Event: Start with the library "c-if".

Theorem: normal-not-legal-begin-label
\[
((\text{car} \ (\text{stmt})) = \begin{array}{l}
'\text{begin-mg} \\
\wedge \text{ok-mg-statement}(\text{stmt}, r-\text{cond-list}, name-alist, \text{proc-list})
\end{array}) \\
\rightarrow \ (('\text{normal} \in \text{when-labels}(\text{stmt})) = \text{f})
\]

Theorem: begin-translation-2
\[
(\text{car} \ (\text{stmt}) = \begin{array}{l}
'\text{begin-mg}
\end{array}) \\
\rightarrow \ (\text{translate}(\text{cinfo}, \text{cond-list}, \text{stmt}, \text{proc-list}) = \\
\text{add-code}(\text{translate}(\text{add-code}(\text{set-label-alist}(\text{translate}(\text{make-cinfo}(\text{code}(\text{cinfo}), \\
\text{append}(\text{make-label-alist}(\text{when-label-alist}(\text{cond-list}(\text{cinfo})), \\
1 + (1 + \text{label-cnt}(\text{cinfo}))), \\
\text{begin-body}(\text{stmt}), \\
\text{proc-list}), \\
\text{label-alist}(\text{cinfo})), \\
\text{list}(\text{list}(\begin{array}{l}
'\text{jump}, \\
1 + \text{label-cnt}(\text{cinfo})
\end{array}), \\
\text{list}(\begin{array}{l}
'\text{dl}, \\
\text{label-cnt}(\text{cinfo}), \\
\text{nil}, \\
'\text{(push-constant} \ (\text{nat} \ 2))\end{array}), \\
'\text{(pop-global c-c)})\), \\
\text{cond-list}, \\
\text{when-handler}(\text{stmt}), \\
\text{proc-list}), \\
\text{list}(\text{list}(\begin{array}{l}
'\text{dl}, 1 + \text{label-cnt}(\text{cinfo}), \text{nil}, \begin{array}{l}
'\text{(no-op})\end{array})
\end{array}))\))
\]

Theorem: begin-meaning-r-2
\[
(\text{car} \ (\text{stmt}) = \begin{array}{l}
'\text{begin-mg}
\end{array}) \\
\rightarrow \ (\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, n, sizes) = \\
\text{if} \ n \not\equiv 0 \ \text{then} \ \text{signal-system-error}(\text{mg-state}, \begin{array}{l}
'\text{timed-out}
\end{array}) \\
\text{elseif} \ \neg \ \text{normal}(\text{mg-state}) \ \text{then} \ \text{mg-state}
\]
elseif resources-inadequatep (stmt, proc-list, sizes) then signal-system-error (mg-state, 'resource-error)
elseif cc (mg-meaning-r (begin-body (stmt),
  proc-list,
  mg-state,
  n − 1,
  sizes)) ∈ when-labels (stmt)
then mg-meaning-r (when-handler (stmt),
  proc-list,
  set-condition (mg-meaning-r (begin-body (stmt),
    proc-list,
    mg-state,
    n − 1,
    sizes),
    'normal),
  n − 1,
  sizes)
else mg-meaning-r (begin-body (stmt),
  proc-list,
  mg-state,
  n − 1,
  sizes) endif

EVENT: Disable begin-meaning-r-2.

THEOREM: begin-body-doesnt-halt
((n ≠ 0)
 ∧ (car (stmt) = 'begin-mg)
 ∧ normal (mg-state)
 ∧ (¬ resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes))))
→ (mg-psw (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n − 1, sizes))
  = 'run)

THEOREM: begin-when-arm-doesnt-halt
((n ≠ 0)
 ∧ (car (stmt) = 'begin-mg)
 ∧ normal (mg-state)
 ∧ (cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n − 1, sizes))
  ∈ when-labels (stmt))
 ∧ (¬ resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes))))
→ (mg-psw (mg-meaning-r (when-handler (stmt),
  proc-list,
  mg-state ('normal,
mg-alist (mg-meaning-r (begin-body (stmt),
proc-list,
mg-state,
n − 1,
sizes)),
mg-psw (mg-meaning-r (begin-body (stmt),
proc-list,
mg-state,
n − 1,
sizes)),

n − 1,
sizes))

= 'run)

(prove-lemma begin-code-rewrite1 (rewrite)
 (implies
 (and (equal (car stmt) 'begin-mg)
  (ok-mg-statement stmt r-cond-list name-alist proc-list))
  (equal (append (code (translate cinfo t-cond-list stmt proc-list))
    code2)
    (append
     (code (translate (make-cinfo (code cinfo)
       (append (make-label-alist (when-labels stmt)
          (label-cnt cinfo))
       (label-alist cinfo))
       (add1 (add1 (label-cnt cinfo))))
       t-cond-list
       (begin-body stmt)
       proc-list))
   (cons
    (list 'jump (add1 (label-cnt cinfo))))
   (cons
    (cons 'dl
      (cons (label-cnt cinfo)
        '(nil (push-constant (nat 2))))))
   (cons
    '(pop-global c-c)
   (append
    (code
     (translate
      (nullify
       (set-label-alist

3
\[(\text{TRANSLATE})\n\quad (\text{MAKE-CINFO} \ (\text{CODE} \ \text{CINFO}))\n\quad (\text{APPEND} \ (\text{MAKE-LABEL-ALIST} \ (\text{WHEN-LABELS} \ \text{STMT}) \n\quad \quad (\text{LABEL-CNT} \ \text{CINFO}))) \n\quad (\text{LABEL-ALIST} \ \text{CINFO}))\n\quad (\text{ADD1} \ (\text{ADD1} \ (\text{LABEL-CNT} \ \text{CINFO})))))\n\quad (\text{T-COND-LIST})\n\quad (\text{BEGIN-BODY} \ \text{STMT}) \n\quad (\text{PROC-LIST})\n\quad (\text{LABEL-ALIST} \ \text{CINFO})))\n\quad (\text{T-COND-LIST})\n\quad (\text{WHEN-HANDLER} \ \text{STMT}) \n\quad (\text{PROC-LIST}))\n\quad (\text{CONS} \ (\text{CONS} \ 'DL \n\quad (\text{CONS} \ (\text{ADD1} \ (\text{LABEL-CNT} \ \text{CINFO}))) \n\quad '('\text{NIL} \ (\text{NO-OP}))))))\n\quad (\text{CODE2))))))))\n\quad ((\text{INSTRUCTIONS}) \n\quad \quad (\text{PROMOTE} \ (\text{DIVE} \ 1 \ 1 \ 1) \ (\text{REWRITE} \ \text{BEGIN-TRANSLATION-2}) \ \text{UP} \ \text{UP} \n\quad \quad (\text{S} \ \text{LEMMA}) \ (\text{DIVE} \ 1) \ (\text{REWRITE} \ \text{NEW-CODE-APPENDED-TO-OLD1}) \ (\text{S} \ \text{LEMMA}) \n\quad \quad \text{UP} \ \text{UP} \ (\text{S} \ \text{LEMMA}) \ X \ (\text{S} \ \text{LEMMA}))))\n\quad ((\text{THEOREM}): \ \text{begin-labels-subset-r-cond-list} \n\quad ((\text{car} \ (\text{stmt}) = '\text{begin-ng}) \n\quad \wedge \ \text{ok-mg-statement} \ (\text{stmt}, \ \text{r-cond-list}, \ \text{name-alist}, \ \text{proc-list})) \n\quad \rightarrow \ \text{cond-subsetp} \ (\text{when-labels} \ (\text{stmt}), \ \text{r-cond-list})\n\quad (\text{prove-lemma} \ \text{begin-body-hyps} \ (\text{rewrite}) \n\quad (\text{IMPLIES}) \n\quad \quad (\text{AND} \ (\text{NOT} \ (\text{ZERO} \ \text{OP}))) \n\quad \quad (\text{NOT} \ (\text{RESOURCES-INADEQUATEP} \ \text{STMT} \ \text{PROC-LIST}) \n\quad (\text{LIST} \ (\text{LENGTH} \ \text{TEMP-STK}) \n\quad \quad (\text{P-CTRL-STK-SIZE} \ \text{CTRL-STK}))))\n\quad (\text{EQUAL} \ (\text{CAR} \ \text{STMT}) '\text{BEGIN-MG}) \n\quad (\text{OK-MG-STATEMENT} \ \text{STMT} \ \text{R-COND-LIST} \ \text{NAME-ALIST} \ \text{PROC-LIST}) \n\quad (\text{OK-MG-DEF-PLISTP} \ \text{PROC-LIST}) \n\quad (\text{OK-TRANSLATION-PARAMETERS} \ \text{CINFO} \ \text{T-COND-LIST} \ \text{STMT} \ \text{PROC-LIST} \ \text{CODE2}) \n\quad (\text{OK-MG-STATEP} \ \text{MG-STATE} \ \text{R-COND-LIST}) \n\quad (\text{COND-SUBSETP} \ \text{R-COND-LIST} \ \text{T-COND-LIST}) \n\quad (\text{EQUAL} \ (\text{CODE} \ (\text{TRANSLATE-DEF-BODY} \ (\text{ASSOC} \ \text{SUBR} \ \text{PROC-LIST}) \n\quad \quad \text{PROC-LIST})) \n\quad (\text{APPEND} \ (\text{CODE} \ (\text{TRANSLATE} \ \text{CINFO} \ \text{T-COND-LIST} \ \text{STMT} \ \text{PROC-LIST})))\n\]
(CODE2))
  (USER-DEFINED-PROC SUBR PROC-LIST)
  (PLISTP TEMP-STK)
  (LISTP CTRL-STK)
  (MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE))
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)
  (NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
  (SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
  (NORMAL MG-STATE)
  (ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
  (NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N
  (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))))
  (AND (OK-MG-STATEMENT (BEGIN-BODY STMT) (APPEND (WHEN-LABELS STMT) R-COND-LIST)
    NAME-ALIST PROC-LIST))
  (OK-TRANSLATION-PARAMETERS
    (MAKE-CINFO (CODE CINFO)
    (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
      (LABEL-CNT CINFO))
    (LABEL-ALIST CINFO))
    (ADD1 (ADD1 (LABEL-CNT CINFO)))
    T-COND-LIST
    (BEGIN-BODY STMT)
    PROC-LIST
    (CONS
      (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
    (CONS
      (CONS 'DL
        (CONS (LABEL-CNT CINFO)
          '(NIL (PUSH-CONSTANT (NAT 2))))))
    (CONS
      '(POP-GLOBAL C-C)
    (APPEND
      (CODE
        (TRANSLATE
          (NULLIFY
            (SET-LABEL-ALIST
              (TRANSLATE
                (MAKE-CINFO (CODE CINFO)
                (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
                  (LABEL-CNT CINFO))
                (LABEL-ALIST CINFO))
                (ADD1 (ADD1 (LABEL-CNT CINFO))))))

