Event: Start with the library "c-proc-call1".

;; We now devote some effort to showing that the temp-stk at this
;; point (following the return) is the final temp-stk. This is
;; perhaps the most difficult section of the proof.

**Definition:**

\[
\text{popn-deposit-induction-hint}(\text{temp-stk}, n) = \begin{cases} 
\text{t} & \text{if } \text{length}(\text{temp-stk}) \approx 0 \\
\text{t} & \text{elseif } n \approx 0 \\
\text{popn-deposit-induction-hint}(\text{cdr}(\text{temp-stk}), n - 1) & \text{endif}
\end{cases}
\]

**Theorem:** popn-rput

\[
((\text{untag}(x) \not< (\text{length}(\text{temp-stk}) - n)) \land \text{ok-temp-stk-index}(x, \text{temp-stk})) \\
\rightarrow (\text{popn}(n, \text{rput}(\text{value}, \text{untag}(x), \text{temp-stk})) = \text{popn}(n, \text{temp-stk}))
\]

Event: Enable deposit-temp.

**Theorem:** popn-deposit-array-value

\[
((\text{untag}(x) \not< (\text{length}(\text{temp-stk}) - n)) \land \text{ok-temp-stk-array-index}(x, \text{temp-stk}) \land \text{length}(\text{array-value})) \\
\rightarrow (\text{popn}(n, \text{deposit-array-value}(\text{array-value}, x, \text{temp-stk})) = \text{popn}(n, \text{temp-stk}))
\]

**Theorem:** popn-locals1

\[
(\text{all-pointers-bigger}(\text{collect-pointers}(\text{bindings}, \text{alist}), \text{length}(\text{temp-stk1}) - n) \\
\land \text{mg-alistp}(\text{alist}) \\
\land \text{mg-vars-list-ok-in-p-state}(\text{alist}, \text{bindings}, \text{temp-stk1})) \\
\rightarrow (\text{popn}(n, \text{map-down-values}(\text{alist}, \text{bindings}, \text{temp-stk1})) = \text{popn}(n, \text{temp-stk1}))
\]

**Theorem:** popn-locals

\[
(\text{all-pointers-bigger}(\text{collect-pointers}(\text{bindings}, \text{alist}), \text{length}(\text{temp-stk})) \\
\land \text{mg-alistp}(\text{alist}) \\
\land (n = \text{length}(\text{lst})) \\
\land \text{mg-vars-list-ok-in-p-state}(\text{alist}, \text{bindings}, \text{append}((\text{lst}, \text{temp-stk}))) \\
\rightarrow (\text{popn}(n, \text{map-down-values}(\text{alist}, \text{bindings}, \text{append}((\text{lst}, \text{temp-stk})))) = \text{temp-stk})
\]

**Definition:**

\[
\text{drop-formals-induction-hint}(\text{alist}, \text{locals}, \text{temp-stk}, n) = \begin{cases} 
\text{t} & \text{if } \text{alist} \approx \text{nil} \\
\end{cases}
\]

1
elseif \text{caar}(\text{alist}) \in \text{listcars}(\text{locals})
then drop-formals-induction-hint(cdr(\text{alist}),
\text{locals},
\text{deposit-alist-value}(\text{car}(\text{alist}),
\text{map-call-locals}(\text{locals}, n),
\text{temp-stk}),
\text{n})
else drop-formals-induction-hint(cdr(\text{alist}), \text{locals}, \text{temp-stk}, \text{n}) \text{ endif}

\text{Theorem: map-down-values-drop-formals-restriction}
(\text{all-cars-unique}(\text{alist})
\land \text{mg-vars-list-ok-in-p-state}(\text{restrict}(\text{alist}, \text{listcars}(\text{locals})),
\text{map-call-locals}(\text{locals}, n),
\text{temp-stk}))
\rightarrow (\text{map-down-values}(\text{restrict}(\text{alist}, \text{listcars}(\text{locals})),
\text{append}(\text{map-call-locals}(\text{locals}, n), \text{lst}),
\text{temp-stk})
= \text{map-down-values}(\text{restrict}(\text{alist}, \text{listcars}(\text{locals})),
\text{map-call-locals}(\text{locals}, n),
\text{temp-stk}))

\text{Definition:}
drop-locals-induction-hint(\text{alist}, \text{formals}, \text{temp-stk}, \text{actuals}, \text{bindings})
= \text{if} \text{alist} \simeq \text{nil} \text{ then } \text{t}
elseif \text{caar}(\text{alist}) \in \text{listcars}(\text{formals})
then drop-locals-induction-hint(cdr(\text{alist}),
\text{formals},
\text{deposit-alist-value}(\text{car}(\text{alist}),
\text{map-call-formals}(\text{formals}, \text{actuals}, \text{bindings}),
\text{temp-stk}),
\text{actuals},
\text{bindings})
else drop-locals-induction-hint(cdr(\text{alist}),
\text{formals},
\text{temp-stk},
\text{actuals},
\text{bindings}) \text{ endif}

\text{Theorem: map-down-drop-locals-restriction}
(\text{all-cars-unique}(\text{alist})
\land \text{no-duplicates}(\text{append}(\text{listcars}(\text{formals}), \text{listcars}(\text{locals}))))
\rightarrow (\text{map-down-values}(\text{restrict}(\text{alist}, \text{listcars}(\text{formals})),
\text{...})
append (map-call-locals (locals, n),
    map-call-formals (formals, actuals, bindings)),
    temp-stk)
= map-down-values (restrict (alist, listcars (formals)),
    map-call-formals (formals, actuals, bindings),
    temp-stk))

**Theorem:** restrict-cons

\[(x \neq y) \rightarrow (\text{assoc } (x, \text{restrict } (\text{lst, cons } (y, z)))) = \text{assoc } (x, \text{restrict } (\text{lst, z})))\]

**Theorem:** copy-out-params-restriction-cons

\[(x \notin \text{listcars } (\text{lst1})) \rightarrow (\text{copy-out-params } (\text{lst1, lst2, restrict } (\text{new-alist, cons } (x, z)), \text{old-alist})) = \text{copy-out-params } (\text{lst1, lst2, restrict } (\text{new-alist, z}), \text{old-alist}))\]

**Theorem:** copy-out-params-restriction

\[\text{all-cars-unique } (\text{formals}) \rightarrow (\text{copy-out-params } (\text{formals, actuals, new-alist, old-alist})) = \text{copy-out-params } (\text{formals, actuals, restrict } (\text{new-alist, listcars } (\text{formals})), \text{old-alist}))\]

**Event:** Disable deposit-temp.

**Theorem:** deposit-temp-deposit-array-value-commute6

\[\text{mg-vars-list-ok-in-p-state } (\text{lst, bindings, temp-stk}) \land \text{no-p-aliasing } (\text{bindings, lst}) \land \text{mg-alistp } (\text{lst}) \land \text{all-cars-unique } (\text{lst}) \land (x \in \text{lst}) \land (y \in \text{lst}) \land (\text{car } (x) \neq \text{car } (y)) \land \neg \text{simple-mg-type-refp } (\text{cdr } (y))) \land (\text{length } (\text{value}) = \text{array-length } (\text{cdr } (y)))) \rightarrow (\text{deposit-temp } (z, \text{cadr } (\text{assoc } (\text{car } (x), \text{bindings})), \text{deposit-array-value } (\text{value}, \text{cadr } (\text{assoc } (\text{car } (y), \text{bindings})), \text{temp-stk}))) = \text{deposit-array-value } (\text{value}, \text{cadr } (\text{assoc } (\text{car } (y), \text{bindings})), \text{deposit-temp } (z, \text{temp-stk}))\]
Theorem: deposit-array-value-deposit-alist-value-commute2
\[ (\text{mg-vars-list-ok-in-p-state} (\text{lst}, \text{bindings}, \text{temp-stk})) \]
\[ \land \ \text{no-p-aliasing} (\text{bindings}, \text{lst}) \]
\[ \land \ \text{mg-alistp} (\text{lst}) \]
\[ \land \ \text{all-cars-unique} (\text{lst}) \]
\[ \land \ (x \in \text{lst}) \]
\[ \land \ (y \in \text{lst}) \]
\[ \land \ (\neg \text{simple-mg-type-refp} (\text{cadr} (x))) \]
\[ \land \ (\text{length} (\text{value}) = \text{array-length} (\text{cadr} (x))) \]
\[ \land \ (\text{car} (x) \neq \text{car} (y)) \]
\[ \rightarrow \ (\text{deposit-array-value} (\text{value}, \]
\[ \text{cdr} (\text{assoc} (\text{car} (x), \text{bindings})), \]
\[ \text{temp-stk}))) = \]
\[ \text{deposit-alist-value} (y, \]
\[ \text{bindings}, \]
\[ \text{deposit-array-value} (\text{value}, \]
\[ \text{cdr} (\text{assoc} (\text{car} (x), \]
\[ \text{bindings})), \]
\[ \text{temp-stk}))) \]

Theorem: deposit-array-value-doesnt-affect-map-down-values
\[ (\text{mg-alistp} (\text{cons} (x, \text{mg-vars}))) \]
\[ \land \ \text{all-cars-unique} (\text{cons} (x, \text{mg-vars})) \]
\[ \land \ \text{no-p-aliasing} (\text{bindings}, \text{cons} (x, \text{mg-vars})) \]
\[ \land \ \text{mg-vars-list-ok-in-p-state} (\text{cons} (x, \text{mg-vars}), \text{bindings}, \text{temp-stk}) \]
\[ \land \ (\neg \text{simple-mg-type-refp} (\text{cadr} (x))) \]
\[ \land \ (\text{length} (\text{value}) = \text{array-length} (\text{cadr} (x))) \]
\[ \rightarrow \ (\text{map-down-values} (\text{mg-vars}, \]
\[ \text{bindings}, \]
\[ \text{deposit-array-value} (\text{value}, \]
\[ \text{cdr} (\text{assoc} (\text{car} (x), \]
\[ \text{bindings})), \]
\[ \text{temp-stk}))) = \]
\[ \text{deposit-array-value} (\text{value}, \]
\[ \text{cdr} (\text{assoc} (\text{car} (x), \text{bindings})), \]
\[ \text{map-down-values} (\text{mg-vars}, \text{bindings}, \text{temp-stk})) \]

