 SStar48 SStarSkip ST47 UnfoldL subset-antisym sup.right-idem*)

lemma *N14*:
STrue·SSkip ⊆ *SMore*
by (*metis N11 SStar19 SStarSkip*)

lemma *N15*:
SMore ⊆ *STrue·SSkip*
by (*metis N08 SStar19 SStarSkip*)

lemma *N19*:
(*SWnext X*) ⊆ (*SEmpty* ∪ (*SNext X*))
by (*smt B02 B20 B09 FusionUnionDistR Morgan SStarSkip STrueTop UnfoldL compl-inf*
    inf-sup-absorb snext-def swnext-def)

lemma *N20*:

$(SEmpty\ ) \subseteq (SWnext\ X)$
proof $-$
have *1*: $((SEmpty\ ) \subseteq (SWnext\ X)\ ) = (-(SWnext\ X) \subseteq SMore)$
by (*simp add*: *smore-def*)
have *2*: $(-(SWnext\ X) \subseteq SMore) = ((SNext\ (-X)) \subseteq SMore)$
by (*simp add*: *snext-def swnext-def*)
have *3*: $(SNext\ (-X)) \subseteq SSkip \cdot STrue$
by (*metis FusionUnionDistR STrueTop snext-def sup.orderI sup.right-idem*)
hence *4*: $(SNext\ (-X)) \subseteq SMore$ **using** *SSkipFusionImpSMore* **by** *auto*
from *1 2 4* **show** *?thesis* **by** *auto*
qed

lemma *N21*:
$(SEmpty \cup (SNext\ X)) \subseteq (SWnext\ X)$
using *N20*
by (*metis B06 SNextImpNotNextNot snext-def swnext-def*)

lemma *N22*:
$(SWnext\ X) = (SEmpty \cup (SNext\ X))$
using *N21 N19* **by** *blast*

lemma *N16*:
$(SNext\ X) = SMore \cap (SWnext\ X)$
using *N12 N22 smore-def* **by** *blast*

lemma *N17*:
$(SWnext\ (X \cap Y)) = (SWnext\ X) \cap (SWnext\ Y)$
by (*simp add*: *N05 N22 Un-Int-distrib*)

lemma *N18*:
$(SWnext\ (X \cup Y)) = (SWnext\ X) \cup (SWnext\ Y)$
by (*smt CH07 SStar16 compl-sup double-compl inf.orderE swnext-def*)

lemma *N27*:
$(SNext\ ((-X) \cup Y)) \subseteq (-(SNext\ X) \cup (SNext\ Y))$
by (*metis N12 N16 N18 Un-Int-distrib double-compl sup-ge2 sup-left-idem*)

lemma *N28*:
$(SPrev\ ((-X) \cup Y)) \subseteq (\ -(SPrev\ X) \cup (SPrev\ Y))$
by (*metis B01 B05 B06 FusionUnionDistL Huntington N25 double-compl sprev-def sup-ge2 swprev-def*)

lemma *N29*:
$(SPrev\ X) = -(SWprev\ (-X)\ )$
by (*simp add*: *sprev-def swprev-def*)

8.6.4 SInit

lemma *ST01*:
$(X \cap SEmpty) \cdot Y \subseteq Y$
by (*metis Int-commute FusionUnionDistL FusionSEmptyL le-iff-sup sup-inf-absorb UnionCommute*)

lemma *ST02*:
$(X \cap SEmpty) \cdot Y \subseteq (X \cap SEmpty) \cdot STrue$
by (*simp add*: *FusionRuleR strue-def*)

lemma *ST03*:
$(X \cap SEmpty) \cdot (X \cap SEmpty) \subseteq (X \cap SEmpty)$
using *ST01* **by** *auto*

lemma *ST04*:
$(X \cap SEmpty) \subseteq (X \cap SEmpty) \cdot (X \cap SEmpty)$
by (*metis B04 Int-commute FusionSEmptyL FusionSEmptyR inf .right-idem inf-top.right-neutral CH10*)

lemma *ST05*:
$(X \cap SEmpty) \subseteq -((-X) \cap SEmpty)$
by *blast*

lemma *ST06*:
$((-X) \cap SEmpty) \subseteq -(X \cap SEmpty)$
by *auto*

lemma *ST07*:
$(X \cap SEmpty) \cap (Y \cap SEmpty) \subseteq (X \cap SEmpty) \cdot STrue$
using *ST02 FusionSEmptyR* **by** *blast*

lemma *ST08*:
$(X \cap SEmpty) \cap (Y \cap SEmpty) \subseteq (STrue \cap SEmpty) \cdot (Y \cap SEmpty)$
by (*simp add*: *le-infI2 strue-def*)