5
T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)
  (LABEL-ALIST CINFO))))
T-COND-LIST
  (WHEN-HANDLER STMT)
  PROC-LIST))
(CONS (CONS 'DL
   (CONS (ADD1 (LABEL-CNT CINFO))
     '(NIL (NO-OP)))))
   CODE2))))))
(DK-MG-STATEP MG-STATE (APPEND (WHEN-LABELS STMT) R-COND-LIST))
(COND-SUBSETP (APPEND (WHEN-LABELS STMT) R-COND-LIST) T-COND-LIST)
(EQUAL
  (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
    PROC-LIST))
  (APPEND
    (CODE (TRANSLATE (MAKE-CINFO (CODE CINFO)
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
      T-COND-LIST
      (BEGIN-BODY STMT)
      PROC-LIST))
    (CONS
      (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
    (CONS
      (CONS 'DL
        (CONS (LABEL-CNT CINFO)
          '(NIL (PUSH-CONSTANT (NAT 2))))))
      (CONS
        '(POP-GLOBAL C-C)
      (APPEND
        (CODE
          (TRANSLATE
            (NULLIFY
              (SET-LABEL-ALIST
                (TRANSLATE
                  (MAKE-CINFO (CODE CINFO)
                    (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
                      (LABEL-CNT CINFO))
                    (LABEL-ALIST CINFO))
                    (ADD1 (ADD1 (LABEL-CNT CINFO))))))
 6
T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)
  (LABEL-ALIST CINFO)))
T-COND-LIST
  (WHEN-HANDLER STMT)
  PROC-LIST))
((CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO))
    '(NIL (NO-OP))))
CODE2))))))))
  (NOT (RESOURCE-ERRORP (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))))
  ((INSTRUCTIONS
    PROMOTE SPLIT (PROVE (ENABLE OK-MG-STATEMENT)) X SPLIT (DEMOTE 6) (DIVE 1)
    X-DUMB (DIVE 2 2) X (DIVE 1) (REWRITE BEGIN-CODE-REWRITE1) TOP PROVE
    (PROVE (ENABLE OK-TRANSLATION-PARAMETERS))
    (PROVE (ENABLE OK-TRANSLATION-PARAMETERS)) X (PROVE (ENABLE OK-MG-STATEP))
    (REWRITE COND-SUBSETP-APPEND) (REWRITE BEGIN-LABELS-SUBSET-R-COND-LIST)
    (DIVE 1) = (REWRITE BEGIN-CODE-REWRITE1) UP S S (DIVE 1)
    (REWRITE BEGIN-BODY-DOESNT-HALT) TOP S)))))
(prove-lemma begin-when-handler-hyps (rewrite)
  (IMPLIES
    (AND (NOT (ZEROP N))
      (NOT (RESOURCES-INADEQUATEP STMT PROC-LIST
      (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK))))))
  (EQUAL (CAR STMT) 'BEGIN-MG)
  (OK-MG-STATEMENT STMT R-COND-LIST NAME-ALIST PROC-LIST)
  (OK-MG-DEF-PLISTP PROC-LIST)
  (OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
  (OK-MG-STATEP MG-STATE R-COND-LIST)
  (COND-SUBSETP R-COND-LIST T-COND-LIST)
  (EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
    PROC-LIST))
  (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
CODE2))
  (USER-DEFINED-PROC SUBR PROC-LIST)

7
(PLISTP TEMP-STK)
(LISTP CTRL-STK)
(MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE))
(BINDINGS (TOP CTRL-STK))

TEMP-STK)

(NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(not (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(when-labels stmt))

(AND (OK-MG-STATEMENT (WHEN-HANDLER STMT) R-COND-LIST NAME-ALIST PROC-LIST)

(DK-TRANSLATION-PARAMETERS

(ADD-CODE

(SET-LABEL-ALIST

(TRANSLATE (MAKE-CINFO (CODE CINFO)

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)

(LABEL-CNT CINFO))

(LABEL-ALIST CINFO))

(ADD1 (LABEL-CNT CINFO)))))

T-COND-LIST

(BEGIN-BODY STMT)

PROC-LIST)

(LABEL-ALIST CINFO))

(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO))))

(CONS (CONS 'DL

(CONS (LABEL-CNT CINFO)

'(NIL (PUSH-CONSTANT (NAT 2))))

'((POP-GLOBAL C-C)))))

T-COND-LIST

(WHEN-HANDLER STMT)

PROC-LIST)

(CONS (CONS 'DL

(CONS (ADD1 (LABEL-CNT CINFO))

'(NIL (NO-OP))))))

CODE2))
(OK-MG-STATEP
  (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT))
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  'NORMAL)
  R-COND-LIST)
  (EQUAL
  (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
  PROC-LIST))
  (APPEND
  (CODE
  (TRANSLATE
  (ADD-CODE
  (SET-LABEL-ALIST
  (TRANSLATE (MAKE-CINFO (CODE CINFO))
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO))
  (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)
  (LABEL-ALIST CINFO))
  (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
  (CONS (CONS 'DL
  (CONS (LABEL-CNT CINFO)
  '(NIL (PUSH-CONSTANT (NAT 2))))
  '((POP-GLOBAL C-C)))))
  T-COND-LIST
  (WHEN-HANDLER STMT)
  PROC-LIST))
  (CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO))
  '(NIL (NO-OP))))
  CODE2))
  (MG-VARS-LIST-OK-IN-P-STATE
  (MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT))
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  'NORMAL))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(NO-P-ALIASING
(BINDINGS (TOP CTRL-STK))
(MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL))
(SIGNATURES-MATCH
(MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL))
NAME-ALIST)
(NORMAL (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL))
(ALL-CARS-UNIQUE
(MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL))
(NOT
RESOURCE-ERRORP
(MG-MEANING-R
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
((INSTRUCTIONS PROMOTE
  (DIVE 2 2 2 2 2 2 2 2 2)
  PUSH TOP
  (= (MG-MEANING-R (BEGIN-BODY STMT)
      PROC-LIST MG-STATE
      (SUB1 N)
      (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK)))
  (MG-MEANING (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)))
  0)
SPLIT
(REWRITE OK-BEGIN-STATEMENT)
X
(S LEMMAS)
SPLIT
(DEMOTE 6)
(DIVE 1)
X
(DIVE 2 1 1 1)
(REWRITE BEGIN-TRANSLATION-2)
UP
(S LEMMAS)
TOP
(ENABLE ADD-CODE)
(S LEMMAS)
S
(PROVE (ENABLE OK-MG-STATEMENT))
S X
(REWRITE MG-MEANING-PRESERVES-MG-ALISTP
  (**R-COND-LIST (APPEND (WHEN-LABELS STMT)
        R-COND-LIST))))
(REWRITE BEGIN-BODY-HYPS)
(REWRITE BEGIN-BODY-HYPS)
(DIVE 1)
=
(DIVE 1 1)
(REWRITE BEGIN-TRANSLATION-2)
TOP PROVE S
(REWRITE SIGNATURES-MATCH-PRESERVES-MG-VARS-LIST-OK
  (**X (MG-ALIST MG-STATE))))
(REWRITE MG-MEANING-PRESERVES-SIGNATURES-MATCH)
(REWRITE OK-MG-STATEP-ALIST-PLISTP)
S
(REWRITE SIGNATURES-MATCH-PRESERVES-NG-P-ALIASING
  (($ALIST1 (MG-ALIST MG-STATE))))
(REWRITE MG-MEANING-PRESERVES-SIGNATURES-MATCH)
(REWRITE OK-MG-STATEP-ALIST-PLISTP)
S
(REWRITE SIGNATURES-MATCH-REORDER
  (($ALIST1 (MG-ALIST MG-STATE))))
(REWRITE OK-MG-STATEP-ALIST-PLISTP)
(REWRITE MG-MEANING-PRESERVES-SIGNATURES-MATCH)
(REWRITE OK-MG-STATEP-ALIST-PLISTP)
S S
(REWRITE SIGNATURES-MATCH-PRESERVES-UNIQUENESS-OF-CARS
  (($X (MG-ALIST MG-STATE))))
(REWRITE MG-MEANING-PRESERVES-SIGNATURES-MATCH)
(REWRITE OK-MG-STATEP-ALIST-PLISTP)
(DIVE 1)
(REWRITE MG-MEANING-EQUIVALENCE)
TOP S S
(DIVE 1)
(REWRITE BEGIN-BODY-DOESNT-HALT)
TOP S S
(DIVE 1)
(REWRITE BEGIN-WHEN-ARM-DOESNT-HALT)
TOP S)))

Theorem: begin-find-labelp-lemma1
((car (stmt) = 'begin-mg)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-translation-parameters (cinfo, t-cond-list, stmt, proc-list, code2))
→ (¬ find-labelp (1 + label-cnt (cinfo), code (cinfo)))

Theorem: begin-find-labelp-lemma2
((car (stmt) = 'begin-mg)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-translation-parameters (cinfo, t-cond-list, stmt, proc-list, code2))
→ (¬ find-labelp (label-cnt (cinfo), code (cinfo)))

Theorem: append-doesnt-affect-fetch-label
(x \notin lst)
→ (fetch-label (x, append (make-label-alist (lst, label), label-alist))
  = fetch-label (x, label-alist))

Event: Enable nullify.
(prove-lemma begin-state2-normal-body-step1 (rewrite)
  (IMPLIES
    (AND (NOT (ZEROP N))
      (NOT (RESOURCES-INADEQUATEP STMT PROC-LIST)
        (LIST (LENGTH TEMP-STK)
          (P-CTRL-STK-SIZE CTRL-STK))))
    (EQUAL (CAR STMT) 'BEGIN-MG)
    (OK-MG-STATE MG-STATE R-COND-LIST)
    (OK-MG-DEF-PLISTP PROC-LIST)
    (OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
    (OK-MG-STATEP MG-STATE R-COND-LIST)
    (COND-SUBSETP R-COND-LIST T-COND-LIST)
    (EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST) PROCLIST))
      (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
        CODE2))
    (USER-DEFINED-PROC SUBR PROC-LIST)
    (PLISTP TEMP-STK)
    (LISTP CTRL-STK)
    (MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE)
      (BINDINGS (TOP CTRL-STK))
      TEMP-STK)
    (NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
    (SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
    (NORMAL MG-STATE)
    (ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
    (NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
      (LIST (LENGTH TEMP-STK)
        (P-CTRL-STK-SIZE CTRL-STK))))
    (NORMAL (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
      (LIST (LENGTH TEMP-STK)
        (P-CTRL-STK-SIZE CTRL-STK))))
    (equal
      (p-step
        (P-STATE ;; state2
          (TAG 'PC
            (CONS SUBR
              (IF
                (NORMAL (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE
                  (SUB1 N)
                  (LIST (LENGTH TEMP-STK)
                    (P-CTRL-STK-SIZE CTRL-STK))))
                (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
                  (LIST (LENGTH TEMP-STK)
                    (P-CTRL-STK-SIZE CTRL-STK))))))
      (p-step
        (P-STATE
          (TAG 'PC
            (CONS SUBR
              (IF
                (NORMAL (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE
                  (SUB1 N)
                  (LIST (LENGTH TEMP-STK)
                    (P-CTRL-STK-SIZE CTRL-STK))))
              (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
                (LIST (LENGTH TEMP-STK)
                  (P-CTRL-STK-SIZE CTRL-STK))))))
  )
(P-CTRL-STK-SIZE CTRL-STK))

(LENGTH
(CODE
  (TRANSLATE (MAKE-CINFO (CODE CINFO))
    (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
      (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
    (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))