Theorem: extra-binding-doesnt-affect-copy-out-params
\[ (\text{car} (x) \notin \text{listcars} (\text{formals})) \]
\[ \rightarrow \ (\text{copy-out-params} (\text{formals}, \text{actuals}, \text{cons} (x, \text{new-alist}), \text{old-alist})) \]
\[ = \ ]
\[ (\text{copy-out-params} (\text{formals}, \text{actuals}, \text{new-alist}, \text{old-alist})) \]

Definition:
map-down-copy-out-params-induction-hint \( (\text{formals}, \text{old-alist}, \text{actuals}, \text{new-alist}) \)

\[
= \begin{cases} 
\text{if } \text{formals} \simeq \text{nil} \text{ then } \text{t} \\
\text{else } \text{map-down-copy-out-params-induction-hint} (\text{cdr (formals)}, \\
\text{set-alist-value (car (actuals),} \\
\text{caddr (assoc (caar (formals),} \\
\text{restrict (new-alist,} \\
\text{listcars (formals))))} \\
\text{old-alist),} \\
\text{cdr (actuals),} \\
\text{cdr (new-alist)}) \end{cases}
\]

\text{endif}

\text{Theorem: map-down-copy-facts}

\[
(\text{listp (formals)} \land (\text{listcars (alist)} = \text{append (listcars (formals), locals)})) \\
\rightarrow (\text{listp (alist)} \\
\land (\text{caar (alist)} = \text{caar (formals)}) \\
\land (\text{caar (alist)} \in \text{listcars (formals)}))
\]

\text{EVENT: Disable map-down-copy-facts.}

\text{Theorem: map-down-copy-facts2}

\[
(\text{listp (formals)} \\
\land (\text{listcars (alist)} = \text{append (listcars (formals), locals)}) \\
\land \text{formal-types-preserved (forms, restrict (alist, listcars (formals))))} \\
\rightarrow (\text{cadar (formals)} = \text{cadar (alist)})
\]

\text{EVENT: Disable map-down-copy-facts2.}

\text{Theorem: map-down-copy-facts3}

\[
(\text{listp (formals)} \\
\land (\text{listcars (alist)} = \text{append (listcars (formals), locals)}) \\
\land \text{all-cars-unique (alist)}) \\
\rightarrow (\text{restrict (alist, listcars (formals)}) \\
= \text{cons (car (alist), restrict (cdr (alist), listcars (cdr (formals))))})
\]

\text{EVENT: Disable map-down-copy-facts3.}

\text{;; This one gives the looping problem when interrupted during the base case.} \\
\text{;; This occurs only when I have the (maintain-rewrite-path)}

\text{Theorem: map-down-copy-out-params-relation-new}

\[
(\text{ok-actual-params-list (actuals, old-alist)} \\
\land \text{data-param-lists-match (actuals, formals, old-alist)}
\]
∧ ok-mg-formal-data-params-plistp (formals)
∧ (listcars (new-alist) = append (listcars (formals), locals))
∧ all-cars-unique (old-alist)
∧ no-p-aliasing (bindings, old-alist)
∧ mg-vars-list-ok-in-p-state (old-alist, bindings, temp-stk)
∧ mg-vars-list-ok-in-p-state (restrict (new-alist, listcars (formals)),
   map-call-formals (formals, actuals, bindings),
   map-down-values (old-alist, bindings, temp-stk))
∧ no-duplicates (actuals)
∧ all-cars-unique (formals)
∧ all-cars-unique (new-alist)
∧ mg-alistp (old-alist)
∧ mg-alistp (new-alist)
∧ formal-types-preserved (formals, restrict (new-alist, listcars (formals))))
→ (map-down-values (restrict (new-alist, listcars (formals)),
   map-call-formals (formals, actuals, bindings),
   map-down-values (old-alist, bindings, temp-stk))
   = map-down-values (copy-out-params (formals, actuals, restrict (new-alist, listcars (formals))),
   old-alist, bindings, temp-stk))

THEOREM: formals-meaning-signature
((car (stmt) = 'proc-call-mg)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-mg-def-plistp (proc-list))
→ signatures-match (make-call-param-alist (def-formals (fetch-called-def (stmt, proc-list))),
   call-actuals (stmt),
   mg-alist (mg-state)),
   restrict (mg-alist (mg-meaning (def-body (fetch-called-def (stmt, proc-list))),
proc-list, mg-state (cc (mg-state)),
make-call-var-alist (mg-alist (mg-state), stmt),
fetch-called-def (stmt, proc-list))
mg-psw (mg-state),
6
\[ n - 1 \],
\[ \text{listcars (def-formals (fetch-called-def (stmt, proc-list))))} \]

**Event:** Disable formals-meaning-signature.

**Theorem:** locals-meaning-signature

\[
((\text{car (stmt)} = \text{'proc-call-mg}) \\
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
\land \text{ok-mg-def-plistp (proc-list)}) \\
\rightarrow \text{signatures-match (def-locals (fetch-called-def (stmt, proc-list))},} \\
\text{restrict (mg-alist (mg-meaning (def-body (fetch-called-def (stmt, proc-list))},} \\
\text{proc-list,} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state), stmt,} \\
\text{fetch-called-def (stmt, proc-list))},} \\
\text{mg-psw (mg-state),} \\
\text{n - 1)}, \\
\text{listcars (def-locals (fetch-called-def (stmt, proc-list))))})} \\
\]

**Theorem:** param-alist-mg-vars-ok0

\[
(\text{ok-actual-params-list (actuals, mg-vars)} \\
\land \text{data-param-lists-match (actuals, formals, mg-vars)} \\
\land \text{ok-mg-formal-data-params-plistp (formals)} \\
\land \text{mg-alistp (mg-vars)} \\
\land \text{all-cars-unique (formals)} \\
\land \text{mg-vars-list-ok-in-p-state (mg-vars, bindings, temp-stk))}) \\
\rightarrow \text{mg-vars-list-ok-in-p-state (make-call-param-alist (formals, actuals, mg-vars),} \\
\text{map-call-formals (formals, actuals, bindings),} \\
\text{temp-stk)} \\
\]

**Theorem:** param-alist-mg-vars-ok

\[
((\text{car (stmt)} = \text{'proc-call-mg}) \\
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
\land \text{ok-mg-def-plistp (proc-list)} \\
\land \text{ok-mg-statetp (mg-state, r-cond-list)} \\
\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),} \\
\text{bindings (top (ctrl-stk)),} \\
\text{temp-stk))} \\
\]

7
\[\text{temp-stk}\]
\[\land \text{signatures-match (mg-alist (mg-state), name-alist)}\]
\[\rightarrow \text{mg-vars-list-ok-in-p-state (make-call-param-alist (def-formals (fetch-called-def (stmt, proc-list))), call-actuals (stmt), mg-alist (mg-state)), map-call-formals (def-formals (fetch-called-def (stmt, proc-list))), call-actuals (stmt), bindings (top (ctrl-stk))}, \text{temp-stk}\]

\textbf{EVENT:} Disable param-alist-mg-vars-ok.

\textbf{THEOREM:} locals-alist-mg-vars-ok0
\[\text{all-cars-unique (locals) \land ok-mg-local-data-plistp (locals)}\]
\[\rightarrow \text{mg-vars-list-ok-in-p-state (locals, map-call-locals (locals, length (temp-stk)), append (reverse (mg-to-p-local-values (locals)), temp-stk))}\]

\textbf{THEOREM:} locals-alist-mg-vars-ok
\[((\text{car (stmt) = 'proc-call-mg}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \land \text{ok-mg-def-plistp (proc-list)} \land \text{ok-mg-statep (mg-state, r-cond-list)})\]
\[\rightarrow \text{mg-vars-list-ok-in-p-state (def-locals (fetch-called-def (stmt, proc-list)), map-call-locals (def-locals (fetch-called-def (stmt, proc-list)), length (temp-stk)), append (reverse (mg-to-p-local-values (def-locals (fetch-called-def (stmt, proc-list)))), temp-stk))}\]

\textbf{THEOREM:} no-p-aliasing-in-call-alists-new
\[((\text{car (stmt) = 'proc-call-mg}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \land \text{ok-mg-def-plistp (proc-list)} \land \text{ok-mg-statep (mg-state, r-cond-list)} \land \text{plistp (temp-stk)} \land \text{listp (ctrl-stk)} \land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}\]
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ all-cars-unique (mg-alist (mg-state)))
→ no-p-aliasing (append (map-call-formals (def-formals (fetch-called-def (stmt, proc-list)),
  call-actuals (stmt),
  bindings (top (ctrl-stk)),
  map-call-locals (def-locals (fetch-called-def (stmt, proc-list)),
  length (temp-stk)),
  append (restrict (mg-alist (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
  proc-list, mg-state (cc (mg-state),
  make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list),
  mg-psw (mg-state)),
  n – 1)),
  listcars (def-formals (fetch-called-def (stmt, proc-list))))),
  restrict (mg-alist (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
  proc-list, mg-state (cc (mg-state),
  make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list),
  mg-psw (mg-state)),
  n – 1)),
  listcars (def-locals (fetch-called-def (stmt, proc-list)))))

;; The proof of this is incredible. This is a prime candidate for
;; cleaning up the proof.

;; At this point, we have exited from the call.

