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

```
;;;
```

```
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
```

```
(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))))))))
   ((INSTRUCTIONS
     PROMOTE (DIVE 1 1 1) (REWRITE BEGIN-TRANSLATION-2) UP UP
     (S LEMMAS) (DIVE 1) (REWRITE NEW-CODE-APPENDED-TO-OLD1) (S LEMMAS)
     UP UP (S LEMMAS) X (S LEMMAS))))
```

THEOREM: begin-labels-subset-r-cond-list
((car (*stmt*) = 'begin-mg)
 ∧ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*))
→ cond-subsetp (when-labels (*stmt*), *r-cond-list*)

```
(prove-lemma begin-body-hyps (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-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))))))
       (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))))
```

```
      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))))))
    (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)))
       (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)))))))
    (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
(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)))))
            (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)
    (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)
    (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)
```

```
    (P-CTRL-STK-SIZE CTRL-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-NO-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`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-translation-parameters (*cinfo*, *t-cond-list*, *stmt*, *proc-list*, *code2*))
$\rightarrow$ ($\neg$ find-labelp (1 + label-cnt (*cinfo*), code (*cinfo*)))

THEOREM: begin-find-labelp-lemma2
((car (*stmt*) = `'begin-mg`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-translation-parameters (*cinfo*, *t-cond-list*, *stmt*, *proc-list*, *code2*))
$\rightarrow$ ($\neg$ find-labelp (label-cnt (*cinfo*), code (*cinfo*)))

THEOREM: append-doesnt-affect-fetch-label
($x \notin lst$)
$\rightarrow$ (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-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)))))
            (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))))
   (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
     (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
```

```
  (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
  PROMOTE (DIVE 1) X (S LEMMAS) (DIVE 1 1 2) (REWRITE TRANSLATE-DEF-BODY-REWRITE)
  (REWRITE BEGIN-CODE-REWRITE1) UP (REWRITE GET-LENGTH-CAR) S UP X UP X (DIVE 1)
  X UP S X (S LEMMAS) (DIVE 1 2 1) (REWRITE DEFINEDP-CAR-ASSOC) NX (DIVE 2)
  (REWRITE TRANSLATE-DEF-BODY-REWRITE) (REWRITE BEGIN-CODE-REWRITE1) UP
  (REWRITE FIND-LABEL-APPEND) (DIVE 2) X (DIVE 1) X (DIVE 1) X (DIVE 1)
  (REWRITE FIND-LABEL-APPEND) (DIVE 2) X TOP (S LEMMAS) (DIVE 1)
  (REWRITE FIND-LABELP-MONOTONIC-LESSP) TOP S S (REWRITE LABEL-CNT-MONOTONIC3)
  PROVE S (DIVE 1) (REWRITE FIND-LABELP-MONOTONIC-LESSP) TOP S PROVE S (DIVE 1)
  (REWRITE BEGIN-FIND-LABELP-LEMMA1) TOP S (REWRITE CAR-DEFINEDP-DEFINED-PROCP))))
```

THEOREM: begin-state2-normal-body-step2-equals-final

```
((n ≠ 0)
 ∧   (¬ resources-inadequatep (stmt,
                               proc-list,
                               list (length (temp-stk),
                                     p-ctrl-stk-size (ctrl-stk))))
 ∧   (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)
 ∧   (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))
∧   (¬ 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,
                         n − 1,
                         list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))
→   (p-step (p-state (tag ('pc,
                       cons (subr,
                             length (code (translate (make-cinfo (code (cinfo),
                                                                  append (make-label-alist (when-labels (stm
                                                                                            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 (ma
                                                                                                                    t-c
                                                                                                                    be
                                                                                                                    pr
                                                                                                  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,
                                                                          n − 1,
                                                                          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,
                                                                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)),
                                                         code2)) endif)),
                 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`))

```
(prove-lemma begin-nonnormal-nonwhen-body-state2-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-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                                                    ;; 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
       (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))))
```

```
    (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
                (S LEMMAS)
                S
                (= (MG-MEANING-R STMT PROC-LIST MG-STATE N
                                 (LIST (LENGTH TEMP-STK)
                                       (P-CTRL-STK-SIZE CTRL-STK)))
                   (MG-MEANING-R (BEGIN-BODY STMT)
                                 PROC-LIST MG-STATE
                                 (SUB1 N)
                                 (LIST (LENGTH TEMP-STK)
                                       (P-CTRL-STK-SIZE CTRL-STK)))
                   0)
                S
                (DIVE 2 2 2 1)
                (= F)
                TOP S
                (DIVE 2)
                (DIVE 2 2 2 1)
                (REWRITE NEW-CODE-APPENDED-TO-OLD1)
                TOP S
                (S LEMMAS)
                X
                (S LEMMAS)
                (DIVE 1)
                (REWRITE BEGIN-MEANING-R-2)
                TOP S)))
```

THEOREM: append-make-label-alist-fetch-label

$(x \in lst)$

$\rightarrow$ (cdr (assoc (x, append (make-label-alist (lst, $label$), $label\text{-}alist$)))

$=$ $label$)