(FIND-LABEL
  (FETCH-LABEL
    (CC (MG-MEANING-R (BEGIN-BODY STMT)
      PROC-LIST MG-STATE
      (SUB1 N)
      (LIST (LENGTH TEMP-STK)
        (P-CTRL-STK-SIZE CTRL-STK)))))
  (LABEL-ALIST
    (TRANSLATE (MAKE-CINFO (CODE CINFO))
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
    (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))))

(APPEND
  (CODE
    (TRANSLATE (MAKE-CINFO (CODE CINFO))
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
    (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))))

(CONS
  (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
  (CONS
    (CONS 'DL
      (CONS (LABEL-CNT CINFO)
        '(NIL (PUSH-CONSTANT (NAT 2))))))
    (CONS
'(POP-GLOBAL C-C)
(APPEND
 (CODE
  (TRANSLATE
   (NULLIFY
    (SET-LABEL-ALIST
     (TRANSLATE
      (MAKE-CINFO (CODE CINFO)
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
        (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
     T-COND-LIST
     (BEGIN-BODY STMT)
     PROC-LIST)
     (LABEL-ALIST CINFO)))))
  T-COND-LIST
  (WHEN-HANDLER STMT)
  PROC-LIST))
(CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO))
    '(NIL (NO-OP))))))
CTRL-STK
(MAP-DOWN-VALUES
(MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK)))))
(BINDINGS (TOP CTRL-STK))
TEMP-STK
  (TRANSLATE-PROC-LIST PROC-LIST)
  (LIST
   (LIST 'C-C
     (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE
       (SUB1 N)
       (LIST (LENGTH TEMP-STK)
         (P-CTRL-STK-SIZE CTRL-STK)))))
     T-COND-LIST))))
   (MG-MAX-CTRL-STK-SIZE)
   (MG-MAX-TEMP-STK-SIZE)
   (MG-WORD-SIZE)
'RUN))
   (P-STATE
(TAG 'PC
   (CONS SUBR
   (PLUS
   (LENGTH
   (CODE (TRANSLATE (MAKE-CINFO (CODE CINFO))
   (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
   (LABEL-CNT CINFO))
   (LABEL-ALIST CINFO))
   (ADD1 (ADD1 (LABEL-CNT CINFO))))
   T-COND-LIST
   (BEGIN-BODY STMT)
   PROC-LIST))))
   (ADD1
   (ADD1
   (ADD1
   (LENGTH
   (CODE
   (TRANSLATE
   (MAKE-CINFO NIL
   (LABEL-ALIST CINFO))
   (LABEL-CNT
   (TRANSLATE (MAKE-CINFO (CODE CINFO))
   (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
   (LABEL-CNT CINFO))
   (LABEL-ALIST CINFO))
   (ADD1 (ADD1 (LABEL-CNT CINFO))))
   T-COND-LIST
   (BEGIN-BODY STMT)
   PROC-LIST))))
   T-COND-LIST
   (WHEN-HANDLER STMT)
   PROC-LIST))))
CTRL-STK
   (MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
   PROC-LIST MG-STATE
   (SUB1 N)
   (LIST (LENGTH TEMP-STK)
   (P-CTRL-STK-SIZE CTRL-STK)))))
   (BINDINGS (TOP CTRL-STK))
   TEMP-STK)
   (TRANSLATE-PROC-LIST PROC-LIST)
   (LIST

   16
\begin{align*}
\text{Theorem: } & \text{begin-state2-normal-body-step2-equals-final} \\
& ((n \not\equiv 0) \land \neg \text{resources-inadequatep} (stmt, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))) \\
& \land (\text{car (stmt)} = '\text{begin-mg}) \\
& \land \text{ok-mg-statement} (stmt, r-cond-list, name-alist, proc-list) \\
& \land \text{ok-mg-def-plistp} (proc-list) \\
& \land \text{ok-translation-parameters} (cinfo, t-cond-list, stmt, proc-list, code2) \\
& \land \text{ok-mg-statep} (mg-state, r-cond-list) \\
& \land \text{cond-subsetp} (r-cond-list, t-cond-list) \\
& \land \text{(code (translate-def-body (assoc (subr, proc-list), proc-list)))} \\
& \quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))} \\
& \land \text{user-defined-procp} (subr, proc-list) \\
& \land \text{plistp} (temp-stk) \\
& \land \text{listp} (ctrl-stk) \\
& \land \text{mg-vars-list-ok-in-p-state} (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk) \\
& \land \text{no-p-aliasing} (bindings (top (ctrl-stk)), mg-alist (mg-state))
\end{align*}
\[\begin{align*}
&\land \text{signatures-match (mg-alist (mg-state), name-alist)} \\
&\land \text{normal (mg-state)} \\
&\land \text{all-cars-unique (mg-alist (mg-state))} \\
&\land (\neg \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}) \\
&\land \text{normal (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))} \\
&\rightarrow (\text{p-step (p-state (tag (p\text{c}, cons (subr, length (code (translate (make-cinfo (code (cinfo), append (make-label-alist (when-labels (stmt, label-cnt (cinfo)), label-alist (cinfo)), 1 + (1 + label-cnt (cinfo))), t-cond-list, begin-body (stmt, proc-list))))) \\
+ (1 + (1 + (1 + length (code (translate (make-cinfo (nil, label-alist (cinfo), label-cnt (translate (make-cinfo (code (t-cond-list, when-handler (stmt, proc-list))))), t-cond-list, when-handler (stmt, proc-list))))))))))}) \\
&\land \text{ctrl-stk}, \\
&\text{map-down-values (mg-alist (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}, \\
\end{align*}\]
bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list ('c-c, mg-cond-to-p-nat (cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),
t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE, 'run))
= p-state (tag ('pc, cons (subr, if normal (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))
then length (code (translate (cinfo, t-cond-list, stmt, proc-list))))
else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),
label-alist (translate (cinfo, t-cond-list, stmt, proc-list))),
append (code (translate (cinfo, t-cond-list, stmt, proc-list))),

ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (stmt,
(prove-lemma begin-nonnormal-nonwhen-body-state2-equals-final (rewrite)
  (implies
    (not (zerop n))
    (not (resources-inadequatep stmt proc-list
      (list (length temp-stk)
        (p-ctrl-stk-size ctrl-stk)))
      (equal (car stmt) 'begin-mg)
      (ok-mg-statement stmt r-cond-list name-alist proc-list)
      (ok-mg-def-plistp proc-list)
      (ok-translation-parameters cinfo t-cond-list stmt proc-list code2)
      (ok-mg-statep mg-state r-cond-list)
      (cond-subsetp r-cond-list t-cond-list)
      (equal (code (translate-def-body (assoc subr proc-list)
        proc-list))
        (append (code (translate cinfo t-cond-list stmt proc-list))
        code2))
      (user-defined-procp subr proc-list)
      (plistp temp-stk)
      (listp ctrl-stk)
      (mg-vars-list-ok-in-p-state (mg-alist mg-state)
        (bindings (top ctrl-stk)))
```
TEMP-STK
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(not (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))))
(not (member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))
(when-labels stmt)))
(equal (P-STATE
(TAG 'PC
(CONS SUBR
(IF
(NORMAL (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(LENGTH
(CODE
(TRANSLATE (MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
(LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST))))
(FIND-LABEL
(FETCH-LABEL
(CC (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(LABEL-ALIST
(TRANSLATE (MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
(LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO)))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)))
(APPEND
(CODE
(TRANSLATE (MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO)))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)))
(CONS
(LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS
(CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2))))))
(CONS
'(POP-GLOBAL C-C)
(APPEND
(CODE
(TRANSLATE
(NULLIFY
(SET-LABEL-ALIST
(TRANSLATE
(MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO)))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO)))))
T-COND-LIST
(WHEN-HANDLER STMT)
PROC-LIST))
(CONS (CONS 'DL
 (CONS (ADD1 (LABEL-CNT CINFO))

'(NIL (NO-OP)))
    CODE2))))))))))
CTRL-STK
    (MAP-DOWN-VALUES
(MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
    (TRANSLATE-PROC-LIST PROC-LIST)
    (LIST
(LIST 'C-C
    (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)
T-COND-LIST)))
    (MG-MAX-CTRL-STK-SIZE)
    (MG-MAX-TEMP-STK-SIZE)
    (MG-WORD-SIZE)
    'RUN)
    (P-STATE
    (TAG 'PC
(CONS SUBR
    (IF
      (NORMAL (MG-MEANING-R STMT PROC-LIST MG-STATE N
      (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK)
      (LENGTH (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))))
      (FIND-LABEL
      (FETCH-LABEL (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
      (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK)
      (LABEL-ALIST (TRANSLATE CINFO T-COND-LIST STMT
      PROC-LIST))))
      (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
      CODE2)))))
CTRL-STK
    (MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R STMT PROC-LIST MG-STATE N
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))

23
Theorem: append-make-label-alist-fetch-label
\[(x \in lst) \rightarrow (cdr (assoc (x, append (make-label-alist (lst, label), label-alist)))) = label\]
(prove-lemma begin-when-signalled-state2-step1 (rewrite)
  (implies
   (and (not (zerop n))
        (not (resources-inadequatep stmt proc-list)
            (list (length temp-stk)
                  (p-ctrl-stk-size ctrl-stk))))
   (equal (car stmt) 'begin-mg)
   (ok-mg-statement stmt r-cond-list name-alist proc-list)
   (ok-mg-def-plistp proc-list)
   (ok-translation-parameters cinfo t-cond-list stmt proc-list code2)
   (ok-mg-state mg-state r-cond-list)
   (cond-subsetp r-cond-list t-cond-list)
   (equal (code (translate-def-body (assoc subr proc-list)
              proc-list))
         (append (code (translate cinfo t-cond-list stmt proc-list))
                code2))
   (user-defined-procp subr proc-list)
   (plistp temp-stk)
   (listp ctrl-stk)
   (mg-vars-list-ok-in-p-state (mg-alist mg-state)
                                (bindings (top ctrl-stk))
                                temp-stk)
   (no-p-aliasing (bindings (top ctrl-stk)) (mg-alist mg-state))
   (signatures-match (mg-alist mg-state) name-alist)
   (normal mg-state)
   (all-cars-unique (mg-alist mg-state))
   (not (resource-errorp (mg-meaning-r stmt proc-list mg-state n)
                         (list (length temp-stk)
                               (p-ctrl-stk-size ctrl-stk)))))))
  (not (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
                       (list (length temp-stk)
                             (p-ctrl-stk-size ctrl-stk))))))
(member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
                       (list (length temp-stk)
                             (p-ctrl-stk-size ctrl-stk))))
        (when-labels stmt)))
  (equal
   (p-step
      (p-state ;; state2
        (tag 'pc
        (cons subr
        (if
(NORMAL (MG-MEANING-R (BEGIN-BODY STMT))
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
(LENGTH
  (CODE
    (TRANSLATE (MAKE-CINFO (CODE CINFO))
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
        (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
    T-COND-LIST
    (BEGIN-BODY STMT)
    PROC-LIST))
  (FIND-LABEL
    (FETCH-LABEL
      (CC (MG-MEANING-R (BEGIN-BODY STMT))
        PROC-LIST MG-STATE
        (SUB1 N)
        (LIST (LENGTH TEMP-STK)
          (P-CTRL-STK-SIZE CTRL-STK))))
      (LABEL-ALIST
        (TRANSLATE (MAKE-CINFO (CODE CINFO))
          (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
            (LABEL-CNT CINFO))
            (LABEL-ALIST CINFO))
          (ADD1 (ADD1 (LABEL-CNT CINFO))))
        T-COND-LIST
        (BEGIN-BODY STMT)
        PROC-LIST))
      (APPEND
        (CODE
          (TRANSLATE (MAKE-CINFO (CODE CINFO))
            (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
              (LABEL-CNT CINFO))
              (LABEL-ALIST CINFO))
            (ADD1 (ADD1 (LABEL-CNT CINFO))))
          T-COND-LIST
          (BEGIN-BODY STMT)
          PROC-LIST))
        (CONS
          (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
          (CONS
            (LIST ...)
(CONS 'DL
(CONS (LABEL-CNT CINFO)
  '(NIL (PUSH-CONSTANT (NAT 2)))))
(CONS
  '(POP-GLOBAL C-C)
  (APPEND
   CODE
   (TRANSLATE
    (NULLIFY
     (SET-LABEL-ALIST
      (TRANSLATE
       (MAKE-CINFO (CODE CINFO)
       (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
        (LABEL-ALIST CINFO)))
       (ADD1 (LABEL-CNT CINFO))))
      T-COND-LIST
      (BEGIN-BODY STMT)
      PROC-LIST)
      (LABEL-ALIST CINFO)))
      T-COND-LIST
      (WHEN-HANDLER STMT)
      PROC-LIST))
    (CONS (CONS 'DL
      (CONS (ADD1 (LABEL-CNT CINFO))
        '(NIL (NO-OP)))))
      CTRL-STK
      (MAP-DOWN-VALUES
       (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE)
       (SUB1 N)
       (LIST (LENGTH TEMP-STK)
         (P-CTRL-STK-SIZE CTRL-STK)))
       (BINDINGS (TOP CTRL-STK))
       TEMP-STK)
       (TRANSLATE-PROC-LIST PROC-LIST)
       (LIST
       (LIST 'C-C
         (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE)
         (SUB1 N)
         (LIST (LENGTH TEMP-STK)
         (P-CTRL-STK-SIZE CTRL-STK))))))
T-COND-LIST)))
  (MG-MAX-CTRL-STK-SIZE)
  (MG-MAX-TEMP-STK-SIZE)
  (MG-WORD-SIZE)
  'RUN))
(P-STATE
(TAG 'PC
(CONS SUBR
  (PLUS
  (LENGTH
  (CODE
  (TRANSLATE (MAKE-CINFO (CODE CINFO)
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO))
  (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO)))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))
  2)))
CTRL-STK
(PUSH
  '(NAT 2)
  (MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK))
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
  (LIST 'C-C
  (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))
  T-COND-LIST)))))
  (MG-MAX-CTRL-STK-SIZE)
  (MG-MAX-TEMP-STK-SIZE)
  (MG-WORD-SIZE)
  'RUN))
  ((INSTRUCTIONS

28

;; The result of the (pop-global c-c) instruction should equal state3

;; The result of the (pop-global c-c) instruction should equal state3

(prove-lemma begin-when-signalled-state2-step2 (rewrite)
   (IMPLIES
      (AND (NOT (ZEROP N))
           (NOT (RESOURCES-INADEQUATEP STMT PROC-LIST)
                (LIST (LENGTH TEMP-STK)
                       (P-CTRL-STK-SIZE CTRL-STK))))
       (EQUAL (CAR STMT) 'BEGIN-MG)
       (OK-MG-STATEMENT STMT R-COND-LIST NAME-ALIST PROC-LIST)
       (OK-MG-DEF-PLISTP PROC-LIST)
       (OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
       (OK-MG-STATEP MG-STATE R-COND-LIST)
       (COND-SUBSETP R-COND-LIST T-COND-LIST)
       (EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST) PROC-LIST))
                (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST)) CODE2))
       (USER-DEFINED-PROC SUBR PROC-LIST)
       (PLISTP TEMP-STK)
       (LISTP CTRL-STK)
       (MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE)))

29
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N
LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(not (NORMAL (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))))
(member (CC (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))))
(when-labels stmt))
(equal
(p-step
  (P-STATE
   (TAG 'PC
   (CONS SUBR
   (PLUS
   (LENGTH
   (CODE
   (TRANSLATE (MAKE-CINFO (CODE CINFO)
   (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
   (LABEL-CNT CINFO))
   (LABEL-ALIST CINFO))
   (ADD1 (ADD1 (LABEL-CNT CINFO)))))
   (T-COND-LIST
   (BEGIN-BODY STMT)
   PROC-LIST)))))
  2)))
CTRL-STK
(PUSH
  '(NAT 2)
  (MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK))
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
  (LIST 'C-C
  (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))
  T-COND-LIST)))
 (MG-MAX-CTRL-STK-SIZE)
 (MG-MAX-TEMP-STK-SIZE)
 (MG-WORD-SIZE)
'RUN))
 (MAP-DOWN ;; state3
(Set-condition (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
 'NORMAL)
PROC-LIST CTRL-STK TEMP-STK
(TAG 'PC
  (CONS SUBR
  (LENGTH
  (CODE
  (ADD-CODE