THEOREM: ret-temp-stk-equals-final-temp-stk
((n \neq 0) 
∧ (¬ resources-inadequatelyp (stmt,}
\(\text{proc-list},\)
\(\text{list}(\text{length}(\text{temp-stk}),\)
\(\text{p-ctrl-stk-size}(\text{ctrl-stk})))\)
\(\land\)
\((\text{car}(\text{stmt}) = '\text{proc-call-mg})\)
\(\land\)
\((\text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})\)
\(\land\)
\((\text{ok-mg-def-plistp}(\text{proc-list})\)
\(\land\)
\((\text{ok-translation-parameters}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2})\)
\(\land\)
\((\text{cond-subsetp}(\text{r-cond-list}, \text{t-cond-list})\)
\(\land\)
\((\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))\)
\(\land\)
\((\text{user-defined-proc}(\text{subr}, \text{proc-list})\)
\(\land\)
\((\text{plistp}(\text{temp-stk})\)
\(\land\)
\((\text{listp}(\text{ctrl-stk})\)
\(\land\)
\((\text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}),\)
\(\text{bindings}(\text{top}(\text{ctrl-stk})),\)
\(\text{temp-stk})\)
\(\land\)
\((\text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))\)
\(\land\)
\((\text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})\)
\(\land\)
\((\text{normal}(\text{mg-state})\)
\(\land\)
\((\text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))\)
\(\land\)
\((\neg\ \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt},\)
\(\text{proc-list},\)
\(\text{mg-state},\)
\(n,\)
\(\text{list}(\text{length}(\text{temp-stk}),\)
\(\text{p-ctrl-stk-size}(\text{ctrl-stk}))))\))\)
\(\rightarrow\)
\((\text{popn}(\text{data-length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))),\)
\(\text{map-down-values}(\text{mg-alist}(\text{mg-meaning-r}(\text{def-body}(\text{fetch-called-def}(\text{stmt},\)
\(\text{proc-list}))))),\)
\(\text{proc-list},\)
\(\text{mg-state}(\text{cc}(\text{mg-state}),\)
\(\text{make-call-var-alist}(\text{mg-alist}(\text{mg-state}),\)
\(\text{stmt},\)
\(\text{fetch-called-def}(\text{stmt},\)
\(\text{proc-list})),\)
\(\text{mg-psw}(\text{mg-state})),\)
\(n - 1,\)
\(\text{list}(\text{data-length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt},\)
\(\text{proc-list}))))\)
\(+\)
\(\text{length}(\text{temp-stk}),\)
\(2\)
\(+\)
\(\text{length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt},\)
\(\text{proc-list}))))\)
+ length (def-formals (fetch-called-def (stmt, proc-list))),
  append (map-call-locals (def-locals (fetch-called-def (stmt, proc-list))),
    length (temp-stk),
    map-call-formals (def-formals (fetch-called-def (stmt, proc-list))),
    call-actuals (stmt),
    bindings (top (ctrl-stk))),
  append (reverse (mg-to-p-local-values (def-locals (fetch-called-def (stmt, proc-list))))),
  map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
    bindings (top (ctrl-stk)),
    temp-stk))

= map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
  bindings (top (ctrl-stk)),
  temp-stk))

Event: Disable proc-call-meaning-2.

;; (push-global c-c)

;; (push-global c-c)

(prove-lemma call-state2-step4-effect (rewrite)
  (implies
    (and
      (not (zerop n))
      (not (resources-inadequately-stmt proc-list)
        (list (length temp-stk))
        (p-ctrl-stk-size ctrl-stk))))
    (equal (car stmt) 'proc-call-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))))

11
CODE2))
(USER-DEFINED-PROC SUBR PROC-LIST)
(PLISTP TEMP-STK)
(LISTP CTRL-STK)
(MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE))
(BINDINGS (TOP CTRL-STK))

TEMP-STK)
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK))
(MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE)
 NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE) N)
(LIST (LENGTH TEMP-STK)
 (P-CTRL-STK-SIZE CTRL-STK)))))
(equal
 (p-step
 (P-STATE
 (TAG 'PC
 (CONS SUBR
 (ADD1 (PLUS (LENGTH (CODE CINFO))
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (CALL-ACTUALS STMT))))))))
 ctrl-stk

(POPN
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(MAP-DOWN-VALUES
 (MG-ALIST
 (MG-MEANING-R
 (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
 PROC-LIST
 (MG-STATE (CC MG-STATE)
 (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
 STMT
 (FETCH-CALLED-DEF STMT PROC-LIST))
 (MG-PSW MG-STATE))
 (SUB1 N)
 (LIST (PLUS (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH TEMP-STK)))
 (PLUS 2
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (DEF-FORMALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(P-CTRL-STK-SIZE CTRL-STK))))
(APPEND (MAP-CALL-LOCALS (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))
  (LENGTH TEMP-STK))
(MAP-CALL-FORMALS (DEF-FORMALS (FETCH-CALLED-DEF STMT PROC-LIST))
  (CALL-ACTUALS STMT)
  (BINDINGS (TOP CTRL-STK))))
(APPEND
  (REVERSE
    (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
  (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
    (BINDINGS (TOP CTRL-STK))
    TEMP-STK)))))
(TRANSLATE-PROC-LIST PROC-LIST)
LIST
  (LIST 'C-C
  (MG-COND-TO-P-NAT
   (CC
    (MG-MEANING-R
     (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
     PROC-LIST
     (MG-STATE (CC MG-STATE)
       (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
        STMT
        (FETCH-CALLED-DEF STMT PROC-LIST))
        (MG-PSW MG-STATE))
        (SUB1 N))
      LIST
      (PLUS
       (LENGTH
        (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
       (LENGTH (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
         (BINDINGS (TOP CTRL-STK))
         TEMP-STK)))))
       (P-CTRL-STK-SIZE
        CONS
        (P-FRAME
         (APPEND
          (MAP-CALL-LOCALS (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))
           (LENGTH TEMP-STK))
          (MAP-CALL-FORMALS (DEF-FORMALS (FETCH-CALLED-DEF STMT PROC-LIST))
            (CALL-ACTUALS STMT)
            (BINDINGS (TOP CTRL-STK))))))
          (TAG 'PC
           (CONS SUBR
(ADD1
 (PLUS (LENGTH (CODE CINFO))
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (CALL-ACTUALS STMT))))
 CTRL-STK)))))
 (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
 (MG-MAX-CTRL-STK-SIZE) (MG-MAX-TEMP-STK-SIZE) (MG-WORD-SIZE)
 'RUN))
 (P-STATE
 (TAG 'PC
 (CONS SUBR
 (PLUS (LENGTH (CODE CINFO))
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (LENGTH (CALL-ACTUALS STMT)) 2)))
 CTRL-STK
 (PUSH
 (MG-COND-TO-P-NAT
 (CC
 (MG-MEANING
 (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
 PROC-LIST
 (MG-STATE (CC MG-STATE)
 (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
 STMT
 (FETCH-CALLED-DEF STMT PROC-LIST))
 (MG-PSW MG-STATE))
 (SUB1 N)))))
 (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
 (MAP-DOWN-VALUES (MG-ALIST (MG-MEANING STMT PROC-LIST MG-STATE N))
 (BINDINGS (TOP CTRL-STK))
 TEMP-STK))
 (TRANSLATE-PROC-LIST PROC-LIST)
 (LIST
 (LIST 'C-C
 (MG-COND-TO-P-NAT
 (CC
 (MG-MEANING
 (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
 PROC-LIST
 (MG-STATE (CC MG-STATE)
 (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
 STMT
 TEMP-STK))
 (TRANSLATE-PROC-LIST PROC-LIST)
 (LIST
 (LIST 'C-C
 (MG-COND-TO-P-NAT
 (CC
 (MG-MEANING
 (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
 PROC-LIST
 (MG-STATE (CC MG-STATE)
 (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
 STMT
(= (PLUS (LENGTH (CODE CINFO))
         (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
         (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
         (LENGTH (CALL-ACTUALS STMT))
         1))
UP
(S LEMMAS)
(REWRITE GET-LENGTH-PLUS)
(REWRITE GET-LENGTH-PLUS)
(REWRITE GET-LENGTH-PLUS)
(REWRITE GET-LENGTH-PLUS)
(S LEMMAS)
UP X UP X
(S LEMMAS)
(DIVE 1) X
(DIVE 1)
(REWRITE MAP-DOWN-VALUES-PRESERVES-LENGTH)
UP
(REWRITE RESOURCES-ADEQUATE-TEMP-STK-NOT-MAX)
UP
(S LEMMAS)
X
(S LEMMAS)
TOP S DROP
(Prove (ENABLE TAG))
(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)
(REWRITE MG-MEANING-PRESERVES-MG-ALISTP)
(S LEMMAS)
(S LEMMAS)
(DIVE 2)
(REWRITE LENGTH-PUSH-LOCALS-VALUES-CODE)
TOP S
(REWRITE CALLED-DEF-FORMALS-OK)
(USE-LEMMA PROC-CALL-DOESNT-HALT)
(DEMOTE 19)
(DIVE 1 1)
S TOP
(S-PROP MAKE-CALL-ENVIRONMENT)
S
(S LEMMAS)
Theorem: call-state2-step5-effect

\[(n \neq 0) \land (\neg \text{resources-inadequatep}(stmt, proc-list, list\ (\text{length}\ (temp-stk), p-ctrl-stk-size\ (ctrl-stk))))\]
\[(\text{car}(stmt) = \text{proc-call-mg})\]
\[\text{ok-mg-statement}(stmt, r-cond-list, name-alist, proc-list)\]
\[\text{ok-mg-def-plistp}(proc-list)\]
\[\text{ok-translation-parameters}(cinfo, t-cond-list, stmt, proc-list, code2)\]
\[\text{ok-mg-statep}(mg-state, r-cond-list)\]
\[\text{cond-subsetp}(r-cond-list, t-cond-list)\]
\[\text{(code}\ (\text{translate-def-body}\ (\text{assoc}(\text{subr}, proc-list), proc-list)) = \text{append}(\text{code}\ (\text{translate}(cinfo, t-cond-list, stmt, proc-list)), code2))\]
\[\text{user-defined-procp}(subr, proc-list)\]
\[\text{plistp}(temp-stk)\]
\[\text{listp}(ctrl-stk)\]
\[\text{mg-vars-list-ok-in-p-state}(mg-alist\ (mg-state), bindings\ (\text{top}(ctrl-stk)), temp-stk)\]
\[\text{no-p-aliasing}(\text{bindings}(\text{top}(ctrl-stk)), mg-alist\ (mg-state))\]
\[\text{signatures-match}(mg-alist\ (mg-state), name-alist)\]
\[\text{normal}(mg-state)\]
\[\text{all-cars-unique}(mg-alist\ (mg-state))\]
\[(\neg \text{resource-errorp}(mg\-meaning-r(stmt, proc-list, mg-state, n, list\ (\text{length}\ (temp-stk), p-ctrl-stk-size\ (ctrl-stk))))))\]