lemma *ST09*:
$((X \cap SEmpty) \cdot STrue) \cap (STrue \cap SEmpty) \cdot (Y \cap SEmpty) \subseteq (X \cap SEmpty) \cdot (Y \cap SEmpty)$
by (*metis compl-bot-eq eq-refl FusionSEmptyR inf .commute inf-top.left-neutral CH09 strue-def*)

lemma *ST10*:
$(X \cap SEmpty) \cdot (Y \cap SEmpty) \subseteq (X \cap SEmpty)$
by (*metis FusionRuleR FusionSEmptyR inf-le2*)

lemma *ST11*:
$(X \cap SEmpty) \cdot (Y \cap SEmpty) \subseteq (Y \cap SEmpty)$
using *ST01* **by** *blast*

lemma *ST12*:
$(X \cap SEmpty) \cap (Y \cap SEmpty) = (X \cap SEmpty) \cdot SEmpty \cap (Y \cap SEmpty) \cdot SEmpty$
by *simp*

lemma *ST14*:
$((X \cap Y) \cap SEmpty) \cdot SEmpty = ((X \cap Y) \cap SEmpty)$
by *auto*

lemma *ST16*:
$(X \cap SEmpty) \cap (Y \cap SEmpty) \subseteq SEmpty$
by (*simp add*: *le-infI2*)

lemma *ST17*:
$(X \cap SEmpty) \cdot (Y \cap SEmpty) \subseteq SEmpty$
using *ST10* **by** *auto*

lemma *ST18*:
$-((X \cap SEmpty) \cup (Y \cap SEmpty)) = -(X \cap SEmpty) \cap -(Y \cap SEmpty)$
by *auto*

lemma *ST19*:
$(X \cap SEmpty) \cdot ((-X) \cap SEmpty) \subseteq (X \cap SEmpty)$
using *ST10* **by** *blast*

lemma *ST20*:
$(X \cap SEmpty) \cdot ((-X) \cap SEmpty) \subseteq ((-X) \cap SEmpty)$
using *ST01* **by** *auto*

lemma *ST22*:
$((X \cap SEmpty) \cdot SSkip) \cdot (Y \cap SEmpty) \subseteq (X \cap SEmpty) \cdot SSkip$
using *FusionRuleR FusionSEmptyR* **by** *blast*

lemma *ST23*:
$((X \cap SEmpty) \cdot SSkip) \cdot (Y \cap SEmpty) \subseteq SSkip \cdot (Y \cap SEmpty)$
by (*simp add*: *ST01 FusionRuleL*)

lemma *ST27*:
$(SInit\ X) \cap (Y \cdot Z) \subseteq ((SInit\ X) \cap Y) \cdot Z$
by (*metis B04 compl-bot-eq FusionAssoc FusionSEmptyL inf-commute inf-top.left-neutral CH09 sinit-def strue-def*)

lemma *ST29*:
$(SInit\ X) \cdot Y \subseteq (SInit\ X)$
using *ST02 FusionAssoc sinit-def* **by** *fastforce*

lemma *ST30*:
$(SInit\ X) \cap (SDi\ Y) = (SDi\ (\ (SInit\ X) \cap Y))$
by (*metis FusionAssoc FusionSEmptyL CH09 compl-bot-eq inf-top.left-neutral sdi-def sinit-def strue-def*)

lemma *ST31*:
$(X \cdot (STrue \cap SEmpty)) \cap (STrue \cdot (Y \cap SEmpty)) = X \cdot (Y \cap SEmpty)$
by (*metis Int-commute compl-bot-eq inf-top.right-neutral CH10 strue-def*)

lemma *ST32*:
$(STrue \cap SEmpty) \cdot SEmpty \cap (SInit\ X) = (X \cap SEmpty)$
by (*metis Compl-empty-eq Int-commute CH09 ST14 inf-top.right-neutral*

sinit-def strue-def)

lemma *ST34*:
$((X \cap SEmpty) \cdot Y) = (SInit\ X) \cap Y$
by (*metis FusionSEmptyL Int-commute CH09 compl-bot-eq inf-top-right sinit-def strue-def*)

lemma *ST35*:
$((SInit\ X) \cap Y) \cdot Z \subseteq (SInit\ X) \cap (Y \cdot Z)$
by (*metis B04 ST34 FusionAssoc*)

lemma *ST39*:
$SEmpty \cap (SInit\ X) \subseteq (X \cap SEmpty)$
using *ST32* **by** *blast*

lemma *ST40*:
$(X \cap SEmpty) \subseteq SEmpty \cap (SInit\ X)$
using *ST32* **by** *auto*

lemma *ST41*:
$SEmpty \cap (SInit\ X) = (X \cap SEmpty)$
using *ST40 ST39* **by** *auto*

lemma *ST42*:
$(X \cap SEmpty) \subseteq ((X \cup Y) \cap SEmpty)$
by *blast*

lemma *ST43*:
$(Y \cap SEmpty) \subseteq ((X \cup Y) \cap SEmpty)$
by *blast*

lemma *ST44*:
$(X \cap SEmpty) \cap ((-X) \cap SEmpty) = SFalse$
by (*simp add*: *sfalse-def*)

lemma *ST45*:
$((X \cup Y) \cap SEmpty) \subseteq (X \cap SEmpty) \cup (Y \cap SEmpty)$
by *auto*

lemma *ST46*:
$(SInit\ X) \cup (SInit\ Y) = (SInit\ (X \cup Y))$
by (*simp add*: *Int-Un-distrib2 FusionUnionDistL sinit-def*)