  (SET-LABEL-ALIST
  (TRANSLATE (MAKE-CINFO (CODE CINFO)
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO)))
  (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)
 (LABEL-ALIST CINFO))
 (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
  (CONS (CONS 'DL
  (CONS (LABEL-CNT CINFO)
  '(NAT (PUSH-CONSTANT (NAT 2))))
  '((POP-GLOBAL C-C))))))
  T-COND-LIST)))
 (INSTRUCTIONS
  PROMOTE (DIVE 1) X (S LEMMAS) (DIVE 1 1 2) (REWRITE TRANSLATE-DEF-BODY-REWRITE)
  (REWRITE BEGIN-CODE-REWRITE1) UP (REWRITE GET-LENGTH-PLUS) X X UP X UP X
  31
(prove-lemma begin-when-nonnormal-state4-equals-final (rewrite)
  (implies
    (and (not (zerop n))
      (not (resources-inadequatep stmt proc-list)
        (list (length temp-stk)
          (p-ctrl-stk-size ctrl-stk))))
    (equal (car stmt) 'begin-mg)
    (ok-mg-statement stmt r-cond-list name-alist proc-list)
    (ok-mg-def-plistp proc-list)
    (ok-translation-parameters cinfo t-cond-list stmt proc-list code2)
    (ok-mg-statep mg-state r-cond-list)
    (cond-subsetp r-cond-list t-cond-list)
    (equal (code (translate-def-body (assoc subr proc-list) proc-list))
      (append (code (translate cinfo t-cond-list stmt proc-list))
        code2))
    (user-defined-proc subr proc-list)
    (plistp temp-stk)
    (listp ctrl-stk)
    (mg-vars-list-ok-in-p-state (mg-alist mg-state)
      (bindings (top ctrl-stk))
      temp-stk))
  (not-p-aliasing (bindings (top ctrl-stk)) (mg-alist mg-state))
  (signatures-match (mg-alist mg-state) name-alist)
  (normal mg-state)
  (all-cars-unique (mg-alist mg-state))
  (not (resource-errorp (mg-meaning-r stmt proc-list proc-list mg-state n)
    (list (length temp-stk)
      (p-ctrl-stk-size ctrl-stk))))
  (not (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
    (list (length temp-stk)
      (p-ctrl-stk-size ctrl-stk))))
  (member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
      (list (length temp-stk)
        (p-ctrl-stk-size ctrl-stk)))
    (when-labels stmt))
  (not (normal
    (mg-meaning-r (when-handler stmt) proc-list
      (set-condition (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
      (list (length temp-stk)
        (p-ctrl-stk-size ctrl-stk))))))
  (not (normal
    (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
    (list (length temp-stk)
      (p-ctrl-stk-size ctrl-stk))))
  (member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
      (list (length temp-stk)
        (p-ctrl-stk-size ctrl-stk)))
    (when-labels stmt))
  (not (normal
    (mg-meaning-r (when-handler stmt) proc-list
      (set-condition (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n))
      (list (length temp-stk)
        (p-ctrl-stk-size ctrl-stk))))))
(LIST (LENGTH TEMP-STK)  
(P-CTRL-STK-SIZE CTRL-STK))

'normal)

(sub1 n) (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))))

(equal

(P-STATE

(TAG 'PC

(CONS SUBR

(IF

NORMAL

(MG-MEANING-R

(WHEN-HANDLER STMT)

PROC-LIST

(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)

PROC-LIST MG-STATE

(SUB1 N)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK)))

'NORMAL)

(SUB1 N)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(LENGTH

(CODE

(TRANSLATE

(ADD-CODE

(SET-LABEL-ALIST

(TRANSLATE

(MAKE-CINFO (CODE CINFO)

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)

(LABEL-CNT CINFO))

(LABEL-ALIST CINFO)

(ADD1 (ADD1 (LABEL-CNT CINFO))))

T-COND-LIST

(BEGIN-BODY STMT)

PROC-LIST)

(LABEL-ALIST CINFO))

(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO))))

(CONS (CONS 'DL

(CONS (LABEL-CNT CINFO)

'(NIL (PUSH-CONSTANT (NAT 2)))))

'((POP-GLOBAL C-C)))))

T-COND-LIST

(WHEN-HANDLER STMT)
(MAKE-CINFO (CODE CINFO))

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO))
  (LABEL-ALIST CINFO)))

(ADD1 (ADD1 (LABEL-CNT CINFO))))

T-COND-LIST

(BEGIN-BODY STMT)

PROC-LIST)

(LABEL-ALIST CINFO))

(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))

(CONS (CONS 'DL
  (CONS (LABEL-CNT CINFO)
    '(NIL (PUSH-CONSTANT (NAT 2))))))

  '((POP-GLOBAL C-C)))))

T-COND-LIST

(WHEN-HANDLER STMT)

PROC-LIST))

  (CONS (CONS 'DL
    (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))

  CODE2))))))))

CTRL-STK

(MAP-DOWN-VALUES

(MG-ALIST
  (MG-MEANING-R
    (WHEN-HANDLER STMT)

    PROC-LIST
    (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)

    PROC-LIST MG-STATE
    (SUB1 N)

    (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK)))))

  'NORMAL)

  (SUB1 N)

  (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK)))))

  (BINDINGS (TOP CTRL-STK))

  TEMP-STK)

TRANSLATE-PROC-LIST PROC-LIST)

LIST

  (LIST 'C-C

  (MG-COND-TO-P-NAT
  (CC
  (MG-MEANING-R

35
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))
'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))
T-COND-LIST))
  (MG-MAX-CTRL-STK-SIZE)
  (MG-MAX-TEMP-STK-SIZE)
  (MG-WORD-SIZE)
'RUN)
  (P-STATE ;; final
  (TAG 'PC
    (CONS SUBR
      (IF
        (NORMAL (MG-MEANING-R STMT PROC-LIST MG-STATE N
          (LIST (LENGTH TEMP-STK)
            (P-CTRL-STK-SIZE CTRL-STK))))
        (LENGTH (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST)))
        (FIND-LABEL
          (FETCH-LABEL (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
            (LIST (LENGTH TEMP-STK)
              (P-CTRL-STK-SIZE CTRL-STK))))
            (LABEL-ALIST (TRANSLATE CINFO T-COND-LIST STMT
              PROC-LIST)))))
        (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
          CODE2)))))
CTRL-STK
(MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R STMT PROC-LIST MG-STATE N
  (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)
(TRANSLATE-PROC-LIST PROC-LIST)
LIST
(LIST 'C-C
  (MG-COND-TO-P-NAT (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
    (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK))))
  36
T-COND-LIST))
   (MG-MAX-CTRL-STK-SIZE)
   (MG-MAX-TEMP-STK-SIZE)
   (MG-WORD-SIZE)
'REUN))
(INSTRUCTIONS
   PROMOTE S
   (= (MG-MEANING-R STMT PROC-LIST MG-STATE N
   (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))
   (MG-MEANING-R
      (WHEN-HANDLER STMT) PROC-LIST
      (MG-STATE 'NORMAL
      (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N)
       (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK))
      (MG-PSW (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N)
       (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK))
       (SUB1 N) (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))) 0)
   S (S LEMMAS) (DIVE 1 2 2 1) (= * F 0) UP S TOP (DIVE 2 2 2 1) (= * F 0) UP S TOP (S LEMMAS)
   (DEMOTE 21) S (S LEMMAS) (DEMOTE 21) S (S LEMMAS) (DIVE 1) (REWRITE BEGIN-MEANING-R-2) TOP

(prove-lemma begin-when-normal-step-state4-equals-final (rewrite)
   (IMPLIES
      (AND (NOT (ZEROP N))
      (NOT (RESOURCES-INADEQUATEP STMT PROC-LIST
      (LIST (LENGTH TEMP-STK)
       (P-CTRL-STK-SIZE CTRL-STK))))
      (EQUAL (CAR STMT) 'BEGIN-MG)
      (OK-MG-STATEMENT STMT R-COND-LIST NAME-ALIST PROC-LIST)
      (OK-MG-DEF-PLISTP PROC-LIST)
      (OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
      (OK-MG-STATEP MG-STATE R-COND-LIST)
      (COND-SUBSETP R-COND-LIST T-COND-LIST)
      (EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
      PROC-LIST))
      (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
      CODE2))
      (USER-DEFINED-PROC SUBR PROC-LIST)
      (PLISTP TEMP-STK)
      (LISTP CTRL-STK)
      (MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE)
      (BINDINGS (TOP CTRL-STK))
      TEMP-STK)

37
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
(not (normal (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
(member (cc (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
(when-labels stmt))
(normal
  (mg-meaning-r (when-handler stmt) proc-list
  (set-condition (mg-meaning-r (begin-body stmt) proc-list mg-state (sub1 n)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
'normal)
(sub1 n) (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK))))
(equal
(p-step
  (P-STATE ;; state4
    (TAG 'PC
  (CONS SUBR
    (IF
  (NORMAL
    (MG-MEANING-R
    (WHEN-HANDLER STMT)
    PROC-LIST
    (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK))))
'NORMAL)
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  (LENGTH
  (CODE
  (TRANSLATE
  (ADD-CODE

38
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO))
(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS (CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2)))))
'((POP-GLOBAL C-C)))))
T-COND-LIST
(WHEN-HANDLER STMT)
PROC-LIST))
(APPEND
(CODE
(TRANSLATE
(ADD-CODE
(SET-LABEL-ALIST
(TRANSLATE
(MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO))
(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS (CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2)))))
'((POP-GLOBAL C-C)))))
T-COND-LIST
(WHEN-HANDLER STMT)
PROC-LIST))
(CONS (CONS 'DL
(CONS (ADD1 (LABEL-CNT CINFO))
'(NIL (NO-OP))))
CODE2))))))
CTRL-STK
(MAP-DOWN-VALUES
(MG-ALIST
(MG-MEANING-R
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
  (TRANSLATE-PROC-LIST PROC-LIST)
  (LIST
  (LIST 'C-C
    (MG-COND-TO-P-NAT
      (CC
        (MG-MEANING-R
          (WHEN-HANDLER STMT)
          PROC-LIST
          (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
            PROC-LIST MG-STATE
            (SUB1 N)
            (LIST (LENGTH TEMP-STK)
            (P-CTRL-STK-SIZE CTRL-STK)))
            'NORMAL)
        (SUB1 N)
        (LIST (LENGTH TEMP-STK)
          (P-CTRL-STK-SIZE CTRL-STK)))
        T-COND-LIST))))
      (MG-MAX-CTRL-STK-SIZE)
      (MG-MAX-TEMP-STK-SIZE)
      (MG-WORD-SIZE)
      'RUN))
(P-STATE ;; final
  (TAG 'PC
    (CONS SUBR
      (IF
        (NORMAL (MG-MEANING-R STMT PROC-LIST MG-STATE N
          (LIST (LENGTH TEMP-STK)
          (P-CTRL-STK-SIZE CTRL-STK)))
          (LENGTH (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))))
        (FIND-LABEL
          (FETCH-LABEL (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N

41
(LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))
(LABEL-ALIST (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
(APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST)) CODE2))))

CTRL-STK
(MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST (LIST 'C-C