\[\rightarrow\ \text{(p-step}\ (\text{p-state}\ (\text{tag}'pc, cons(subr, length\ (\text{code}\ (cinfo))) + \text{data-length}\ (\text{def-locals}\ (\text{fetch-called-def}(stmt, proc-list)))) + \text{length}\ (\text{def-locals}\ (\text{fetch-called-def}(stmt,}\ proc-list))))\]
\[
\begin{align*}
\text{ctrl-stk}, \\
push (\text{mg-cond-to-p-nat} (cc (\text{mg-meaning} (\text{def-body} (\text{fetch-called-def} (stmt, proc-list)))), \text{proc-list}, \\
\text{mg-state} (cc (mg-state), \\
\text{make-call-var-alist} (\text{mg-alist} (mg-state), stmt, \\
\text{fetch-called-def} (stmt, proc-list)), \\
n - 1)), \\
\text{make-cond-list} (\text{fetch-called-def} (stmt, proc-list)), \\
\text{map-down-values} (\text{mg-alist} (\text{mg-meaning} (stmt, proc-list, mg-state, n)), \\
\text{bindings} (\text{top} (\text{ctrl-stk}), \text{temp-stk})), \\
\text{translate-proc-list} (\text{proc-list}), \\
\text{list} (\text{list} ('c-c, \\
\text{mg-cond-to-p-nat} (cc (\text{mg-meaning} (\text{def-body} (\text{fetch-called-def} (stmt, proc-list)))), \\
\text{proc-list}, \\
\text{mg-state} (cc (mg-state), \\
\text{make-call-var-alist} (\text{mg-alist} (mg-state, stmt, \\
\text{fetch-called-def} (stmt, proc-list)), \\
n - 1)), \\
\text{make-cond-list} (\text{fetch-called-def} (stmt, proc-list)))))
\end{align*}
\]
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt, fetch-called-def (stmt, proc-list),
mg-psw (mg-state, n - 1)),
make-cond-list (fetch-called-def (stmt, proc-list),

cons (label-cnt (cinfo),
cons (label-cnt (cinfo),
append (cond-case-jump-label-list (1 + label-cnt (cinfo),
1 + length (call-conds (stmt, proc-list))),
label-cnt-list (label-cnt (cinfo),
length (def-cond-locals (fetch-called-def (stmt, proc-list))))),
append (code (translate (cinfo, t-cond-list, stmt, proc-list)),

'code2)),
ctrl-stk,
map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list ('c-c, mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list),
proc-list), mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state, stmt),
fetch-called-def (stmt, proc-list),
mg-psw (mg-state),

n - 1)),
make-cond-list (fetch-called-def (stmt, proc-list))))),
MG-MAX-CTRL-STK-SIZE,
The Normal Return Case

In the schema for normal return, n = 1;

**Theorem:** call-add1-lc-not-in-code
\[ ((\text{car} (\text{stmt}) = \text{'proc-call-mg}) \land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \land \text{ok-mg-def-plistp} (\text{proc-list}) \land \text{ok-translation-parameters} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2})) \rightarrow (\neg \text{find-labelp} (1 + \text{label-cnt} (\text{cinfo}), \text{code} (\text{cinfo})))\]

**Theorem:** mg-cond-to-p-nat-normal
\[ \text{mg-cond-to-p-nat} (\text{'normal}, \text{state}) = \text{'(nat} 2)\]

**Theorem:** call-state2-step6-effect-normal-body-equals-final
\[ ((n \not= 0) \land (\neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk})))) \land (\text{car} (\text{stmt}) = \text{'proc-call-mg}) \land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \land \text{ok-mg-def-plistp} (\text{proc-list}) \land \text{ok-translation-parameters} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2}) \land \text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list}) \land \text{cond-subsetp} (\text{r-cond-list}, \text{t-cond-list}) \land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list})) = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}), \text{code2}))) \land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \land \text{plisp} (\text{temp-stk}) \land \text{listp} (\text{ctrl-stk}) \land \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk}) \land \text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state})) \land \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist}) \land \text{normal} (\text{mg-state}) \land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \land (\neg \text{resource-errorp} (\text{mg-meaning-r} (\text{stmt}, \text{20})))\]
\[\begin{align*}
\text{proc-list}, \\
\text{mg-state}, \\
n, \\
\text{list (length (temp-stk),} \\
p\text{-ctrl-stk-size (ctrl-stk)))}
\end{align*}\]

\[\land \text{ normal (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),} \\
\text{proc-list,} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{stmt,} \\
\text{fetch-called-def (stmt,} \\
\text{proc-list)),} \\
\text{mg-psw (mg-state)),} \\
n - 1)\]

\[\rightarrow (p\text{-step (p-state (tag ('pc,} \\
\text{cons (subr,} \\
\text{find-label (get (untag (mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
\text{proc-list),} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{stmt,} \\
\text{fetch-called-def (stmt,} \\
\text{proc-list)),} \\
\text{mg-psw (mg-state)),} \\
n - 1)),} \\
\text{make-cond-list (fetch-called-def (stmt,} \\
\text{proc-list,} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{stmt,} \\
\text{fetch-called-def (stmt,} \\
\text{proc-list)),} \\
\text{mg-psw (mg-state),} \\
n - 1))},} \\
\text{cons (label-cnt (cinfo),} \\
\text{cons (label-cnt (cinfo),} \\
\text{append (cond-case-jump-label-list (1 + label-cnt (cinfo),} \\
1 + length (call-conds (stmt)),} \\
\text{label-cnt-list (label-cnt (cinfo),} \\
length (def-cond-locals (fetch-called-def (stmt,} \\
\text{proc-list))),} \\
\text{append (code (translate (cinfo,} \\
t-\text{cond-list,} \\
stmt,} \\
\text{proc-list)),} \\
code2)))))},} \\
c\text{ctrl-stk,} \\
\text{map-down-values (mg-alist (mg-meaning (stmt,} \\
\text{proc-list,} \\
\text{mg-state),} \\
\text{21}}\]
bindings (top (\texttt{ctrl-stk})),
(temp-stk),
translate-proc-list (\texttt{proc-list}),
list (list (\texttt{\texttt{c-c}}),
  mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, 
  proc-list)),
  proc-list, 
  mg-state (cc (mg-state),
    make-call-var-alist (mg-alist (mg-state stmt, 
      fetch-called-def 
      mg-psw (mg-state),
      make-cond-list (fetch-called-def (stmt, 
        proc-list)))))),
  mg-max-ctrl-stk-size, 
  mg-max-temp-stk-size, 
  mg-word-size, 
  \texttt{\texttt{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))),
        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)

;;;;;;;;;;;;;;;;;;;;;; The Non-Normal Return Cases ;;;;;;;;;;;;;;;;;;;;;;;;;

THEOREM: call-nc-not-in-code
((car (stmt) = 'proc-call-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))
→ (¬ find-labelp (label-cnt (cinfo), code (cinfo)))

EVENT: Disable call-nc-not-in-code.

;; The 'routineerror case

THEOREM: mg-cond-to-p-nat-routineerror
mg-cond-to-p-nat ('routineerror, state) = '(nat 1)
Theorem: call-state2-step6-effect-routine-error-body

\((n \neq 0)\) 
\(\land (\neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)})))\) 
\(\land (\text{car (stmt)} = \text{proc-call-mg})\) 
\(\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\) 
\(\land \text{ok-mg-def-plistp (proc-list)}\) 
\(\land \text{ok-translation-parameters (cinfo, t-cond-list, stmt, proc-list, code2)}\) 
\(\land \text{cond-subsetp (r-cond-list, t-cond-list)}\) 
\(\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)} = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)})\) 
\(\land \text{user-defined-procp (subr, proc-list)}\) 
\(\land \text{plistp (temp-stk)}\) 
\(\land \text{listp (ctrl-stk)}\) 
\(\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}\) 
\(\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\) 
\(\land \text{signatures-match (mg-alist (mg-state), name-alist)}\) 
\(\land \text{normal (mg-state)}\) 
\(\land \text{all-cars-unique (mg-alist (mg-state))}\) 
\(\land (\neg \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)})))\) 
\(\land (\text{cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list, mg-state (cc (mg-state)), make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)), n − 1)}) = \text{routineerror})\)
→ (p-step (p-state (tag 'pc,
    cons (subr,
      find-label (get (untag (mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def
        proc-list,
        mg-state (cc (mg-state
        make-call-
        mg-psw (m
        n − 1)),
        make-cond-list (fetch-called-def (stmt,
        proc-
        cons (label-cnt (cinfo),
        cons (label-cnt (cinfo),
        append (cond-case-jump-label-list (1 + label-cnt (cinfo),
        1 + length (call-conds
        label-cnt-list (label-cnt (cinfo),
        length (def-cond-locals (fetch-
        append (code (translate (cinfo,
        t-cond-list,
        stmt,
        proc-list)),
        code2))))),
        ctrl-stk,
        map-down-values (mg-alist (mg-meaning (stmt,
        proc-list,
        mg-state,
        n)),
        bindings (top (ctrl-stk)),
        temp-stk),
        translate-proc-list (proc-list),
        list (c-c,
        mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt,
        proc-list)),
        proc-list,
        mg-state (cc (mg-state),
        make-call-var-alist (mg-alist (mg-state (stmt,
        fetch-called-de-
        mg-psw (mg-state)))))
25
\[ n - 1 \]

\[ \text{make-cond-list} \left( \text{fetch-called-def} \left( \textit{stmt}, \textit{proc-list} \right) \right) \right) \]

\[ \text{mg-max-ctrl-stk-size}, \]
\[ \text{mg-max-temp-stk-size}, \]
\[ \text{mg-word-size}, \]
\[ \text{run} \]

\[ = \text{p-state} \left( \text{tag} \left( \text{pc}, \right. \right. \]
\[ \text{cons} \left( \text{subr}, \right. \right.
\[ \text{length} \left( \text{code} \left( \textit{cinfo} \right) \right) \]
\[ + \text{length} \left( \text{push-parameters-code} \left( \text{def-locals} \left( \text{fetch-called-def} \left( \textit{stmt}, \textit{proc-list} \right) \right), \right. \right. \]
\[ \text{call-actuals} \left( \textit{stmt} \right) \right) \]
\[ + 4 \), \]
\[ \text{ctrl-stk}, \]
\[ \text{push} \left( \text{nat} 1 \right), \]
\[ \text{map-down-values} \left( \text{mg-alist} \left( \text{mg-meaning} \left( \textit{stmt}, \textit{proc-list}, \textit{mg-state}, \text{run} \right) \right), \right. \]
\[ \text{bindings} \left( \text{top} \left( \text{ctrl-stk} \right), \right. \]
\[ \text{temp-stk} \right), \]
\[ \text{translate-proc-list} \left( \textit{proc-list} \right), \]
\[ \text{\langle\langle c-c \text{nat} 1\rangle\rangle}, \]
\[ \text{mg-max-ctrl-stk-size}, \]
\[ \text{mg-max-temp-stk-size}, \]
\[ \text{mg-word-size}, \]
\[ \text{run} \right) \]

\[ \right) \]
∧ (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)))))
∧ (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list, mg-state (cc (mg-state),
                                  make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)),
                                  mg-psw (mg-state)),
                                 n − 1)) = 'routineerror)
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo))
                                  + length (push-parameters-code (def-locals (fetch-called-def (stmt, proc-list)),
                                    call-actuals (stmt)))))
                                  + 4)),
                       ctrl-stk, push ('(nat 1),
                       map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
                       bindings (top (ctrl-stk)),
                       temp-stk)),