8.6.5 SStar

lemma *SStar02*:
assumes $X \subseteq Y$
shows $X \cdot (SStar\ Y) \cup SEmpty \subseteq (SStar\ Y)$
using *assms*
by (*smt FusionUnionDistL semigroup.assoc UnfoldL Subsumption sup.semigroup-axioms sup-left-idem UnionCommute*)

lemma *SStar04*:
$(SStar\ X) \subseteq (SStar\ X) \cdot (SStar\ X)$
by (*metis Un-absorb UnfoldL UnionAssoc ST47 sup.absorb-iff2*)

lemma *SStar09*:
assumes $(X \cdot (SEmpty \cup (X \cdot (SStar\ X)))) \cup SEmpty \subseteq (SEmpty \cup (X \cdot (SStar\ X)))$
shows $(SStar\ X) \subseteq SEmpty \cup (X \cdot (SStar\ X))$
using *assms*
by (*simp add*: *UnfoldL*)

lemma *SStar10*:
$(X \cdot (SEmpty \cup (X \cdot (SStar\ X)))) \subseteq (SEmpty \cup (X \cdot (SStar\ X)))$
by (*metis UnfoldL sup-ge2*)

lemma *SStar11*:
$SEmpty \subseteq (SEmpty \cup (X \cdot (SStar\ X)))$
by *auto*

lemma *SStar13*:
$(SStar\ SSkip) = STrue$
by (*simp add*: *SStarSkip*)

lemma *SStar14*:
$(SSometime\ X) = (SStar\ SSkip) \cdot X$
by (*simp add*: *SStarSkip ssometime-def*)

lemma *SStar20*:
$(SStar\ SEmpty) = SEmpty$
by (*metis FusionSEmptyR ST15 ST33*)

lemma *SStar21*:
$(SStar\ (SEmpty \cap X)) \cdot (SEmpty \cap X) = (SEmpty \cap X)$
by (*metis ST15 FusionSEmptyL inf-commute*)

lemma *SStar24*:
$(SStar\ SFalse) = SEmpty$
by (*metis SStar20 SStar47 inf-compl-bot sfalse-def smore-def*)

lemma *SStar26*:
$X \subseteq (SStar\ X)$
by (*smt FusionUnionDistR FusionSEmptyR SStar03 SStar04 Subsumption UnfoldL UnionAssoc UnionCommute*)

lemma *SStar27*:
$SEmpty \subseteq (SStar\ X)$
using *UnfoldL* **by** *blast*

lemma *SStar31*:
assumes $X \cup (X \cdot Y) \cdot (X \cdot (SStar\ (Y \cdot X))) \subseteq X \cdot (SStar\ (Y \cdot X))$
shows $(SStar\ (X \cdot Y)) \cdot X \subseteq X \cdot (SStar\ (Y \cdot X))$

using *assms SStarInductL* **by** *blast*

lemma *SStar32*:
$X \cup (X \cdot Y) \cdot (X \cdot (SStar\ (Y \cdot X))) \subseteq X \cdot (SStar\ (Y \cdot X))$
by (*metis B06 SStar10 SStar11 FusionAssoc FusionRuleR FusionSEmptyR UnfoldL*)

lemma *SStar33*:
$(SStar\ (X \cdot Y)) \cdot X \subseteq X \cdot (SStar\ (Y \cdot X))$
using *SStar31 SStar32* **by** *blast*

lemma *SStar37*:
assumes $X \cdot Z \subseteq Z \cdot Y$
shows $(SStar\ X) \cdot Z \subseteq Z \cdot (SStar\ Y)$
by (*smt Un-commute assms FusionAssoc FusionUnionDistL FusionUnionDistR FusionSEmptyR semigroup.assoc UnfoldL SStarInductL Subsumption sup.right-idem sup.semigroup-axioms*)

lemma *SStar38*:
assumes $Z \cdot X \subseteq Y \cdot Z$
shows $Z \cdot (SStar\ X) \subseteq (SStar\ Y) \cdot Z$
by (*smt SStar30 assms FusionAssoc FusionUnionDistL FusionUnionDistR FusionSEmptyL semigroup.assoc UnfoldL SStarInductR Subsumption sup.right-idem sup.semigroup-axioms UnionCommute*)