(MG-COND-TO-P-NAT (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))))
T-COND-LIST))
(MG-MAX-CTRL-STK-SIZE)
(MG-MAX-TEMP-STK-SIZE)
(MG-WORD-SIZE)
'RUN)))

((INSTRUCTIONS
  PROMOTE (DIVE 1) X (S LEMMAS) (DIVE 2 1 2 1) (= * T 0) UP S UP UP UP UP (DIVE 1 1
  (= * T 0) UP S UP (DIVE 2) (REWRITE TRANSLATE-DEF-BODY-REWRITE)
  (REWRITE BEGIN-CODE-REWRITE1) UP (DIVE 1 1) (REWRITE NEW-CODE-APPENDED-TO-OLD1) UP UP
  (ENABLE LENGTH-CONS) (S LEMMAS) (REWRITE GET-LENGTH-PLUS) X X (REWRITE GET-LENGTH-CAR)
  S UP X UP X (DIVE 1) X UP S X (S LEMMAS) TOP S
  (= (MG-MEANING-R STMT PROC-LIST MG-STATE N
  (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))))
  (MG-MEANING-R

  (WHEN-HANDLER STMT) PROC-LIST
  (MG-STATE 'NORMAL
  (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N))
  (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))))
  'RUN)
  (SUB1 N)
  (LIST (LENGTH TEMP-STK) (P-CTRL-STK-SIZE CTRL-STK)))) 0)
  S (DIVE 2 2 2 1) (= * T 0) TOP S (DIVE 2 2 2 1 1) (REWRITE BEGIN-TRANSLATION-2)
  TOP (PROVE (ENABLE ADD-CODE SET-LABEL-ALIST)) (DEMOTE 21) S (S LEMMAS) (DIVE 1)
  (REWRITE BEGIN-MEANING-R-2) TOP S (S LEMMAS) X (S LEMMAS) (DEMOTE 21) S (S LEMMAS)
  (DEMOTE 21) S (S LEMMAS))))

42
Theorem: begin-body-normal-exact-time-schema
\((\text{stmt-time} = (2 + \text{body-time}))\)
\(\land\ (p(\text{initial}, \text{body-time}) = \text{state2})\)
\(\land\ (p(\text{state2}, 2) = \text{final}))\)
\(\rightarrow (p(\text{initial}, \text{stmt-time}) = \text{final})\)

Theorem: begin-body-normal-clock
\((\text{car (stmt)} = '\text{begin-mg})\)
\(\land\ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land\ (n \not\equiv 0)\)
\(\land\ \text{normal (mg-state)}\)
\(\land\ (\neg \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes)))}\)
\(\land\ \text{normal (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, sizes)))}\)
\(\rightarrow (\text{clock (stmt, proc-list, mg-state, n)}\)
\(= (2 + \text{clock (begin-body (stmt), proc-list, mg-state, n - 1)))\)

Theorem: begin-body-nonnormal-nonwhen-exact-time-schema
\((\text{stmt-time} = \text{body-time})\)
\(\land\ (p(\text{initial}, \text{body-time}) = \text{state2})\)
\(\land\ (\text{state2} = \text{final}))\)
\(\rightarrow (p(\text{initial}, \text{stmt-time}) = \text{final})\)

Theorem: begin-body-nonnormal-nonwhen-clock
\((\text{car (stmt)} = '\text{begin-mg})\)
\(\land\ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land\ (n \not\equiv 0)\)
\(\land\ \text{normal (mg-state)}\)
\(\land\ (\neg \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes)))}\)
\(\land\ (\neg \text{normal (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, sizes)))}\)
\(\land\ \text{cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, sizes))}\)
\(\land\ (\text{cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n - 1, sizes))})\)
\(\not\in\ \text{when-labels (stmt)})\)
\(\rightarrow (\text{clock (stmt, proc-list, mg-state, n)}\)
\(= \text{clock (begin-body (stmt), proc-list, mg-state, n - 1))}\)

Theorem: begin-when-normal-exact-time-schema
\((\text{stmt-time} = (\text{body-time} + (3 + \text{when-arm-time})))\)
\(\land\ (p(\text{initial}, \text{body-time}) = \text{state2})\)
\(\land\ (p(\text{state2}, 2) = \text{state3})\)
\(\land\ (p(\text{state3}, \text{when-arm-time}) = \text{state4})\)
\(\land\ (p\text{-step (state4) = final})\)
\(\rightarrow (p(\text{initial}, \text{stmt-time}) = \text{final})\)
THEOREM: begin-when-normal-clock
\[
((\text{car}(\text{stmt}) = \begin{command}{begin-mg}) \\
\land \text{ok-mg-statement}(\text{stmt}, r\text{-cond-list}, \text{name-alist}, \text{proc-list}) \\
\land (n \not\equiv 0) \\
\land \text{normal}(\text{mg-state}) \\
\land (\neg \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, n, \text{sizes}))) \\
\land (\neg \text{normal}(\text{mg-meaning-r}(\text{begin-body}(\text{stmt}), \\
\quad \text{proc-list}, \\
\quad \text{mg-state}, \\
\quad n - 1, \\
\quad \text{sizes})))) \\
\land (\text{cc}(\text{mg-meaning-r}(\text{begin-body}(\text{stmt}), \text{proc-list}, \text{mg-state}, n - 1, \text{sizes}))) \\
\in \text{when-labels}(\text{stmt}) \\
\land \text{normal}(\text{mg-meaning-r}(\text{when-handler}(\text{stmt}), \\
\quad \text{proc-list}, \\
\quad \text{set-condition}(\text{mg-meaning-r}(\text{begin-body}(\text{stmt}), \\
\quad \text{proc-list}, \\
\quad \text{mg-state}, \\
\quad n - 1, \\
\quad \text{sizes})))) \\
\quad \begin{command}{normal}, \\
\quad n - 1, \\
\quad \text{sizes}))) \\
\rightarrow (\text{clock}(\text{stmt}, \text{proc-list}, \text{mg-state}, n) \\
= (\text{clock}(\text{begin-body}(\text{stmt}), \text{proc-list}, \text{mg-state}, n - 1) \\
\quad + 3 \\
\quad + \text{clock}(\text{when-handler}(\text{stmt}), \\
\quad \text{proc-list}, \\
\quad \text{set-condition}(\text{mg-meaning-r}(\text{begin-body}(\text{stmt}), \\
\quad \text{proc-list}, \\
\quad \text{mg-state}, \\
\quad n - 1, \\
\quad \text{sizes})))) \\
\quad \begin{command}{normal}, \\
\quad n - 1, \\
\quad \text{sizes})))
\]

THEOREM: begin-when-nonnormal-exact-time-schema
\[
((\text{stmt-time} = (\text{body-time} + (2 + \text{when-arm-time}))) \\
\land (\text{p}(\text{initial}, \text{body-time}) = \text{state2}) \\
\land (\text{p}(\text{state2}, 2) = \text{state3}) \\
\land (\text{p}(\text{state3}, \text{when-arm-time}) = \text{state4}) \\
\land (\text{state4} = \text{final}) \\
\rightarrow (\text{p}(\text{initial}, \text{stmt-time}) = \text{final})
\]

THEOREM: begin-when-nonnormal-clock
\[
((\text{car}(stmt) = \text{'begin-mg}) \\
\land \text{ok-mg-statement}(stmt, r\text{-}\text{cond-list}, \text{name-alist}, \text{proc-list}) \\
\land (n \not\equiv 0) \\
\land \text{normal}(mg\text{-state}) \\
\land (\neg \text{resource-errorp}(\text{mg}\text{-}\text{meaning-r}(stmt, proc-list, mg\text{-state}, n, sizes))) \\
\land (\neg \text{normal}(\text{mg}\text{-}\text{meaning-r}(\text{begin-body}(stmt), \\
\quad \text{proc-list}, \\
\quad mg\text{-state}, \\
\quad n - 1, \\
\quad sizes))) \\
\land (cc(\text{mg}\text{-}\text{meaning-r}(\text{begin-body}(stmt), proc-list, mg\text{-state}, n - 1, sizes)) \\
\quad \in \text{when-labels}(stmt)) \\
\land (\neg \text{normal}(\text{mg}\text{-}\text{meaning-r}(\text{when-handler}(stmt), \\
\quad proc-list, \\
\quad set-condition(\text{mg}\text{-}\text{meaning-r}(\text{begin-body}(stmt), \\
\quad proc-list, \\
\quad mg\text{-state}, \\
\quad n - 1, \\
\quad sizes)))); \\
\quad '\text{normal}), \\
\quad n - 1, \\
\quad sizes))) \\
\rightarrow (\text{clock}(stmt, proc-list, mg\text{-state}, n) \\
\quad = (\text{clock}(\text{begin-body}(stmt), proc-list, mg\text{-state}, n - 1) \\
\quad \quad + 2 \\
\quad \quad + \text{clock}(\text{when-handler}(stmt), \\
\quad \quad proc-list, \\
\quad \quad set-condition(\text{mg}\text{-}\text{meaning-r}(\text{begin-body}(stmt), \\
\quad \quad proc-list, \\
\quad \quad mg\text{-state}, \\
\quad \quad n - 1, \\
\quad \quad sizes)));
\quad '\text{normal}), \\
\quad n - 1)))
\]

(prove-lemma begin-exact-time-lemma (rewrite)
  (IMPLIES (AND (NOT (ZEROP N)) \\
\quad (NOT (RESOURCES-INADEQUATEP STMT PROC-LIST \\
\quad (LIST (LENGTH TEMP-STK) \\
\quad (P-CTRL-STK-SIZE CTRL-STK)))))) \\
\quad (EQUAL (CAR STMT) 'BEGIN-MG))

45
(OK-MG-STATEMENT STMT R-COND-LIST NAME-ALIST PROC-LIST)
(OK-MG-DEF-PLISTP PROC-LIST)
(OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
(OK-MG-STATEP MG-STATE R-COND-LIST)
(COND-SUBSETP R-COND-LIST T-COND-LIST)
(EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST) PROC-LIST))
(APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST)) CODE2))
(USER-DEFINED-PROC SUBR PROC-LIST)
(PLISTP TEMP-STK)
(LISTP CTRL-STK)
(MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE)
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK)) (MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE) NAME-ALIST)
(NORMAL MG-STATE)
(AL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(IMPLIES
(AND
(OK-MG-STATEMENT (BEGIN-BODY STMT)
(APPEND (WHEN-LABELS STMT) R-COND-LIST)
NAME-ALIST PROC-LIST)
(OK-MG-DEF-PLISTP PROC-LIST)
(OK-TRANSLATION-PARAMETERS
MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
LABEL-CNT CINFO))
LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO))))
(T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST
(CONS
(LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS
(CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2)))))))
(CONS
  '(POP-GLOBAL C-C)
  (APPEND
   (CODE
    (TRANSLATE
     (NULLIFY
      (SET-LABEL-ALIST
       (TRANSLATE
        (MAKE-CINFO (CODE CINFO)
         (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
           (LABEL-CNT CINFO))
          (LABEL-ALIST CINFO))
         (ADD1 (ADD1 (LABEL-CNT CINFO))))
        T-COND-LIST
        (BEGIN-BODY STMT)
        PROC-LIST)
       (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
    T-COND-LIST
    (WHEN-HANDLER STMT)
    PROC-LIST))
  (CONS (CONS 'DL
    (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP))))
  CODE2))))))
  (OK-MG-STATEP MG-STATE
   (APPEND (WHEN-LABELS STMT)
    R-COND-LIST))
  (COND-SUBSETP (APPEND (WHEN-LABELS STMT)
    R-COND-LIST)
    T-COND-LIST)
  (EQUAL
   (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
    PROC-LIST))
   (APPEND
    (CODE (TRANSLATE (MAKE-CINFO (CODE CINFO)
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
     T-COND-LIST
     (BEGIN-BODY STMT)
     PROC-LIST))
    (CONS
     (LIST 'JUMP (ADD1 (LABEL-CNT CINFO))))
(P-CTRL-STK-SIZE CTRL-STK)))))

(EQUAL

(P

(MAP-DOWN MG-STATE PROC-LIST CTRL-STK TEMP-STK) ;; state1

(TAG 'PC

(CONS SUBR

(LENGTH (CODE (MAKE-CINFO (CODE CINFO)

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)

(LABEL-CNT CINFO))

(LABEL-ALIST CINFO))

(ADD1 (ADD1 (LABEL-CNT CINFO)))))

T-COND-LIST)

(CLOCK (BEGIN-BODY STMT) PROC-LIST MG-STATE (SUB1 N))) ;; body-time

(P-STATE

(TAG 'PC

(CONS SUBR

(IF

(NORMAL (MG-MEANING-R (BEGIN-BODY STMT)

PROC-LIST MG-STATE

(SUB1 N)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(LENGTH

(CODE

(TRANSLATE (MAKE-CINFO (CODE CINFO)

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)

(LABEL-CNT CINFO))

(LABEL-ALIST CINFO))

(ADD1 (ADD1 (LABEL-CNT CINFO)))))

T-COND-LIST

(BEGIN-BODY STMT)

PROC-LIST))))