```
(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-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)))))
    (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
```

```
       (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
     (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
```

```
      PROMOTE (DIVE 1) X (S LEMMAS) UP (DIVE 1 1 1 1) (REWRITE FIND-LABEL-APPEND) (DIVE 2)
      X (DIVE 1) X UP UP NX (REWRITE TRANSLATE-DEF-BODY-REWRITE) (REWRITE BEGIN-CODE-REWRIT
      UP (REWRITE GET-LENGTH-PLUS) X UP X UP X (S LEMMAS) (DIVE 1) X (DIVE 1)
      (REWRITE MAP-DOWN-VALUES-PRESERVES-LENGTH) UP (REWRITE RESOURCES-ADEQUATE-TEMP-STK-NO
      UP S (S LEMMAS) (DIVE 1 2 2 1) (REWRITE FIND-LABEL-APPEND) (DIVE 2)
      (= * 1 ((ENABLE FIND-LABELP))) TOP S PROVE (DIVE 1) (REWRITE FIND-LABELP-MONOTONIC-LE
      UP S S PROVE S (DIVE 1) (REWRITE BEGIN-FIND-LABELP-LEMMA2) TOP S
      (REWRITE SIGNATURES-MATCH-PRESERVES-MG-VARS-LIST-OK (($X (MG-ALIST MG-STATE))))
      (DIVE 2 1) (REWRITE MG-MEANING-EQUIVALENCE) TOP
      (REWRITE MG-MEANING-PRESERVES-SIGNATURES-MATCH) (REWRITE OK-MG-STATEP-ALIST-PLISTP) S
      (DIVE 1) (REWRITE BEGIN-BODY-DOESNT-HALT) TOP S (DIVE 1 1) (REWRITE MG-MEANING-EQUIVA
      TOP
      (REWRITE MG-MEANING-PRESERVES-MG-ALISTP (($R-COND-LIST (APPEND (WHEN-LABELS STMT) R-C
      (REWRITE BEGIN-BODY-HYPS) (REWRITE BEGIN-BODY-HYPS) S (DIVE 1) (REWRITE BEGIN-BODY-DO
      TOP S (DIVE 1) (REWRITE FIND-LABELP-MONOTONIC-LESSP) TOP S S PROVE S (DIVE 1)
      (REWRITE BEGIN-FIND-LABELP-LEMMA2) TOP S)))

;; 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-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
   (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)
      '(NIL (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
```

```
        (S LEMMAS) (DIVE 1) X UP S (S LEMMAS) S UP (ENABLE MAP-DOWN) S
        (PROVE (ENABLE MG-COND-TO-P-NAT CONDITION-INDEX)))))


(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-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))
            (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)))
       '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)
```

```
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
 (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
     (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 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-S
                   (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 LEMM
  (DEMOTE 21) S (S LEMMAS) (DEMOTE 21) S (S LEMMAS) (DIVE 1) (REWRITE BEGIN-MEANING-R-2) T
```

```
(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-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))
            (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
```

```
       (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
       (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
```

```
     (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 2 1) (= * T 0) UP S UP UP UP UP (DIVE 1 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 X (REWRITE GET-LENGTH-C
      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))))
```

THEOREM: begin-body-normal-exact-time-schema
((*stmt-time* = (2 + *body-time*))
∧ (p (*initial*, *body-time*) = *state2*)
∧ (p (*state2*, 2) = *final*))
→ (p (*initial*, *stmt-time*) = *final*)

THEOREM: begin-body-normal-clock
((car (*stmt*) = `'begin-mg`)
∧ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
∧ ($n \not\simeq 0$)
∧ normal (*mg-state*)
∧ (¬ resource-errorp (mg-meaning-r (*stmt*, *proc-list*, *mg-state*, *n*, *sizes*)))
∧ normal (mg-meaning-r (begin-body (*stmt*), *proc-list*, *mg-state*, $n - 1$, *sizes*)))
→ (clock (*stmt*, *proc-list*, *mg-state*, *n*)
= (2 + clock (begin-body (*stmt*), *proc-list*, *mg-state*, $n - 1$)))

THEOREM: begin-body-nonnormal-nonwhen-exact-time-schema
((*stmt-time* = *body-time*)
∧ (p (*initial*, *body-time*) = *state2*)
∧ (*state2* = *final*))
→ (p (*initial*, *stmt-time*) = *final*)

THEOREM: begin-body-nonnormal-nonwhen-clock
((car (*stmt*) = `'begin-mg`)
∧ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
∧ ($n \not\simeq 0$)
∧ normal (*mg-state*)
∧ (¬ resource-errorp (mg-meaning-r (*stmt*, *proc-list*, *mg-state*, *n*, *sizes*)))
∧ (¬ normal (mg-meaning-r (begin-body (*stmt*),
proc-list,
mg-state,
$n - 1$,
sizes)))
∧ (cc (mg-meaning-r (begin-body (*stmt*), *proc-list*, *mg-state*, $n - 1$, *sizes*))
∉ when-labels (*stmt*)))
→ (clock (*stmt*, *proc-list*, *mg-state*, *n*)
= clock (begin-body (*stmt*), *proc-list*, *mg-state*, $n - 1$))

THEOREM: begin-when-normal-exact-time-schema
((*stmt-time* = (*body-time* + (3 + *when-arm-time*)))
∧ (p (*initial*, *body-time*) = *state2*)
∧ (p (*state2*, 2) = *state3*)
∧ (p (*state3*, *when-arm-time*) = *state4*)
∧ (p-step (*state4*) = *final*))
→ (p (*initial*, *stmt-time*) = *final*)

THEOREM: begin-when-normal-clock