lemma *SStar39*:
$Y \cdot (SStar\ ((SStar\ X) \cdot Y)) \subseteq (SStar\ (Y \cdot (SStar\ X))) \cdot Y$
by (*simp add*: *SStar38*)

lemma *SStar40*:
$(SStar\ (Y \cdot (SStar\ X))) \cdot Y \subseteq Y \cdot (SStar\ ((SStar\ X) \cdot Y))$
by (*simp add*: *SStar33*)

lemma *SStar41*:
$Y \cdot (SStar\ ((SStar\ X) \cdot Y)) = (SStar\ (Y \cdot (SStar\ X))) \cdot Y$
using *SStar39 SStar40* **by** *blast*

lemma *SStar42*:
$Z \cdot (SStar\ (Y \cdot Z)) \subseteq (SStar\ (Z \cdot Y)) \cdot Z$
by (*simp add*: *SStar38*)

lemma *SStar43*:
$(SStar\ (Z \cdot Y)) \cdot Z \subseteq Z \cdot (SStar\ (Y \cdot Z))$
by (*simp add*: *SStar33*)

lemma *SStar44*:
$Z \cdot (SStar\ (Y \cdot Z)) = (SStar\ (Z \cdot Y)) \cdot Z$
using *SStar42 SStar43* **by** *blast*

lemma *SStar49*:
$(SStar\ X) = SEmpty \cup (SStar\ X) \cdot X$
using *SStar30 UnfoldL* **by** *blast*

8.6.6 Box and Diamond

lemma *BD01*:
 $(SSometime\ SEmpty) = STrue$
by (*simp add*: *ssometime-def*)

lemma *BD02*:
 $X \subseteq (SSometime\ X)$
by (*metis FusionUnionDistL SEmptyChop STrueTop Subsumption Un-absorb semigroup.assoc ssometime-def sup.semigroup-axioms*)

lemma *BD03*:
 $(SNext\ (SSometime\ X)) \subseteq (SSometime\ X)$
by (*metis FusionUnionDistL N03 SStar16 SStar03 SStar04 SStarSkip snext-def ssometime-def sup.absorb-iff2 sup.orderE*)

lemma *BD04*:
 $(SSometime\ (SNext\ X)) \subseteq (SSometime\ X)$
by (*metis CH01 FusionAssoc FusionUnionDistL FusionUnionDistR SStar16 SStarSkip snext-def ssometime-def sup.absorb-iff2*)

lemma *BD05*:
 $(SSometime\ X) \cup (SSometime\ Y) = (SSometime\ (X \cup Y))$
by (*simp add*: *FusionUnionDistR ssometime-def*)

lemma *BD06*:
 $(SSometime\ STrue) = STrue$
by (*simp add*: *CH01 ssometime-def*)

lemma *BD07*:
 $(SSometime\ (X \cap Y)) \subseteq (SSometime\ X) \cap (SSometime\ Y)$
by (*simp add*: *FusionRuleR ssometime-def*)

lemma *BD08*:
 $(SAlways\ STrue) = STrue$
by (*simp add*: *SBoxGen*)

lemma *BD09*:
 $-(SAlways\ X) = (SSometime\ (-X))$
by (*simp add*: *salways-def*)

lemma *BD10*:
 $(SAlways\ X) \subseteq (SSometime\ X)$
by (*metis B02 BD02 BD09 set-rev-mp subsetI*)

lemma *BD11*:
 $(SSometime\ (SSometime\ X)) = (SSometime\ X)$
by (*simp add*: *CH01 ssometime-def*)

lemma *BD12*:
 $(SAlways\ X) \subseteq X$

by (*simp add*: *B02 BD02 BD09*)

lemma *BD13*:
 (*SDi STrue*) = *STrue*
by (*simp add*: *CH01 sdi-def*)

lemma *BD14*:
 (*SDi SEmpty*) = *STrue*
by (*simp add*: *sdi-def*)

lemma *BD15*:
 (*SBi STrue*) = *STrue*
by (*simp add*: *SBiGen*)

lemma *BD16*:
 (*SDi* ($X \cup Y$)) = (*SDi* X) $\cup$ (*SDi* Y)
by (*simp add*: *FusionUnionDistL sdi-def*)

lemma *BD17*:
 assumes $X \subseteq Y$
 shows (*SDi* X) $\subseteq$ (*SDi* Y)
using *assms*
by (*metis FusionUnionDistL Subsumption sdi-def*)

lemma *BD18*:
 (*SDi* (*SDi* X)) = (*SDi* X)
by (*metis CH01 FusionAssoc sdi-def*)

lemma *BD19*:
 (*SDa SEmpty*) = *STrue*
by (*simp add*: *CH01 sda-def*)