(FIND-LABEL

(FETCH-LABEL

(CC (MG-MEANING-R (BEGIN-BODY STMT)

PROC-LIST MG-STATE

(SUB1 N)

(LIST (LENGTH TEMP-STK)

(P-CTRL-STK-SIZE CTRL-STK))))

(LABEL-ALIST

(TRANSLATE (MAKE-CINFO (CODE CINFO)

(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)

(LABEL-CNT CINFO))

(LABEL-ALIST CINFO))

(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST))
(APPEND
(CODE
  (TRANSLATE (MAKE-CINFO (CODE CINFO))
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
    (LABEL-CNT CINFO))
    (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST))
  (CONS
   (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
   (CONS
    (CONS 'DL
      (CONS (LABEL-CNT CINFO)
        '(NIL (PUSH-CONSTANT (NAT 2)))))
    (CONS
      '(POP-GLOBAL C-C))
APPEND
(CODE
  (TRANSLATE
   (NULLIFY
    (SET-LABEL-ALIST
     (TRANSLATE
      (MAKE-CINFO (CODE CINFO))
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
        (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
    T-COND-LIST
    (BEGIN-BODY STMT)
    PROC-LIST))
    (LABEL-ALIST CINFO))
T-COND-LIST
(WHEN-HANDLER STMT)
PROC-LIST))
  (CONS (CONS 'DL
    (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))
    (CONS (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))
  (CONS (CONS (ADD1 (LABEL-CNT CINFO))
    (CONS (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))
  (CONS (CONS (ADD1 (LABEL-CNT CINFO))
    (CONS (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))
  (CONS (CONS (ADD1 (LABEL-CNT CINFO))
    (CONS (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))
  (CONS (CONS (ADD1 (LABEL-CNT CINFO))
    (CONS (CONS (ADD1 (LABEL-CNT CINFO))
      '(NIL (NO-OP)))))))))

50
CTRL-STK
(MAP-DOWN-VALUES
(MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
(LIST 'C-C
(MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))
T-COND-LIST)))
(MG-MAX-CTRL-STK-SIZE)
(MG-MAX-TEMP-STK-SIZE)
(MG-WORD-SIZE)
'RUN)))
(IMPLIES
(AND
(OK-MG-STATEMENT (WHEN-HANDLER STMT)
R-COND-LIST NAME-ALIST PROC-LIST)
(OK-MG-DEF-PLISTP PROC-LIST)
(OK-TRANSLATION-PARAMETERS
(ADD-CODE
(SET-LABEL-ALIST
(TRANSLATE (MAKE-CINFO (CODE CINFO))
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
(LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO)))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO))
(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS (CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2))))
'((POP-GLOBAL C-C)))))
T-COND-LIST
  (WHEN-HANDLER STMT)
PROC-LIST
  (CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO)))
  '(NIL (NO-OP)))))
CODE2))
  (OK-MG-STATEP
  (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))
  'NORMAL)
R-COND-LIST)
  (COND-SUBSETP R-COND-LIST T-COND-LIST)
  (EQUAL
  (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
  PROC-LIST)))
  (APPEND
  (CODE
  (TRANSLATE
  (ADD-CODE
  (SET-LABEL-ALIST
  (TRANSLATE (MAKE-CINFO (CODE CINFO))
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO))
  (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)
  (LABEL-ALIST CINFO))
  (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
  (CONS (CONS 'DL
  (CONS (LABEL-CNT CINFO)
  '(NIL (PUSH-CONSTANT (NAT 2))))))))
  '((POP-GLOBAL C-C)))))
T-COND-LIST
  (WHEN-HANDLER STMT)
PROC-LIST))
  (CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO))
  '(NIL (NO-OP)))))
(USER-DEFINED-PROC SUBR PROC-LIST)

(PLISTP TEMP-STK)

(LISTP CTRL-STK)

(MG-VARS-LIST-O.K.-IN-P-STATE
  (MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))

  'NORMAL))

  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)

  (NO-P-ALIASING
    (BINDINGS (TOP CTRL-STK))
    (MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK)))

  'NORMAL))

  (SIGNATURES-MATCH
    (MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK)))

  'NORMAL))

  NAME-ALIST)

  (NORMAL (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK)))

  'NORMAL))

  (ALL-CARS-UNIQUE
    (MG-ALIST (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
    (P-CTRL-STK-SIZE CTRL-STK)))

  'NORMAL))

  (NUT
   (RESOURCE-ERRORP
(MG-MEANING-R
 (WHEN-HANDLER STMT)
 PROC-LIST
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
 PROC-LIST MG-STATE
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK)))
 'NORMAL)
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK))))
 (EQUAL
 (P
 (MAP-DOWN ;; state3
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
 PROC-LIST MG-STATE
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK)))
 'NORMAL)
 PROC-LIST CTRL-STK TEMP-STK
 (TAG 'PC
 (CONS SUBR
 (LENGTH
 (CODE
 (ADD-CODE
 (SET-LABEL-ALIST
 (TRANSLATE (MAKE-CINFO (CODE CINFO)
 (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
 (LABEL-ALIST CINFO))
 (ADD1 (ADD1 (LABEL-CNT CINFO))))
 T-COND-LIST
 (BEGIN-BODY STMT)
 PROC-LIST)
 (LABEL-ALIST CINFO))
 (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
 (CONS (CONS 'DL
 (CONS (LABEL-CNT CINFO)
 '(NIL (PUSH-CONSTANT (NAT 2))))
 '(((POP-GLOBAL C-))))))
 T-COND-LIST)
 (CLOCK (WHEN-HANDLER STMT) ;; when-arm-time
 54)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
‘NORMAL)
(SUB1 N)))
(P-STATE
(TAG ’PC
(CONS SUBR
(IF
(NORMAL
(MG-MEANING-R
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
‘NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
LENGTH
(CODE
(TRANSLATE
(ADD-CODE
(SET-LABEL-ALIST
(TRANSLATE
(MAKE-CINFO (CODE CINFO)
(APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
 (LABEL-ALIST CINFO))
 (ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO))
(CONS (LIST ’JUMP (ADD1 (LABEL-CNT CINFO)))
 (CONS (CONS ’DL
 (CONS (LABEL-CNT CINFO)
 ’(NIL (PUSH-CONSTANT (NAT 2)))))))
'((POP-GLOBAL C-C)))))
  T-COND-LIST
  (WHEN-HANDLER STMT)
  PROC-LIST))
(FIND-LABEL
 (FETCH-LABEL
 (CC
 (MG-MEANING-R
 (WHEN-HANDLER STMT)
 PROC-LIST
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
 PROC-LIST MG-STATE
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK))))
 'NORMAL)
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK))))
 (LABEL-ALIST
 (TRANSLATE
 (ADD-CODE
 (SET-LABEL-ALIST
 (TRANSLATE
 (MAKE-CINFO (CODE CINFO)
 (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
 (LABEL-ALIST CINFO))
 (ADD1 (ADD1 (LABEL-CNT CINFO))))
 T-COND-LIST
 (BEGIN-BODY STMT)
 PROC-LIST)
 (LABEL-ALIST CINFO))
 (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
 (CONS (CONS 'DL
 (CONS (LABEL-CNT CINFO)
 '(NIL (PUSH-CONSTANT (NAT 2))))))
 '(((POP-GLOBAL C-C)))))
 T-COND-LIST
 (WHEN-HANDLER STMT)
 PROC-LIST)))
(APPEND
 (CODE
 (TRANSLATE

56
(ADD-CODE
(SET-LABEL-ALIST
(TRANSLATE
 (MAKE-CINFO (CODE CINFO))
 (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
 (LABEL-CNT CINFO))
 (LABEL-ALIST CINFO))
 (ADD1 (ADD1 (LABEL-CNT CINFO))))
 T-COND-LIST
 (BEGIN-BODY STMT)
 PROC-LIST)
 (LABEL-ALIST CINFO))
 (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
 (CONS (CONS 'DL
 (CONS (LABEL-CNT CINFO)
 ' ((NIL (PUSH-CONSTANT (NAT 2)))))
 ' ((POP-GLOBAL C-C))))
 T-COND-LIST
 (WHEN-HANDLER STMT)
 PROC-LIST))
 (CONS (CONS 'DL
 (CONS (ADD1 (LABEL-CNT CINFO))
 ' ((NIL (NO-OP)))))
 CODE2))))))
 CTRL-STK
 (MAP-DOWN-VALUES
 (MG-ALIST
 (MG-MEANING-R
 (WHEN-HANDLER STMT)
 PROC-LIST
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
 PROC-LIST MG-STATE
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK))))
 'NORMAL)
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK)))
 (BINDINGS (TOP CTRL-STK))
 TEMP-STK)
 (TRANSLATE-PROC-LIST PROC-LIST)
 (LIST
 (LIST 'C-C

57
(MG-COND-TO-P-NAT
  (CC
  (MG-MEANING-R
   (WHEN-HANDLER STMT)
   PROC-LIST
   (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
     PROC-LIST MG-STATE
     (SUB1 N)
     (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK))))
   'NORMAL)
   (SUB1 N)
   (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK)))
   T-COND-LIST))
   (MG-MAX-CTRL-STK-SIZE)
   (MG-MAX-TEMP-STK-SIZE)
   (MG-WORD-SIZE)
   'RUN))
   (EQUAL
   (P (MAP-DOWN MG-STATE PROC-LIST CTRL-STK TEMP-STK)
     ;; initial
     (TAG 'PC
     (CONS SUBR (LENGTH (CODE CINFO))))
     T-COND-LIST)
     (CLOCK STMT PROC-LIST MG-STATE N))
     ;; stmt-time
     (P-STATE
     ;; final
     (TAG 'PC
     (CONS SUBR
     (IF
     (NORMAL (MG-MEANING-R STMT PROC-LIST MG-STATE N
     (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK)))
     (LENGTH (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST)))
     (FIND-LABEL
     (FETCH-LABEL (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
     (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK)))
     (LABEL-ALIST (TRANSLATE CINFO T-COND-LIST STMT
     PROC-LIST)))
     (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
     CODE2)))))
CTRL-STK
(MAP-DOWN-VALUES (MG-ALIST (MG-MEANING-R STMT PROC-LIST MG-STATE N
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(Translate-Proc-List Proc-List)
(List
(List 'C-C
(MG-COND-TO-P-NAT (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
T-COND-LIST)))
(MG-MAX-CTRL-STK-SIZE)
(MG-MAX-TEMP-STK-SIZE)
(MG-WORD-SIZE)
'RUN))
((INSTRUCTIONS PROMOTE
(ADD-ABBREVIATION @INITIAL
(MAP-DOWN MG-STATE PROC-LIST CTRL-STK TEMP-STK
(TAG 'PC
(CONS SUBR (LENGTH (CODE CINFO))))
T-COND-LIST))
(ADD-ABBREVIATION @STMT-TIME
(CLOCK STMT PROC-LIST MG-STATE N))
(ADD-ABBREVIATION @FINAL
(P-STATE
(TAG 'PC
(CONS SUBR
(IF
(NORMAL (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(LENGTH (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))))
(FIND-LABEL
(FETCH-LABEL (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(LABEL-ALIST (TRANSLATE CINFO T-COND-LIST STMT
PROC-LIST))
(APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT
PROC-LIST))
CODE2))))))
CTRL-STK
(MAP-DOWN-VALUES
(MG-ALIST (MG-MEANING-R STMT PROC-LIST MG-STATE N
(List LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
59
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
(LIST 'C-C
  (MG-COND-TO-P-NAT (CC (MG-MEANING-R STMT PROC-LIST MG-STATE N
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
          T-COND-LIST)))))
(MG-MAX-CTRL-STK-SIZE)
(MG-MAX-TEMP-STK-SIZE)
(MG-WORD-SIZE)
'RUN))
(ADD-ABBREVIATION @BODY-TIME
  (CLOCK (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N))))
(ADD-ABBREVIATION @STATE2
(P-STATE
(TAG 'PC
(CONS SUBR
(IF
  (NORMAL (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  (LENGTH
  (CODE
  (TRANSLATE (MAKE-CINFO (CODE CINFO)