27
\[
\text{translate-proc-list } (\text{proc-list},
\langle (c\,-\,c \text{ (nat 1)})\rangle),
\text{MG-MAX-CTRL-STK-SIZE},
\text{MG-MAX-TEMP-STK-SIZE},
\text{MG-WORD-SIZE},
\langle \text{run}\rangle)
\]
\[
= \text{p-state} (\text{tag } \langle \text{pc,}
\text{cons}(\text{subr},
\text{length (code } \text{cinfo})
+ \text{length (push-locals-values-code (def-locals (fetch-called-def } \text{stmt,}
\text{proc-list)))}
+ \text{length (def-locals (fetch-called-def } \text{stmt,}
\text{proc-list)))}
+ \text{length (call-actuals } \text{stmt))}
+ 5)),
\text{ctrl-stk},
\text{map-down-values (mg-alist (mg-meaning } \text{stmt,}
\text{proc-list,}
\text{mg-state,}
\text{n}),
\text{bindings (top } \text{ctrl-stk)}),
\text{temp-stk},
\text{translate-proc-list } (\text{proc-list},
\langle (c\,-\,c \text{ (nat 1)})\rangle),
\text{MG-MAX-CTRL-STK-SIZE},
\text{MG-MAX-TEMP-STK-SIZE},
\text{MG-WORD-SIZE},
\langle \text{run}\rangle)
\]

;; (jump "routineerror")

\text{THEOREM: call-state2-step8-effect-routineerror-body-equals-final}
\]
\[
((n \neq 0)
\land (\neg \text{resources-inadequatep } \text{stmt,}
\text{proc-list},
\text{list (length } \text{temp-stk),}
\text{p-ctrl-stk-size } \text{ctrl-stk)})))
\land (\text{car } \text{stmt} = \langle \text{proc-call-mg}\rangle)
\land \text{ok-mg-statement } \text{stmt, r-cond-list, name-alist, proc-list}
\land \text{ok-mg-def-plistp } \text{proc-list}
\land \text{ok-translation-parameters } \text{cinfo, t-cond-list, stmt, proc-list, code2}
\land \text{ok-mg-statep } \text{mg-state, r-cond-list}
\land \text{cond-subsetp } \text{r-cond-list, t-cond-list}
\]

28
\( (\text{code (translate-def-body (assoc (}\text{subr}, \text{proc-list}), \text{proc-list}))} \\
\quad = \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),} \\
\quad \quad \text{code2))} \\
\wedge \text{user-defined-procp (subr, proc-list)} \\
\wedge \text{plistp (temp-stk)} \\
\wedge \text{listp (ctrl-stk)} \\
\wedge \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),} \\
\text{bindings (top (ctrl-stk)),} \\
\text{temp-stk)} \\
\wedge \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
\wedge \text{signatures-match (mg-alist (mg-state), name-alist)} \\
\wedge \text{normal (mg-state)} \\
\wedge \text{all-cars-unique (mg-alist (mg-state))} \\
\wedge \text{(\neg resource-errorp (mg-meaning-r (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n,} \\
\text{list (length (temp-stk),} \\
\text{p-ctrl-stk-size (ctrl-stk))))))} \\
\wedge \text{(cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),} \\
\text{proc-list,} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{stmt,} \\
\text{fetch-called-def (stmt,} \\
\text{proc-list)),} \\
\text{mg-psw (mg-state))},} \\
\text{n} - 1)) \\
\quad = \quad '\text{routineerror}) \\
\rightarrow \text{(p-step (p-state (tag ('pc,} \\
\text{cons (subr,} \\
\text{length (code (cinfo))} \\
\text{+ length (push-locals-values-code (def-locals (fetch-called-def (stmt,} \\
\text{proc-list))))))} \\
\text{+ length (def-locals (fetch-called-def (stmt,} \\
\text{proc-list))))} \\
\text{+ length (call-actuals (stmt))} \\
\text{+ 5)),} \\
\text{ctrl-stk,} \\
\text{map-down-values (mg-alist (mg-meaning (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n))},} \\
\text{bindings (top (ctrl-stk)))} \)}
\[
\text{temp-stk),}
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
'((c-c (nat 1)))
\]
\[
\text{MG-MAX-CTRL-STK-SIZE,}
\]
\[
\text{MG-MAX-TEMP-STK-SIZE,}
\]
\[
\text{MG-WORD-SIZE,}
\]
\[
'\text{run})
\]
\[
= \text{p-state (tag ('pc,}
\[
\text{cons (subr,}
\]
\[
\text{if normal (mg-meaning-r (stmt,}
\]
\[
\text{proc-list,}
\]
\[
\text{mg-state,}
\]
\[
\text{n,}
\]
\[
\text{list (length (temp-stk),}
\]
\[
\text{p-ctrl-stk-size (ctrl-stk))})}
\]
\[
\text{then length (code (translate (cinfo,}
\]
\[
\text{t-cond-list,}
\]
\[
\text{stmt,}
\]
\[
\text{proc-list)))}
\]
\[
\text{else find-label (fetch-label (cc (mg-meaning-r (stmt,}
\]
\[
\text{proc-list,}
\]
\[
\text{mg-state,}
\]
\[
\text{n,}
\]
\[
\text{list (length (temp-stk),}
\]
\[
\text{p-ctrl-stk-size (ctrl-stk))}),}
\]
\[
\text{label-alist (translate (cinfo,}
\]
\[
\text{t-cond-list,}
\]
\[
\text{stmt,}
\]
\[
\text{proc-list))}),}
\]
\[
\text{append (code (translate (cinfo,}
\]
\[
\text{t-cond-list,}
\]
\[
\text{stmt,}
\]
\[
\text{proc-list))},}
\]
\[
\text{code2)) endif),}
\]
\[
\text{ctrl-stk,}
\]
\[
\text{map-down-values (mg-alist (mg-meaning-r (stmt,}
\]
\[
\text{proc-list,}
\]
\[
\text{mg-state,}
\]
\[
\text{n,}
\]
\[
\text{list (length (temp-stk),}
\]
\[
\text{p-ctrl-stk-size (ctrl-stk))}),}
\]
\[
\text{bindings (top (ctrl-stk)),}
\]
\[
\text{temp-stk),}
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
30
\]
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))

;;;;;;;;;;;;;;;;;; The Non-Normal/Non-Routineerror Exit Case ;;;;;;;;;;;;;;;;;;;

Theorem: body-condition-member-make-cond-list
((n ≠ 0) 
  ∧ (car (stmt) = 'proc-call-mg) 
  ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list) 
  ∧ ok-mg-def-plistp (proc-list) 
  ∧ ok-mg-statep (mg-state, r-cond-list) 
  ∧ signatures-match (mg-alist (mg-state), name-alist) 
  ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state), 
      bindings (top (ctrl-stk)), 
      temp-stk) 
  ∧ normal (mg-state) 
  ∧ (¬ resource-errorp (mg-meaning-r (stmt, 
          proc-list, 
          mg-state, 
          n, 
          list (length (temp-stk), 
              p-ctrl-stk-size (ctrl-stk))))) 
  → (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), 
            proc-list, 
            mg-state (cc (mg-state), 
                make-call-var-alist (mg-alist (mg-state), 
                    stmt, 
                    fetch-called-def (stmt, 
                        proc-list)), 
                mg-psw (mg-state)), 
                n - 1)) 
      ∈ cons ('normal, 
          cons ('routineerror, 
              make-cond-list (fetch-called-def (stmt, proc-list))))))
**Theorem:** leave-not-in-make-cond-list

\[
\left((\text{car}(\text{stmt}) = \text{'proc-call-mg}) \wedge \text{ok-mg-statement}(\text{stmt}, r\text{-cond-list}, \text{name-alist}, \text{proc-list}) \wedge \text{ok-mg-def-plistp}(\text{proc-list}) \wedge \text{ok-mg-statep}(\text{mg-state}, r\text{-cond-list})\right) \rightarrow ('\text{leave} \notin \text{make-cond-list}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))
\]

**Theorem:** body-condition-not-leave

\[
\left((n \neq 0) \wedge (\text{car}(\text{stmt}) = \text{'proc-call-mg}) \wedge \text{ok-mg-statement}(\text{stmt}, r\text{-cond-list}, \text{name-alist}, \text{proc-list}) \wedge \text{ok-mg-def-plistp}(\text{proc-list}) \wedge \text{ok-mg-statep}(\text{mg-state}, r\text{-cond-list}) \wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}((\text{ctrl-stk})))\right)
\]

\[
\left(\text{temp-stk}) \wedge \text{normal}(\text{mg-state}) \wedge (\neg \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, n, \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk})))))\right) \rightarrow (\text{cc}(\text{mg-meaning}(\text{def-body}(\text{fetch-called-def}(\text{stmt}, \text{proc-list}))), \text{proc-list}, \text{mg-state}(\text{cc}(\text{mg-state}), \text{make-call-var-alist}(\text{mg-alist}(\text{mg-state}), \text{stmt}, \text{fetch-called-def}(\text{stmt}, \text{proc-list})), \text{mg-psw}(\text{mg-state}), n - 1)) \neq ('\text{leave})
\]

;; checkpoint

**Definition:**

\[
\text{find-def-conds-induction-hint}(\text{index}, \text{call-conds}, \text{lc}) = \begin{cases} 
\text{t} & \text{if } \text{call-conds} \simeq \text{nil} \\
\text{find-def-conds-induction-hint}(\text{index} - 1, \text{cdr}(\text{call-conds}), 1 + \text{lc}) & \text{else}
\end{cases}
\]

32
Theorem: get-indexed-push-constant-instruction
\[
((k \not\equiv 0) \land (k < 1 + \text{length}(\text{call-conds}))) \rightarrow
\text{get}((k - 1) \ast 3, \\
\text{append}(\text{cond-conversion}(\text{call-conds}, 1 + lc, \text{cond-list}, \text{label-alist}), \\
\text{code}))
\]
\[
= \text{list}(\text{‘d1}, \\
k + lc, \\
\text{nil}, \\
\text{list}(\text{‘push-constant}, \\
\text{mg-cond-to-p-nat}(\text{get}(k - 1, \text{call-conds}), \text{cond-list})))
\]