lemma *BD20*:
 (*SDa STrue*) = *STrue*
by (*simp add*: *CH01 sda-def*)

lemma *BD21*:
 (*SBa STrue*) = *STrue*
by (*metis BD15 BD08 BD09 sba-def sbi-def sda-def sdi-def ssometime-def*)

lemma *BD22*:
 (*SDa* ($X \cup Y$)) = (*SDa* X) $\cup$ (*SDa* Y)
by (*simp add*: *FusionUnionDistL FusionUnionDistR sda-def*)

lemma *BD23*:
 assumes $X \subseteq Y$
 shows (*SDa* X) $\subseteq$ (*SDa* Y)
using *assms*
by (*metis BD22 Subsumption*)

lemma *BD24*:
assumes $X \subseteq Y$
shows $(SDa\ (-Y)) \subseteq (SDa\ (-X))$
using *assms*
by (*simp add*: *BD23*)

lemma *BD25*:
$(SDi\ X) \subseteq (SDa\ X)$
by (*metis BD02 FusionAssoc sda-def sdi-def ssometime-def*)

lemma *BD26*:
$(SSometime\ X) \subseteq (SDa\ X)$
by (*metis BD01 BD02 FusionSEmptyR FusionUnionDistR SStar14 le-iff-sup sda-def*)

lemma *BD27*:
$(SBa\ X) \subseteq (SBi\ X)$
by (*simp add*: *BD25 sba-def sbi-def*)

lemma *BD28*:
$(SBa\ X) \subseteq (SAlways\ X)$
by (*simp add*: *B02 BD26 BD09 sba-def*)

lemma *BD29*:
$(SAlways\ X) \cap (SAlways\ Y) = (SAlways\ (X \cap Y))$
by (*metis BD05 BD09 Morgan compl-inf salways-def*)

lemma *BD30*:
$(SAlways\ X) \cup (SAlways\ Y) \subseteq (SAlways\ (X \cup Y))$
using *BD07*
by (*metis B02 BD09 compl-sup*)

lemma *BD31*:
$(SDi\ (X \cap Y)) \subseteq (SDi\ X) \cap (SDi\ Y)$
by (*simp add*: *BD17*)

lemma *BD32*:
$(SBi\ X) \cup (SBi\ Y) \subseteq (SBi\ (X \cup Y))$
using *BD31*
by (*metis* (*mono-tags*, *lifting*) *B02 compl-sup double-compl sbi-def*)

lemma *BD33*:
$(SDa\ (X \cap Y)) \subseteq (SDa\ X) \cap (SDa\ Y)$
by (*simp add*: *BD23*)

lemma *BD34*:
$(SBa\ X) \cup (SBa\ Y) \subseteq (SBa\ (X \cup Y))$
using *BD33*
by (*metis* (*mono-tags*, *lifting*) *B02 compl-sup double-compl sba-def*)

lemma *BD35*:

$(SAlways\ SEmpty) = SEmpty$
by (*metis N13 SStar14 SStar30 SStar48 SStarSkip double-complement salways-def smore-def*)

lemma *BD36*:
$(SBi\ SEmpty) = SEmpty$
using *N13 sbi-def sdi-def smore-def* **by** *fastforce*

lemma *BD37*:
$(SBa\ SEmpty) = SEmpty$
by (*metis N13 SStar30 SStar48 double-complement sba-def sda-def smore-def*)

lemma *BD38*:
assumes $X \subseteq Y$
shows $(SAlways\ X) \subseteq (SAlways\ Y)$
using *assms*
by (*simp add*: *BD29 inf.absorb-iff2*)

lemma *BD39*:
assumes $X \subseteq Y$
shows $(SBi\ X) \subseteq (SBi\ Y)$
using *assms*
by (*simp add*: *BD17 sbi-def*)

lemma *BD40*:
assumes $X \subseteq Y$
shows $(SBa\ X) \subseteq (SBa\ Y)$
using *assms*
by (*simp add*: *BD24 sba-def*)

lemma *BD41*:
$(SBi\ (SBi\ X)) = (SBi\ X)$
by (*simp add*: *BD18 sbi-def*)

lemma *BD42*:
$(SAlways\ (SAlways\ X)) = (SAlways\ X)$
by (*simp add*: *BD11 salways-def*)

lemma *BD43*:
$(SDa\ (SDa\ X)) = (SDa\ X)$
by (*metis CH01 FusionAssoc sda-def*)

lemma *BD44*:
$(SBa\ (SBa\ X)) = (SBa\ X)$
by (*simp add*: *BD43 sba-def*)

lemma *BD47*:
$(SAlways\ (\ (-X) \cup Y)) \subseteq (\ -(SAlways\ X) \cup (SAlways\ Y))$
by (*metis B20 BD12 BD29 BD38 BD42 double-compl*)

lemma *BD48*:

$(SAlways\ X) \subseteq X \cap (SWnext\ (SAlways\ X))$
by (*metis B02 B16 BD03 BD09 BD12 N12 salways-def*)

lemma *BD49*:
$(SBi\ (\ (-X) \cup Y)) \subseteq (-(SBi\ X) \cup (SBi\ Y)\ )$
by (*smt B20 FusionUnionDistL Huntington Subsumption compl-inf double-compl inf-commute sbi-def sdi-def sup-ge2 sup-left-commute*)

lemma *BD50*:
$(SPrev\ (SDi\ X)) \subseteq (SDi\ X)$
by (*metis BD13 FusionAssoc FusionUnionDistR SStar16 SStarSkip Subsumption sdi-def sprev-def*)

lemma *BD51*:
$-(SBi\ X) = (SDi\ (-X))$
by (*simp add*: *sbi-def*)

lemma *BD52*:
$X \subseteq (SDi\ X)$
by (*metis FusionSEmptyR FusionUnionDistR ST33 Subsumption UnionCommute sdi-def sup-inf-absorb*)

lemma *BD53*:
$(SBi\ X) \subseteq X$
by (*simp add*: *B02 BD51 BD52*)

lemma *BD54*:
$(SBi\ X) \subseteq X \cap (SWprev\ (SBi\ X))$
by (*metis B02 B16 BD50 BD51 BD53 N29 sbi-def*)

8.6.7 Time Reversal

lemma *TR01*:
$(SRev\ SMore) = SMore$
by (*simp add*: *SRevSEmpty SRevSNot smore-def*)

lemma *TR02*:
$(SRev\ SSkip) = SSkip$
by (*metis SRevFusion SRevSEmpty SRevSNot SRevUnion TR01 sskip-def*)

lemma *TR03*:
$(SRev\ STrue) = STrue$
by (*metis SRevSStar SStar48 TR01*)

lemma *TR04*:
$(SRev\ SFalse) = SFalse$
by (*metis B03 SRevSNot TR03 sfalse-def strue-def*)

lemma *TR05*:
$(SRev\ (SSometime\ X)) = (SDi\ (SRev\ X))$
by (*simp add*: *SRevFusion TR03 sdi-def ssometime-def*)

lemma *TR06*:
$(SRev\ (SAlways\ X)) = (SBi\ (SRev\ X))$
by (*simp add*: *SRevSNot TR05 salways-def sbi-def*)

lemma *TR07*:
$(SRev\ (SDi\ X)) = (SSometime\ (SRev\ X))$
by (*simp add*: *SRevFusion TR03 sdi-def ssometime-def*)

lemma *TR08*:
$(SRev\ (SBi\ X)) = (SAlways\ (SRev\ X))$
by (*metis SRevSRev TR06*)

lemma *TR09*:
$(SRev\ (SNext\ X)) = (SPrev\ (SRev\ X))$
by (*simp add*: *SRevFusion TR02 snext-def sprev-def*)

lemma *TR10*:
$(SRev\ (SWnext\ X)) = (SWprev\ (SRev\ X))$
by (*simp add*: *SRevFusion SRevSNot TR02 swnext-def swprev-def*)

lemma *TR11*:
$(SRev\ (SPrev\ X)) = (SNext\ (SRev\ X))$
by (*simp add*: *SRevFusion TR02 snext-def sprev-def*)

lemma *TR12*:
$(SRev\ (SWprev\ X)) = (SWnext\ (SRev\ X))$
by (*metis SRevSRev TR10*)

lemma *TR13*:
$(SRev\ (SDa\ X)) = (SDa\ (SRev\ X))$
by (*simp add*: *SRevFusion TR03 sda-def*)

lemma *TR14*:
$(SRev\ (SBa\ X)) = (SBa\ (SRev\ X))$
by (*simp add*: *SRevSNot TR13 sba-def*)

lemma *TR15*:
$(SRev\ (SPower\ SSkip\ n)) = (SPower\ SSkip\ n)$
by (*simp add*: *SRevSPower TR02*)