```
((car (stmt) = 'begin-mg)
 ∧  ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧  (n ≰ 0)
 ∧  normal (mg-state)
 ∧  (¬ resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes)))
 ∧  (¬ normal (mg-meaning-r (begin-body (stmt),
                             proc-list,
                             mg-state,
                             n − 1,
                             sizes)))
 ∧  (cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n − 1, sizes))
     ∈  when-labels (stmt))
 ∧  normal (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)))
 →  (clock (stmt, proc-list, mg-state, n)
     =  (clock (begin-body (stmt), proc-list, mg-state, n − 1)
        +  3
        +  clock (when-handler (stmt),
                  proc-list,
                  set-condition (mg-meaning-r (begin-body (stmt),
                                               proc-list,
                                               mg-state,
                                               n − 1,
                                               sizes),
                                 'normal),
                  n − 1)))
```

THEOREM: begin-when-nonnormal-exact-time-schema

```
((stmt-time = (body-time + (2 + when-arm-time)))
 ∧  (p (initial, body-time) = state2)
 ∧  (p (state2, 2) = state3)
 ∧  (p (state3, when-arm-time) = state4)
 ∧  (state4 = final))
 →  (p (initial, stmt-time) = final)
```

THEOREM: begin-when-nonnormal-clock

```
((car (stmt) = 'begin-mg)
 ∧   ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧   (n ≠ 0)
 ∧   normal (mg-state)
 ∧   (¬ resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, sizes)))
 ∧   (¬ normal (mg-meaning-r (begin-body (stmt),
                              proc-list,
                              mg-state,
                              n − 1,
                              sizes)))
 ∧   (cc (mg-meaning-r (begin-body (stmt), proc-list, mg-state, n − 1, sizes))
      ∈   when-labels (stmt))
 ∧   (¬ normal (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))))
→    (clock (stmt, proc-list, mg-state, n)
      =   (clock (begin-body (stmt), proc-list, mg-state, n − 1)
           +   2
           +   clock (when-handler (stmt),
                      proc-list,
                      set-condition (mg-meaning-r (begin-body (stmt),
                                                   proc-list,
                                                   mg-state,
                                                   n − 1,
                                                   sizes),
                                     'normal),
                      n − 1)))
```

```
(prove-lemma begin-exact-time-lemma (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-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)))))
    (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)))
     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)))
```

```
 (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)))))))
      (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 (BEGIN-BODY STMT)
  PROC-LIST MG-STATE
  (SUB1 N)
  (LIST (LENGTH TEMP-STK)
```

```
(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                                                      ;; 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
       (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)))
    (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))))
```

```
      CODE2)))
      (USER-DEFINED-PROCP SUBR PROC-LIST)
      (PLISTP TEMP-STK)
      (LISTP CTRL-STK)
      (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)
       (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-C)))))))))
T-COND-LIST)
       (CLOCK (WHEN-HANDLER STMT)                               ;; when-arm-time
```

```
      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                                                    ;; 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
       (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
```

```
       (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))))
       (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))))
```

```
                (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))))
```

```
 (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))
(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-C)))))))))
  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
```

```
   (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
        (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))
(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)
```

```
                              (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-SCHEMA)
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-0-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)
(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, 12
append-make-label-alist-fetch-label, 24

begin-body, 1–3, 18, 19, 43–45
begin-body-doesnt-halt, 2
begin-body-nonnormal-nonwhen-clock, 43
begin-body-nonnormal-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-nonnormal-clock, 45
begin-when-nonnormal-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

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