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
  (LABEL-CNT CINFO))
  (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))))
(FIND-LABEL
(FETCH-LABEL
(CC (MG-MEANING-R (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))

60
(LABEL-ALIST
  (TRANSLATE (MAKE-CINFO (CODE CINFO))
    (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
        (LABEL-CNT CINFO))
      (LABEL-ALIST CINFO))
    (ADD1 (ADD1 (LABEL-CNT CINFO))))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST)))

(APPEND
  (CODE
    (TRANSLATE (MAKE-CINFO (CODE CINFO))
      (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
          (LABEL-CNT CINFO))
        (LABEL-ALIST CINFO))
      (ADD1 (ADD1 (LABEL-CNT CINFO))))
    T-COND-LIST
    (BEGIN-BODY STMT)
    PROC-LIST))

(CONS
  (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS 'DL
  (CONS (LABEL-CNT CINFO)
    '(NIL (PUSH-CONSTANT (NAT 2)))))
(CONS
  '(POP-GLOBAL C-C)
(CONS 'DL
  (CONS (LABEL-CNT CINFO)
    '(NIL (PUSH-CONSTANT (NAT 2)))))
  (LABEL-ALIST CINFO)))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST))

(LABEL-ALIST CINFO))
  T-COND-LIST
  (BEGIN-BODY STMT)
  PROC-LIST))

T-COND-LIST
(WHEN-HANDLER STMT)
(CONS (CONS 'DL
  (CONS (ADD1 (LABEL-CNT CINFO))
    '(NIL (NO-OP)))))
CODE2))))))))))
CTRL-STK
(MAP-DOWN-VALUES
  (MG-ALIST (MG-MEANING-R (BEGIN-BODY STMT)
    PROC-LIST MG-STATE
    (SUB1 N)
    (LIST (LENGTH TEMP-STK)
      (P-CTRL-STK-SIZE CTRL-STK))))
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)
(TTRANSLATE-PROC-LIST PROC-LIST)
 (LIST
   (LIST 'C-C
     (MG-COND-TO-P-NAT (CC (MG-MEANING-R (BEGIN-BODY STMT)
       PROC-LIST MG-STATE
       (SUB1 N)
       (LIST (LENGTH TEMP-STK)
         (P-CTRL-STK-SIZE CTRL-STK))))
     T-COND-LIST)))
  (MG-MAX-CTRL-STK-SIZE)
  (MG-MAX-TEMP-STK-SIZE)
  (MG-WORD-SIZE)
  'RUN))
(ADD-ABBREVIATION @STATE3
  (MAP-DOWN
   (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
     PROC-LIST MG-STATE
     (SUB1 N)
     (LIST (LENGTH TEMP-STK)
       (P-CTRL-STK-SIZE CTRL-STK))))
   'NORMAL)
PROC-LIST CTRL-STK TEMP-STK
(TAG 'PC
  (CONS SUBR
    (LENGTH
     (CODE
      (ADD-CODE
       (SET-LABEL-ALIST
        (TRANSLATE (MAKE-CINFO (CODE CINFO)
          (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
(LABEL-CNT CINFO))
(LABEL-ALIST CINFO))
(ADD1 (ADD1 (LABEL-CNT CINFO))))
T-COND-LIST
(BEGIN-BODY STMT)
PROC-LIST)
(LABEL-ALIST CINFO))
(CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
(CONS (CONS 'DL
(CONS (LABEL-CNT CINFO)
'(NIL (PUSH-CONSTANT (NAT 2)))))))
'((POP-GLOBAL C-0))))
T-COND-LIST))
(ADD-ABBREVIATION @WHEN-ARM-TIME
(CLOCK (WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL)
(SUB1 N)))
(ADD-ABBREVIATION @STATE4
(P-STATE
(TAG 'PC
(CONS SUBR
(IF
(NORMAL
(MG-MEANING-R
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
(LENGTH
(CODE
(TRANSLATE

63
(ADD-CODE
(SET-LABEL-ALIST
(TRANSLATE
 (MAKE-CINFO (CODE CINFO)
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
   (LABEL-CNT CINFO))
   (LABEL-ALIST CINFO))
  (ADD1 (ADD1 (LABEL-CNT CINFO))))
 T-COND-LIST
 (BEGIN-BODY STMT)
 PROC-LIST)
 (LABEL-ALIST CINFO))
 (CONS (LIST 'JUMP (ADD1 (LABEL-CNT CINFO)))
 (CONS (CONS 'DL
   (CONS (LABEL-CNT CINFO)
     '(NIL (PUSH-CONSTANT (NAT 2))))
     '((POP-GLOBAL C-C))))
 T-COND-LIST
 (WHEN-HANDLER STMT)
 PROC-LIST)))
 (FIND-LABEL
 (FETCH-LABEL
 (CG
 (MG-MEANING-R
  (WHEN-HANDLER STMT)
 PROC-LIST
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
   PROC-LIST MG-STATE
   (SUB1 N)
   (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK)))
   'NORMAL)
   (SUB1 N)
   (LIST (LENGTH TEMP-STK)
     (P-CTRL-STK-SIZE CTRL-STK))))
 (LABEL-ALIST
 (TRANSLATE
 (ADD-CODE
 (SET-LABEL-ALIST
 (TRANSLATE
 (MAKE-CINFO (CODE CINFO)
  (APPEND (MAKE-LABEL-ALIST (WHEN-LABELS STMT)
   (LABEL-CNT CINFO))
   (LABEL-ALIST CINFO))))
 64
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))

'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
(LIST 'C-C
(MG-COND-TO-P-NAT
(CC
(MG-MEANING-R
(WHEN-HANDLER STMT)
PROC-LIST
(SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))))

'NORMAL)
(SUB1 N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK)))
(T-COND-LIST)))
(MG-MAX-CTRL-STK-SIZE)
(MG-MAX-TEMP-STK-SIZE)
(MG-WORD-SIZE)
'RUN))
(DEMOTE 19)
(DIVE 1 1)
PUSH UP
(DIVE 2 1)
(DIVE 1 5 2 2 1)
S TOP PROMOTE
(CLAIM (NORMAL (MG-MEANING-R (BEGIN-BODY STMT)
PROC-LIST MG-STATE
(SUB1 N))

66
(list (length temp-stk)
    (p-ctrl-stk-size ctrl-stk)))

  0)
(drop 19)
(claim (equal @stmt-time (plus 2 @body-time)) 0)
(claim (equal (p @state2 2) @final) 0)
(demote 19 21 22)
(generalize ((@state4 state4)
  (@when-arm-time when-arm-time)
  (@state3 state3)
  (@state2 state2)
  (@body-time body-time)
  (@final final)
  (@stmt-time stmt-time)
  (@initial initial)))
(drop
(use-lemma begin-body-normal-exact-time-schema)
prove
(contradict 22)
(drop 19 21 22)
(dive 1)
(rewrite p-add1-3)
(rewrite p-add1-3)
(dive 1)
(dive 1)
(rewrite begin-state2-normal-body-step1)
up
(rewrite begin-state2-normal-body-step2-equals-final)
up
(rewrite p-0-unwinding-lemma)
top s-prop
(contradict 21)
(dive 1)
(rewrite begin-body-normal-clock)
top s
(claim (not (member (cc (mg-meaning-r (begin-body stmt)
  proc-list mg-state
  (sub1 n)
  (list (length temp-stk)
    (p-ctrl-stk-size ctrl-stk)))))
    (when-labels stmt)))
  0)
(drop 19)
(CLAIM (EQUAL @STMT-TIME @BODY-TIME) 0)
(CLAIM (EQUAL @STATE2 @FINAL) 0)
(DEMOTE 19 22 23)
(GENERALIZE ((@STATE4 STATE4)
 (@WHEN-ARM-TIME WHEN-ARM-TIME)
 (@STATE3 STATE3)
 (@STATE2 STATE2)
 (@BODY-TIME BODY-TIME)
 (@FINAL FINAL)
 (@STMT-TIME STMT-TIME)
 (@INITIAL INITIAL)))

DROP
(USE-LEMMA BEGIN-BODY-NONNORMAL-NONWHEN-EXACT-TIME-SHEMA)
PROVE
(CONTRADICT 23)
(DROP 19 22 23)
(DIVE 1)
(REWRITE BEGIN-NONNORMAL-NONWHEN-BODY-STATE2-EQUALS-FINAL)
TOP S-PROP
(CONTRADICT 22)
(DIVE 1)
(REWRITE BEGIN-BODY-NONNORMAL-NONWHEN-CLOCK)
TOP S
(DEMOTE 19)
(DIVE 1 1)
PUSH TOP PROMOTE
(CLAIM (EQUAL (P @STATE2 2) @STATE3) 0)

(CLAIM
(NORMAL
(MG-MEANING-R
 (WHEN-HANDLER STMT)
 PROC-LIST
 (SET-CONDITION (MG-MEANING-R (BEGIN-BODY STMT)
 PROC-LIST MG-STATE
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK)))
 'NORMAL)
 (SUB1 N)
 (LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK))))
 0)
(CLAIM (EQUAL @STMT-TIME
               (PLUS @BODY-TIME 3 @WHEN-ARM-TIME))
          0)
(CLAIM (EQUAL (P-STEP @STATE4) @FINAL)
          0)
(DEMOTE 19 22 23 25 26)
(GENERALIZE ((@STATE4 STATE4)
               (@WHEN-ARM-TIME WHEN-ARM-TIME)
               (@STATE3 STATE3)
               (@STATE2 STATE2)
               (@BODY-TIME BODY-TIME)
               (@FINAL FINAL)
               (@STMT-TIME STMT-TIME)
               (@INITIAL INITIAL)))
DROP
(USE-LEMMA BEGIN-WHEN-NORMAL-EXACT-TIME-SCHEMA)
PROVE
(CONTRADICT 26)
(DIVE 1)
(REWRITE BEGIN-WHEN-NORMAL-STEP-STATE4-EQUALS-FINAL)
TOP S-PROP
(CONTRADICT 25)
(DIVE 1)
(REWRITE BEGIN-WHEN-NORMAL-CLOCK
 (($SIZES (LIST (LENGTH TEMP-STK)
                (P-CTRL-STK-SIZE CTRL-STK))))
TOP S
(CLAIM (EQUAL @STMT-TIME
               (PLUS @BODY-TIME 2 @WHEN-ARM-TIME))
          0)
(CLAIM (EQUAL @STATE4 @FINAL) 0)
(DEMOTE 19 22 23 25 26)
(GENERALIZE ((@STATE4 STATE4)
               (@WHEN-ARM-TIME WHEN-ARM-TIME)
               (@STATE3 STATE3)
               (@STATE2 STATE2)
               (@BODY-TIME BODY-TIME)
               (@FINAL FINAL)
               (@STMT-TIME STMT-TIME)
               (@INITIAL INITIAL)))
DROP
(USE-LEMMA BEGIN-WHEN-NONNORMAL-EXACT-TIME-SCHEMA)
PROVE
(CONTRADICT 26)
(DIVE 1)
(REWRITE BEGIN-WHEN-NONNORMAL-STATE4-EQUALS-FINAL)
TOP S-PROP
(CONTRADICT 25)
(DIVE 1)
(REWRITE BEGIN-WHEN-NONNORMAL-CLOCK
  (($SIZES (LIST (LENGTH TEMP-STK)
                (P-CTRL-STK-SIZE CTRL-STK))))
TOP S
(CONTRADICT 23)
(DIVE 1)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-O-UNWINDING-LEMMA)
(DIVE 1)
(REWRITE BEGIN-WHEN-SIGNALLED-STATE2-STEP1)
UP
(REWRITE BEGIN-WHEN-SIGNALLED-STATE2-STEP2)
TOP S-PROP SPLIT
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(DIVE 1)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
TOP S
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
(DIVE 1)
(REWRITE BEGIN-WHEN-HANDLER-HYPS)
UP S
(DROP 19)
SPLIT
(REWRITE BEGIN-BODY-HYPS)
(REWRITE BEGIN-BODY-HYPS)
(REWRITE BEGIN-BODY-HYPS)
(REWRITE BEGIN-BODY-HYPS)
(REWRITE BEGIN-BODY-HYPS)
(DIVE 1)
(REWRITE BEGIN-BODY-HYPS)
TOP S
(DIVE 1)
(REWRITE BEGIN-BODY-HYPS)
**TOP S)))

EVENT:** Make the library "c-begin".
Index

add-code, 1
all-cars-unique, 18
append-doesnt-affect-fetch-label, 1, 12
append-make-label-alist-fetch-label, 24

begin-body, 1–3, 18, 19, 43–45
begin-body-doesnt-halt, 2
begin-body-nonnornal-nonwhen-clock, 43
begin-body-nonnornal-nonwhen-exact-time-schema, 43
begin-body-normal-clock, 43
begin-body-normal-exact-time-schema, 43
begin-find-labelp-lemma1, 12
begin-find-labelp-lemma2, 12
begin-labels-subset-r-cond-list, 4
begin-meaning-r-2, 1
begin-state2-normal-body-step2-equals-final, 17
begin-translation-2, 1
begin-when-arm-doesnt-halt, 2
begin-when-nonnornal-clock, 45
begin-when-nonnornal-exact-time-schema, 44
begin-when-normal-clock, 44
begin-when-normal-exact-time-schema, 43
bindings, 17, 19, 20

cc, 2, 19, 20, 43–45
clock, 43–45
code, 1, 12, 17–19
cond-subsetp, 4, 17

code, 1, 12, 17–19
cond-subsetp, 4, 17

fetch-label, 12, 19
find-label, 19
find-labelp, 12

label-alist, 1, 18, 19

label-cnt, 1, 12, 18
length, 17–20

make-cinfo, 1, 18
make-label-alist, 1, 12, 18, 24
map-down-values, 19, 20
mg-alist, 3, 17, 18, 20
mg-cond-to-p-nat, 19, 20
mg-max-ctrl-stk-size, 19, 20
mg-max-temp-stk-size, 19, 20
mg-meaning-r, 1–3, 18–20, 43–45
mg-psw, 2, 3
mg-state, 3
mg-vars-list-ok-in-p-state, 17
mg-word-size, 19, 20

no-p-aliasing, 17
normal, 1, 2, 18, 19, 43–45
normal-not-legal-begin-label, 1

ok-mg-def-plistp, 17
ok-mg-statement, 1, 4, 12, 17, 43–45
ok-mg-statep, 17
ok-translation-parameters, 12, 17

p, 43, 44
p-ctrl-stk-size, 17–20
p-state, 19, 20
p-step, 19, 43
plistp, 17

resource-errorp, 2, 18, 43–45
resources-inadequatep, 2, 17

set-condition, 2, 44, 45
set-label-alist, 1
signal-system-error, 1, 2
signatures-match, 18

tag, 18, 19
top, 17, 19, 20
translate, 1, 17–19
translate-def-body, 17
translate-proc-list, 19, 20

user-defined-procp, 17

when-handler, 1, 2, 18, 44, 45
when-labels, 1, 2, 4, 18, 43–45