8.7 Link between Set of Intervals and ITL

lemma *interval-lan* [*simp*]:
$\sigma \in (lan\ f) \longleftrightarrow (\sigma \models f)$
by (*simp add*: *lan-def*)

lemma *valid-lan-eqv* :
$(\ (lan\ f) = (lan\ g)\ ) \longleftrightarrow (\ \vdash f \equiv_i g\ )$
using *interval-lan lan-def* **by** *force*

lemma *valid-lan-imp* :
$((lan\ f) \subseteq (lan\ g)) \longleftrightarrow (\vdash f \supset_i g)$
by (*meson implies-defs interval-lan subset-eq valid-def*)

lemma *valid-strue* :
$((lan\ f) = STrue) \longleftrightarrow (\vdash f)$
using *strue-def* **by** *fastforce*

lemma *strue-true*:
$\sigma \in STrue \longleftrightarrow (\sigma \models true_i)$
by (*simp add*: *strue-elim*)

lemma *strue-true-1*:
$STrue = (lan\ true_i)$
using *lan-def strue-true* **by** *fastforce*

lemma *sfalse-false*:
$\sigma \in SFalse \longleftrightarrow (\sigma \models false_i)$
by (*simp add*: *sfalse-def*)

lemma *sfalse-false-1*:
$SFalse = (lan\ false_i)$
using *sfalse-false* **using** *lan-def* **by** *fastforce*

lemma *not-negation*:
$\sigma \in (-(lan\ f)) \longleftrightarrow (\sigma \models \neg_i f)$
by *simp*

lemma *not-negation-1*:
$-(lan\ f\ ) = (lan\ (\neg_i f))$
using *interval-lan lan-def* **by** *fastforce*

lemma *inter-and*:
$(\sigma \in ((lan\ f) \cap (lan\ g))) \longleftrightarrow (\sigma \models f \wedge_i g)$
by (*simp add*: *lan-def*)

lemma *inter-and-1*:
$((lan\ f) \cap (lan\ g)) = (lan\ (f \wedge_i g))$
using *inter-and lan-def* **by** *fastforce*

lemma *union-or*:
$(\sigma \in ((lan\ f) \cup (lan\ g))) \longleftrightarrow (\sigma \models f \vee_i g)$
by (*simp add*: *lan-def*)

lemma *union-or-1*:
$((lan\ f) \cup (lan\ g)) = (lan\ (f \vee_i g))$
using *union-or lan-def* **by** *fastforce*

lemma *subset-impl*:
$(\sigma \in (-(lan\ f) \cup (lan\ g))) \longleftrightarrow (\sigma \models f \supset_i g)$

by *simp*

lemma *subset-impl-1*:
$(-(lan\ f) \cup (lan\ g)) = (lan\ (f \supset_l g))$
using *subset-impl lan-def* **by** *fastforce*

lemma *fusion-chop*:
$(\sigma \in ((lan\ f) \cdot (lan\ g))) \longleftrightarrow (\sigma \models f;g)$
by (*simp add*: *chop-fuse fusion-iff*)

lemma *fusion-chop-1*:
$((lan\ f) \cdot (lan\ g)) = (lan\ (f;g))$
using *fusion-chop lan-def* **by** *blast*

lemma *sempty-empty*:
$\sigma \in SEmpty \longleftrightarrow (\sigma \models empty)$
by (*simp add*: *sempty-elim*)

lemma *sempty-empty-1*:
$SEmpty = (lan\ empty)$
using *sempty-empty lan-def* **by** *fastforce*

lemma *smore-more*:
$\sigma \in SMore \longleftrightarrow (\sigma \models more)$
by (*simp add*: *smore-elim*)

lemma *smore-more-1*:
$SMore = (lan\ more)$
using *smore-more lan-def* **by** *fastforce*

lemma *sskip-skip*:
$\sigma \in SSkip = (\sigma \models skip)$
by (*simp add*: *sskip-elim*)

lemma *sskip-skip-1*:
$SSkip = (lan\ skip)$
using *sskip-skip lan-def* **by** *fastforce*

lemma *snext-next*:
$\sigma \in (SNext\ (lan\ f)) \longleftrightarrow (\sigma \models \bigcirc\ f)$
by (*metis snext-def fusion-chop next-d-def sskip-skip-1*)

lemma *snext-next-1*:
$(SNext\ (lan\ f)) = (lan\ (\bigcirc\ f))$
using *snext-next lan-def* **by** *fastforce*

lemma *swnext-wnext*:
$\sigma \in (SWnext\ (lan\ f)) \longleftrightarrow (\sigma \models wnext\ f)$
by (*simp add*: *swnext-def fusion-chop-1 next-d-def not-negation-1 sskip-skip-1 wnext-d-def*)

lemma *swnext-wnext-1*:
$(SWnext\ (lan\ f)) = (lan\ (wnext\ f))$
using *swnext-wnext lan-def* **by** *fastforce*

lemma *sprev-prev*:
$\sigma \in (SPrev\ (lan\ f)) \longleftrightarrow (\sigma \models prev\ f)$
by (*metis fusion-chop prev-d-def sprev-def sskip-skip-1*)

lemma *sprev-prev-1*:
$(SPrev\ (lan\ f)) = (lan\ (prev\ f))$
using *sprev-prev lan-def* **by** *fastforce*

lemma *swprev-wprev*:
$\sigma \in (SWprev\ (lan\ f)) \longleftrightarrow (\sigma \models wprev\ f)$
by (*simp add*: *fusion-chop-1 not-negation-1 prev-d-def sskip-skip-1 swprev-def wprev-d-def*)