Theorem: find-def-conds-label1
\[
((\text{index} < (1 + \text{length}(\text{call-conds})) \land (\text{index} \not\equiv 0) \land (lc \not\equiv 0)) \rightarrow
\text{find-label}(\text{index} + lc, \\
\text{append}(\text{cond-conversion}(\text{call-conds}, \\
1 + lc, \\
\text{t-cond-list}, \\
\text{label-alist}), \\
\text{code}))
\]
\[
= ((\text{index} - 1) \ast 3))
\]

Theorem: find-labelp-def-conds
\[
((\text{index} < (1 + \text{length}(\text{call-conds})) \land (\text{index} \not\equiv 0) \land (lc \not\equiv 0)) \rightarrow
\text{find-labelp}(\text{index} + lc, \\
\text{append}(\text{cond-conversion}(\text{call-conds}, \\
1 + lc, \\
\text{t-cond-list}, \\
\text{label-alist}), \\
\text{code}))
\]

Theorem: call-conds-index-lessp
\[
((\text{car}(\text{stmt}) = ‘\text{proc-call-mg}) \\
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\land (x \in \text{def-conds}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))) \\
\rightarrow ((\text{index}(x, \text{def-conds}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))) \\
< (1 + \text{length}(\text{call-conds}(\text{stmt}))))
\]
\[
= t)
\]

Theorem: call-def-cond-label-find-labelp
\[
((\text{car}(\text{stmt}) = ‘\text{proc-call-mg}) \\
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\land \text{ok-mg-def-plistp}(\text{proc-list}) \\
\land \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list}) \\
\land \text{ok-translation-parameters}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2}) \\
\land (x \in \text{def-conds}(\text{fetch-called-def}(\text{stmt}, \text{proc-list}))))
\]
\rightarrow (\neg \text{find-labelp} (\text{index} (x, \text{def-conds} (\text{fetch-called-def} (\text{stmt}, \text{proc-list}))))
+ (1 + \text{label-cnt} (\text{cinfo})), \\
\text{code} (\text{cinfo}))

;; Because of the above lemma, we know that the body-condition is either in the 
;; def-conds or in the def-local-conds. We consider each case separately.

;; (\text{push-constant} (\text{list} '\text{nat condition-index}))

\textbf{Theorem:} call-state2-step6-effect-call-conds-body
\begin{align*}
& ((n \not\approx 0) \\
& \land (\neg \text{resources-inadequatep} (\text{stmt}, \\
& \quad \text{proc-list}, \\
& \quad \text{list} (\text{length} (\text{temp-stk}), \\
& \quad \quad \text{p-ctrl-stk-size} (\text{ctrl-stk})))) \\
& \land (\text{car} (\text{stmt}) = \text{'proc-call-mg}) \\
& \land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
& \land \text{ok-mg-def-plistp} (\text{proc-list}) \\
& \land \text{ok-translation-parameters} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2}) \\
& \land \text{cond-subsetp} (\text{r-cond-list}, \text{t-cond-list}) \\
& \land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list})), \\
& \quad \quad \text{assoc} (\text{subr}, \text{proc-list})))) = \\
& \quad \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \\
& \quad \quad \text{code2})) \\
& \land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \\
& \land \text{plistp} (\text{temp-stk}) \\
& \land \text{listp} (\text{ctrl-stk}) \\
& \land \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})), \\
& \quad \text{temp-stk}) \\
& \land \text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state})) \\
& \land \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist}) \\
& \land \text{normal} (\text{mg-state}) \\
& \land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \\
& \land (\neg \text{resource-errorp} (\text{mg-meaning-r} (\text{stmt}, \\
& \quad \text{proc-list}, \\
& \quad \text{mg-state}, \\
& \quad n, \\
& \quad \text{list} (\text{length} (\text{temp-stk}), \\
& \quad \quad \text{p-ctrl-stk-size} (\text{ctrl-stk})))))) \\
& \land (\neg \text{normal} (\text{mg-meaning} (\text{def-body} (\text{fetch-called-def} (\text{stmt}, \text{proc-list})), \\
\quad \text{proc-list}, \\
\quad \text{mg-state} (\text{cc} (\text{mg-state})), \\
\quad \text{cc} (\text{mg-state})), \\
\quad \text{cc} (\text{mg-state}))}
\[\text{make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)},
\]
\[n - 1)\)
\[\wedge (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list, mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)), n - 1))
\]
\[\neq \text{routineerror}\]
\[\wedge (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list, mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)), n - 1))
\]
\[\in \text{def-conds (fetch-called-def (stmt, proc-list)))}\]
\[\rightarrow (\text{p-step (p-state (tag (\'pc, cons (subr, find-label (get (untag (mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list), mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)), n - 1))
\]
\[\in \text{def-conds (fetch-called-def (stmt, proc-list)))}\]
append (code (translate (cinfo, t-cond-list, stmt, proc-list)),

    code2))),

ctrl-stk,
map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),

    bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list ('c-c,

    mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),

        proc-list, mg-state)

    make-call-var-alist (mg-alist (mg-state stmt, fetch-called-def

    mg-psw (mg-state)),

    n − 1)),
make-cond-list (fetch-called-def (stmt, proc-list))))),

MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, cons (subr,

    length (code (cinfo))
    + length (push-parameters-code (def-locals (fetch-called-def (stmt, proc-list)),

    call-actuals (stmt)))

    + 6
    + ((index (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),

    proc-list, mg-state (cc (mg-state)),

36
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt, proc-list),
mg-psw (mg-state)),
n − 1),
def-conds (fetch-called-def (stmt, proc-list)) − 1)

+ 3)
ctrl-stk,
push (mg-cond-to-p-nat (get (index (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt, proc-list),
mg-psw (mg-state)),
n − 1)),
def-conds (fetch-called-def (stmt, proc-list))) − 1,
call-conds (stmt),
t-cond-list),
map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
bindings (top (ctrl-stk)),
temp-stk)),
translate-proc-list (proc-list),
list (list ("c-c,
mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt, fetch-called-def (stmt, proc-list),
m-g-state (mg-state)),
n − 1)),
make-cond-list (fetch-called-def (stmt,
\[\begin{align*}
\text{MG-MAX-CTRL-STK-SIZE}, \\
\text{MG-MAX-TEMP-STK-SIZE}, \\
\text{MG-WORD-SIZE}, \\
\'\text{run})
\end{align*}\]

**Theorem:** get-indexed-pop-global-instruction

\[
((k \not\approx 0) \land (k < (1 + \text{length}(\text{call-conds})))) \rightarrow \\
(\text{get} (((k - 1) \ast 3) + 1, \\
\text{append} (\text{cond-conversion}(\text{call-conds}, 1 + lc, \text{cond-list}, \text{label-alist}, \\
\text{code})) \\
= \'\text{pop-global c-c)})
\]

**Theorem:** call-state2-step7-effect-call-conds-body

\[
((n \not\approx 0) \\
\land (\neg \text{resources-inadequatep}(\text{stmt}, \\
\text{proc-list}, \\
\text{list}(\text{length}(\text{temp-stk}), \\
p-\text{ctrl-stk-size}(\text{ctrl-stk})))))
\land (\text{car}(\text{stmt}) = \'\text{proc-call-mg})
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})
\land \text{ok-mg-def-plistp}(\text{proc-list})
\land \text{ok-translation-parameters}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2})
\land \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list})
\land \text{cond-subsetp}(\text{r-cond-list}, \text{t-cond-list})
\land (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})))
\land \text{user-defined-procp}(\text{subr}, \text{proc-list})
\land \text{plistp}(\text{temp-stk})
\land \text{listp}(\text{ctrl-stk})
\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk})
\land \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))
\land \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})
\land \text{normal}(\text{mg-state})
\land \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))
\land (\neg \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, \text{n}, \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk}))))))
\]
∧ (¬ normal (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt,
proc-list)),
mg-psw (mg-state)),
n − 1)))
∧ (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt,
proc-list)),
mg-psw (mg-state)),
n − 1)) \not= 'routineerror
∧ (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt,
proc-list)),
mg-psw (mg-state)),
n − 1)) ∈ def-conds (fetch-called-def (stmt, proc-list)))
→ (p-step (p-state (tag ('pc,
cons (subr,
length (code (cinfo))
+ length (push-parameters-code (def-locals (fetch-called-def (stmt, proc-list))),
call-actuals (stmt)))
+ 6
+ ((index (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state)
stmt,
fetch-called-def (stmt, proc-list))))

39
def-conds (fetch-called-def (stmt, proc-list))) - 1) * 3) + 1)),
call-conds (stmt),
map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)),
bindings (top (ctrl-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list, mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state)), stmt, fetch-called-def (stmt, proc-list)))),
mg-psw (mg-state)),
n - 1)),
def-conds (fetch-called-def (stmt, proc-list))) - 1,
call-conds (stmt),
t-cond-list),
bindings (top (ctrl-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list, mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state)), stmt, fetch-called-def (stmt, proc-list)))),
mg-psw (mg-state)),
n - 1)),
make-cond-list (fetch-called-def (stmt, proc-list))))),

MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
\[
\text{run})
\]

\[
\text{= p-state (tag ('pc,}
\]
\begin{align*}
\text{cons (subr,} \\
\text{length (code (cinfo))} \\
+ \text{length (push-locals-values-code (def-locals (fetch-called-def (stmt,} \\
\text{proc-list)))))} \\
+ \text{length (def-locals (fetch-called-def (stmt,} \\
\text{proc-list))})} \\
+ \text{length (call-actuals (stmt))} \\
+ \text{6} \\
+ \text{((index (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
\text{proc-list),} \\
\text{mg-list (mg-meaning (stmt,} \\
\text{proc-list),} \\
\text{mg-state n)}},} \\
\text{mg-state (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{stmt,} \\
\text{fetch-called-def (stmt,} \\
\text{proc-list))}} \\
\text{mg-psw (mg-state))},} \\
\text{def-conds (fetch-called-def (stmt,} \\
\text{proc-list))}) - 1) \\
\text{* 3} \\
+ \text{2}),}
\end{align*}
\]
\[
\text{ctrl-stk,}
\]
\[
\text{map-down-values (mg-alist (mg-meaning (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n)))},}
\]
\[
\text{bindings (top (ctrl-stk))},
\]
\[
\text{temp-stk),}
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
\text{list (cons ('c-c,}
\]
\begin{align*}
\text{put (mg-cond-to-p-nat (get (index (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
\text{proc-list),} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{n - 1))},} \\
\text{mg-psw (mg-state))},} \\
\text{def-conds (fetch-called-def (stmt,} \\
\text{proc-list))}) - 1) \\
\text{* 3} \\
+ \text{2}),}
\end{align*}
\]
\[
\text{ctrl-stk,}
\]
\[
\text{map-down-values (mg-alist (mg-meaning (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n)))},}
\]
\[
\text{bindings (top (ctrl-stk))},
\]
\[
\text{temp-stk),}
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
\text{list (cons ('c-c,}
\]
\begin{align*}
\text{put (mg-cond-to-p-nat (get (index (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
\text{proc-list),} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{n - 1))},} \\
\text{mg-psw (mg-state))},} \\
\text{def-conds (fetch-called-def (stmt,} \\
\text{proc-list))}) - 1) \\
\text{* 3} \\
+ \text{2}),}
\end{align*}
\]
\[
\text{ctrl-stk,}
\]
\[
\text{map-down-values (mg-alist (mg-meaning (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n)))},}
\]
\[
\text{bindings (top (ctrl-stk))},
\]
\[
\text{temp-stk),}
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
\text{list (cons ('c-c,}
\]
\begin{align*}
\text{put (mg-cond-to-p-nat (get (index (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
\text{proc-list),} \\
\text{mg-state (cc (mg-state),} \\
\text{make-call-var-alist (mg-alist (mg-state),} \\
\text{n - 1))},} \\
\text{mg-psw (mg-state))},} \\
\text{def-conds (fetch-called-def (stmt,} \\
\text{proc-list) n - 1)))),}
\end{align*}
\]
\[
\text{41}
Theorem: get-indexed-jump-instruction
\[ (k \not\equiv 0) \land (k < (1 + \text{length}(\text{call-conds}))) \]
\[ \rightarrow \quad (\text{get}(((k - 1) \ast 3) + 2, \]
\[ \quad \text{append}(\text{cond-conversion}(\text{call-conds}, 1 + \text{lc}, \text{cond-list}, \text{label-alist}), \]
\[ \quad \text{code})) \]
\[ = \quad \text{list}(\text{'jump}, \text{fetch-label}(\text{get}(k - 1, \text{call-conds}), \text{label-alist}))) \]

Theorem: convert-condition1-index-equivalence
\[ (\text{length}(\text{def-conds}) = \text{length}(\text{call-conds})) \land (x \in \text{def-conds}) \]
\[ \rightarrow \quad (\text{convert-condition1}(x, \text{def-conds}, \text{call-conds}) \]
\[ = \quad \text{get}(\text{index}(x, \text{def-conds}) - 1, \text{call-conds})) \]

Theorem: nonnormal-cond-conversion-not-normal
\[ (\text{'normal} \not\in \text{call-conds}) \]
\[ \rightarrow \quad (\text{convert-condition1}(\text{cc}, \text{def-conds}, \text{call-conds}) \neq \text{'normal}) \]

Theorem: call-state2-step8-effect-call-conds-body-equals-final
\[ ((n \not\equiv 0) \land (\neg \text{resources-inadequatep}(\text{stmt}, \]
\[ \quad \text{proc-list}, \]
\[ \quad \text{list}(\text{length}(\text{temp-stk}), \]
\[ \quad \text{p-ctrl-stk-size}(\text{ctrl-stk})))) \]
\[ \land \quad (\text{car}(\text{stmt}) = \text{'proc-call-mg}) \]
\[ \land \quad \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \]
\( \land \) ok-mg-def-plistp (proc-list)
\( \land \) ok-translation-parameters (cinfo, t-cond-list, stmt, proc-list, code2)
\( \land \) ok-mg-statep (mg-state, r-cond-list)
\( \land \) cond-subsetp (r-cond-list, t-cond-list)
\( \land \) (code (translate-def-body (assoc (subr, proc-list), proc-list))
\( \quad = \) append (code (translate (cinfo, t-cond-list, stmt, proc-list), code2))
\( \land \) user-defined-procp (subr, proc-list)
\( \land \) plistp (temp-stk)
\( \land \) listp (ctrl-stk)
\( \land \) mg-vars-list-ok-in-p-state (mg-alist (mg-state),
\( \quad \) bindings (top (ctrl-stk)),
\( \quad \) temp-stk)
\( \land \) no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
\( \land \) signatures-match (mg-alist (mg-state), name-alist)
\( \land \) normal (mg-state)
\( \land \) all-cars-unique (mg-alist (mg-state))
\( \land \) (\( \neg \) resource-errorp (mg-meaning-r (stmt,
\( \quad \) proc-list,
\( \quad \) mg-state,
\( \quad \) n,
\( \quad \) list (length (temp-stk),
\( \quad \) p-ctrl-stk-size (ctrl-stk)))))
\( \land \) (\( \neg \) normal (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
\( \quad \) proc-list,
\( \quad \) mg-state (cc (mg-state),
\( \quad \) make-call-var-alist (mg-alist (mg-state), stmt,
\( \quad \) fetch-called-def (stmt, proc-list),
\( \quad \) mg-psw (mg-state)),
\( \quad \) n − 1)))
\( \land \) (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
\( \quad \) proc-list,
\( \quad \) mg-state (cc (mg-state),
\( \quad \) make-call-var-alist (mg-alist (mg-state), stmt,
\( \quad \) fetch-called-def (stmt, proc-list),
\( \quad \) mg-psw (mg-state)),
\( \quad \) n − 1))
\( \neq \) `routineerror`
\( \land \) (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
\( \quad \) proc-list),
\( \quad \) mg-state (cc (mg-state),
\( \quad \) make-call-var-alist (mg-alist (mg-state), stmt,
\( \quad \) fetch-called-def (stmt, proc-list),
\( \quad \) mg-psw (mg-state)),
\( \quad \) n − 1))
\( \land \) (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
\( \quad \) proc-list),
\( \quad \) mg-state (cc (mg-state),
\( \quad \) make-call-var-alist (mg-alist (mg-state), stmt,
\( \quad \) fetch-called-def (stmt, proc-list),
\( \quad \) mg-psw (mg-state)),
\( \quad \) n − 1))
\( \neq \) `routineerror`
\[ \text{mg-state}((\text{cc } (\text{mg-state})), \text{make-call-var-alist}((\text{mg-alist } (\text{mg-state})), \text{stmt}, \text{fetch-called-def}((\text{stmt}, \text{proc-list})), \text{mg-psw}((\text{mg-state})), n - 1)) \in \text{def-conds}((\text{fetch-called-def}((\text{stmt}, \text{proc-list})))) \rightarrow (\text{p-step}((\text{p-state}(\text{tag } (\text{pc}, \text{cons } (\text{subr,}}
\text{length}(\text{code}(\text{cinfo}))) + \text{length}(\text{push-locals-values-code}(\text{def-locals}((\text{fetch-called-def}((\text{stmt}, \text{proc-list})))))) + \text{length}(\text{def-locals}((\text{fetch-called-def}((\text{stmt}, \text{proc-list})))))) + \text{length}(\text{call-actuals}(\text{stmt})) + 6 + ((\text{index}(\text{cc}((\text{mg-meaning}((\text{def-body}((\text{fetch-called-def}((\text{stmt, \text{proc-list}}))))), \text{proc-list}), \text{mg-state}((\text{cc}((\text{mg-state})), \text{make-call-var-alist}((\text{mg-alist } (\text{mg-state})), \text{stmt}, \text{fetch-called-def}((\text{stmt}, \text{proc-list}))), \text{mg-psw}((\text{mg-state})), n - 1)), \text{def-conds}((\text{fetch-called-def}((\text{stmt}, \text{proc-list})))) - 1) * 3) + 2)), \text{ctrl-stk}, \text{map-down-values}((\text{mg-alist } (\text{mg-meaning } (\text{stmt}, \text{proc-list}, \text{mg-state}, n)), \text{bindings}((\text{top}((\text{ctrl-stk})), \text{temp-stk})), \text{translate-proc-list}(\text{proc-list}), \text{list } (\text{cons } (\text{c-c,}}
\text{put}(\text{mg-cond-to-p-nat}(\text{get}(\text{index}(\text{cc}((\text{mg-meaning}((\text{def-body}((\text{fetch-called-def}((\text{stmt}, \text{proc-list}, \text{mg-state}((\text{cc}((\text{mg-state})))))))))}, \text{proc-list}), \text{mg-state}((\text{cc}((\text{mg-state}))))), n - 1))))
make-call-var-alist

mg-psw (mg-state)

$\text{def-conds (fetch-called-def (stmt, proc-list)}) - 1,$

call-conds (stmt),

t-cond-list),

0,

list (mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list),

proc-list, mg-state (cc mg-state),

make-call-var-alist (mg-state)

mg-psw (mg-state),

make-cond-list (fetch-called-def (stmt, proc-list))))))$,

MG-MAX-CTRL-STK-SIZE,

MG-MAX-TEMP-STK-SIZE,

MG-WORD-SIZE,

'rerun)

= 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),

45
Theorem: get-member-label-cnt-list

\( cc \in \text{lst} \)

\( \rightarrow \) \( \text{get} (\text{index} (cc, \text{lst}) - 1, \text{label-cnt-list} (lc, \text{length} (\text{lst}))) = lc \)

;; This is the case where the condition returned by the call is not in the
;; def-conds list. It must therefore be in the def-cond-locals list and we
;; return routine-error.
;;
;; (push-constant (list 'nat condition-index))

Theorem: call-state2-step6-effect-local-conds-body
((n ≠ 0) \\
\land (\neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))) \\
\land (\text{car (stmt)} = \text{proc-call-mg}) \\
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\land \text{ok-mg-def-plistp}(\text{proc-list}) \\
\land \text{ok-translation-parameters}(\text{cinfo, t-cond-list, stmt, proc-list, code2}) \\
\land \text{ok-mg-statetp}(\text{mg-state, r-cond-list}) \\
\land \text{cond-subsetp}(\text{r-cond-list, t-cond-list}) \\
\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))} = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)}) \\
\land \text{user-defined-procp}(\text{subr, proc-list}) \\
\land \text{plistp}(\text{temp-stk}) \\
\land \text{listp}(\text{ctrl-stk}) \\
\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk}) \\
\land \text{no-p-aliasing}(\text{bindings (top (ctrl-stk)), mg-alist (mg-state)}) \\
\land \text{signatures-match}(\text{mg-alist (mg-state), name-alist}) \\
\land \text{normal (mg-state)} \\
\land \text{all-cars-unique (mg-alist (mg-state))} \\
\land (\neg \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, n, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))) \\
\land (\neg \text{normal}(\text{mg-meaning}(\text{def-body (fetch-called-def (stmt, proc-list)), proc-list}, \text{mg-state (cc (mg-state)}, \text{make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list)), mg-psw (mg-state)}, n - 1)))) \\
\land (\text{cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list}, \text{mg-state (cc (mg-state)}, \text{make-call-var-alist (mg-alist (mg-state), stmt,}}
fetch-called-def (stmt, proc-list),
mg-psw (mg-state),