lemma *swprev-wprev-1*:
$(SWprev\ (lan\ f)) = (lan\ (wprev\ f))$
using *swprev-wprev lan-def* **by** *fastforce*

lemma *sinit-init*:
$\sigma \in SInit\ (lan\ f) \longleftrightarrow (\sigma \models init\ f)$
by (*simp add*: *Int-commute fusion-chop-1 init-d-def inter-and-1 sempty-empty-1 sinit-def strue-true-1*)

lemma *sinit-init-1*:
$SInit\ (lan\ f) = (lan\ (init\ f))$
using *sinit-init lan-def* **by** *fastforce*

lemma *and-inter-more*:
$\sigma \in (((lan\ f) \cap SMore)) \longleftrightarrow (\sigma \models (f\ \wedge_i\ more)\ )$
using *smore-more inter-and* **by** *auto*

lemma *and-inter-more-1*:
$\sigma \in (((lan\ f) \cap SMore)) \longleftrightarrow (\sigma \in (lan\ (f\ \wedge_i\ more)\ ))$
using *and-inter-more lan-def* **by** *fastforce*

lemma *and-inter-more-2*:
$((lan\ f) \cap SMore) = \ (lan\ (f\ \wedge_i\ more)\ )$
using *and-inter-more-1* **by** *blast*

lemma *and-chop*:
$\sigma \in (((lan\ f) \cap SMore) \cdot\ (lan\ g)) \longleftrightarrow (\sigma \models (f\ \wedge_i\ more);g)$
by (*metis fusion-chop inter-and-1 smore-more-1*)

lemma *and-chop-1*:
$(((lan\ f) \cap SMore) \cdot\ (lan\ g)) = (lan\ ((f\ \wedge_i\ more);g))$
using *and-chop lan-def* **by** *blast*

lemma *spower-chop-power*:
$(SPower\ (lan\ f)\ n) = (lan\ (power\text{-}chop\text{-}d\ f\ n))$

by (*induct n*, *simp add*: *sempty-empty-1*, *simp add*: *and-chop-1*)

lemma *sstar-spower*:
$\sigma \in SStar\ (lan\ f) \longleftrightarrow (\exists\ n.\ \sigma \in SPower\ (lan\ f)\ n)$
by (*simp add*: *sstar-def*)

lemma *sstar-chopstar*:
$\sigma \in (SStar\ (lan\ f)) \longleftrightarrow \sigma \in (lan\ (f^\star))$
using *chopstar-eqv-power-chop sstar-spower interval-lan spower-chop-power* **by** *blast*

lemma *chopstar-sstar-1*:
$(SStar\ (lan\ f)) = (lan\ (f^\star))$
using *sstar-chopstar lan-def* **by** *blast*

lemma *chopstar-seqv*:
$\sigma \in (lan\ (f^\star)) \longleftrightarrow \sigma \in (lan\ (empty\ \vee_i\ (f\ \wedge_i\ more);\ f^\star))$
using *ChopstarEqv valid-lan-eqv* **by** *blast*

lemma *chopstar-seqv-1*:
$(lan\ (f^\star)) = (lan\ (empty\ \vee_i\ (f\ \wedge_i\ more);\ f^\star))$
using *chopstar-seqv lan-def* **by** *blast*

lemma *srev-rev*:
$\sigma \in (SRev\ (lan\ f)) \longleftrightarrow \sigma \in (lan\ (f^r))$
by (*metis TimeReverseSem interval-lan interval-rev-rev-ident srev-elim*)

lemma *srev-rev-1*:
$(SRev\ (lan\ f)) = (lan\ (f^r))$
using *srev-rev lan-def* **by** *blast*

end

References

[1] A. Armstrong, G. Struth, and T. Weber. Kleene algebra. *Archive of Formal Proofs*, 2013. http://isa-afp.org/entries/Kleene_Algebra.shtml, Formal proof development.

[2] A. Cau. Interval temporal algebra, 2009. http://antonio-cau.co.uk/ITL/itl-atp/index.html.

[3] B. Moszkowski. A Hierarchical Completeness Proof for Propositional Interval Temporal Logic with Finite Time. *Journal of Applied Non-Classical Logics*, 14(1–2):55–104, 2004.

[4] B. Moszkowski. Compositional reasoning using intervals and time reversal. *Annals of Mathematics and Artificial Intelligence*, 71(1-3):175–250, 2014.

[5] B. C. Moszkowski. Imperative reasoning in interval temporal logic. Technical report, University of Newcastle upon Tyne, 1996.

[6] D. Smallwood. *ITL Monitor: Compositional Runtime Analysis with Interval Temporal Logic*. PhD thesis, De Montfort University, 2018.