(n - 1) ≠ 'routineerror)
∧ (cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)),
proc-list,
mg-state (cc (mg-state),
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt, proc-list)),
mg-psw (mg-state)),

(n - 1)
∉ def-conds (fetch-called-def (stmt, proc-list)))
→ (p-step (p-state (tag ('pc, cons (subr,
find-label (get (untag (mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def
proc-list, stmt, proc-list,
make-call-var-alist (mg-alist (mg-state),
stmt,
fetch-called-def (stmt, proc-list),
mg-psw (mg-state)),

(n - 1)),
make-cond-list (fetch-called-def (stmt, proc-list),
label-cnt (cinfo),
cons (label-cnt (cinfo),
append (cond-case-jump-label-list (1 + label-cnt (cinfo),
1 + length (call-conds (stmt)),
label-cnt-list (label-cnt (cinfo),
length (def-cond-locals (fetch-called-def (stmt, proc-list),
append (code (translate (cinfo,
t-cond-list, stmt, proc-list)),

ctrl-stk,
map-down-values (mg-alist (mg-meaning (stmt, proc-list),

48
\[\begin{align*}
&mg\text{-state}, \\
&n)), \\
&\text{bindings (top (ctrl-stk)),} \\
&\text{temp-stk)}, \\
&\text{translate-proc-list (proc-list),} \\
&\text{list (list ('c-c,} \\
&m-g\text{-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
&\text{proc-list)),} \\
&m-g\text{-state (cc (mg-state),} \\
&m-g\text{-psw (mg-state),} \\
&n - 1)),} \\
&m-g\text{-make-cond-list (fetch-called-def (stmt,} \\
&\text{proc-list)))))),} \\
&\text{MG-MAX-CTRL-STK-SIZE,} \\
&\text{MG-MAX-TEMP-STK-SIZE,} \\
&\text{MG-WORD-SIZE,} \\
&\text{'run))} \\
&= p\text{-state (tag ('pc,} \\
&\text{cons (subr,} \\
&\text{length (code (cinfo))} \\
&+ \text{length (push-parameters-code (def-locals (fetch-called-def (stmt,} \\
&\text{proc-list)),} \\
&\text{call-actuals (stmt)))))} \\
&+ 4)),} \\
&\text{ctrl-stk,} \\
&\text{push ('nat 1,} \\
&m-g\text{-make-call-var-alist (mg-alist (mg-meaning (stmt,} \\
&\text{proc-list,} \\
&m-g\text{-state,} \\
&n))},} \\
&\text{bindings (top (ctrl-stk)),} \\
&\text{temp-stk))}, \\
&\text{translate-proc-list (proc-list),} \\
&\text{list (list ('c-c,} \\
&m-g\text{-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt,} \\
&\text{proc-list)),} \\
&m-g\text{-state (cc (mg-state),} \\
&m-g\text{-make-call-var-alist (mg-alist (mg-state}} \\
&\text{run)))}. \\
\end{align*}\]
Theorem: call-state2-step7-effect-local-conds-body

\((n \not\equiv 0)\)
\(\land\) \(\neg\) resources-inadequatep \((stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))\)
\(\land\) (car \((stmt)\) = 'proc-call-mg)
\(\land\) ok-mg-statement \((stmt, r-cond-list, name-alist, proc-list)\)
\(\land\) ok-mg-def-plistp \((proc-list)\)
\(\land\) ok-translation-parameters \((cinfo, t-cond-list, stmt, proc-list, code2)\)
\(\land\) ok-mg-statep \((mg-state, r-cond-list)\)
\(\land\) cond-subsetp \((r-cond-list, t-cond-list)\)
\(\land\) (code (translate-def-body (assoc \((subr, proc-list)\), proc-list))
\(\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\produce streamer:2023-10-15T08:53:16Z
\[\text{proc-list},\]
\[\text{mg-state (cc (mg-state)},\]
\[\text{make-call-var-alist (mg-alist (mg-state)},\]
\[\text{stmt},\]
\[\text{fetch-called-def (stmt},\]
\[\text{proc-list))},\]
\[\text{mg-psw (mg-state))},\]
\[n - 1)))\]
\[\land (cc (mg-meaning (def-body (fetch-called-def (stmt},\]
\[\text{proc-list)},\]
\[\text{proc-list},\]
\[\text{mg-state (cc (mg-state)},\]
\[\text{make-call-var-alist (mg-alist (mg-state)},\]
\[\text{stmt},\]
\[\text{fetch-called-def (stmt},\]
\[\text{proc-list))},\]
\[\text{mg-psw (mg-state)}),\]
\[n - 1))\]
\[\neq \ \text{routineerror})\]
\[\land (cc (mg-meaning (def-body (fetch-called-def (stmt},\]
\[\text{proc-list)},\]
\[\text{proc-list},\]
\[\text{mg-state (cc (mg-state)},\]
\[\text{make-call-var-alist (mg-alist (mg-state)},\]
\[\text{stmt},\]
\[\text{fetch-called-def (stmt},\]
\[\text{proc-list))},\]
\[\text{mg-psw (mg-state))},\]
\[n - 1))\]
\[\notin \ \text{def-conds (fetch-called-def (stmt},\]
\[\text{proc-list))))\]
\[\rightarrow (\text{p-step (p-state (tag ('pc},}\]
\[\text{cons (subr},\]
\[\text{length (code (cinfo)))}\]
\[+ \ \text{length (push-parameters-code (def-locals (fetch-called-def (stmt},\]
\[\text{proc-list))},\]
\[\text{call-actuals (stmt))})}\]
\[+ \ 4))},\]
\[\text{ctrl-stk},\]
\[\text{push ('(nat 1)},\]
\[\text{map-down-values (mg-alist (mg-meaning (stmt},\]
\[\text{proc-list},\]
\[\text{mg-state},\]
\[n))},\]
\[\text{bindings (top (ctrl-stk))},\]
\[\text{temp-stk))},\]
\[\text{translate-proc-list (proc-list)},\]
\[51\]
list (list ('c-c,
    mg-cond-to-p-nat (cc (mg-meaning (def-body (fetch-called-def (stmt,
                                        proc-list)),
                                           proc-list,
                                           mg-state (cc (mg-state),
                                           make-call-var-alist (mg-alist (mg-state,
                                                                           stmt,
                                                                           fetch-called-def
                                                                           mg-psw (mg-state),
                                                                           n - 1)),
               make-cond-list (fetch-called-def (stmt,
                                             proc-list))),)),
    mg-max-ctrl-stk-size,
    mg-max-temp-stk-size,
    mg-word-size,
    'run))
    =  p-state (tag ('pc,
         cons (subr,
            length (code (cinfo))
            +  length (push-locals-values-code (def-locals (fetch-called-def (stmt,
                                                   proc-list))))
            +  length (def-locals (fetch-called-def (stmt,
                                                      proc-list))))
            +  length (call-actuals (stmt))
            +  5)),
    ctrl-stk,
    map-down-values (mg-alist (mg-meaning (stmt,
                                           proc-list,
                                           mg-state,
                                           n)),
       bindings (top (ctrl-stk)),
    temp-stk),
    translate-proc-list (proc-list),
    '((c-c (nat 1))),
    MG-MAX-CTRL-STK-SIZE,
    MG-MAX-TEMP-STK-SIZE,
    MG-WORD-SIZE,
    'run))

THEOREM: call-state2-step8-effect-local-conds-body-equals-final
((n ≠ 0)
 ∧  (¬ resources-inadequatep (stmt,
                                      proc-list),
                                      52
\begin{align*}
\text{list} \left( \text{length} \left( \text{temp-stk} \right),
\text{p-ctrl-stk-size} \left( \text{ctrl-stk} \right) \right) \\
\land \left( \text{car} \left( \text{stmt} \right) = \text{'proc-call-mg} \right)
\land \text{ok-mg-statement} \left( \text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list} \right)
\land \text{ok-mg-def-plistp} \left( \text{proc-list} \right)
\land \text{ok-translation-parameters} \left( \text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list}, \text{code2} \right)
\land \text{ok-mg-statep} \left( \text{mg-state}, \text{r-cond-list} \right)
\land \text{cond-subsetp} \left( \text{r-cond-list}, \text{t-cond-list} \right)
\land \left( \text{code} \left( \text{translate-def-body} \left( \text{assoc} \left( \text{subr}, \text{proc-list} \right), \text{proc-list} \right) \right)
\right.
\left. = \text{append} \left( \text{code} \left( \text{translate} \left( \text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list} \right) \right),
\text{code2} \right) \right)
\land \text{user-defined-procp} \left( \text{subr}, \text{proc-list} \right)
\land \text{plistp} \left( \text{temp-stk} \right)
\land \text{listp} \left( \text{ctrl-stk} \right)
\land \text{mg-vars-list-ok-in-p-state} \left( \text{mg-alist} \left( \text{mg-state} \right),
\text{bindings} \left( \text{top} \left( \text{ctrl-stk} \right) \right),
\text{temp-stk} \right)
\land \text{no-p-aliasing} \left( \text{bindings} \left( \text{top} \left( \text{ctrl-stk} \right) \right), \text{mg-alist} \left( \text{mg-state} \right) \right)
\land \text{signatures-match} \left( \text{mg-alist} \left( \text{mg-state} \right), \text{name-alist} \right)
\land \text{normal} \left( \text{mg-state} \right)
\land \text{all-cars-unique} \left( \text{mg-alist} \left( \text{mg-state} \right) \right)
\land \left( \neg \text{resource-errorp} \left( \text{mg-meaning-r} \left( \text{stmt},
\text{proc-list},
\text{mg-state},
n,\right.
\text{list} \left( \text{length} \left( \text{temp-stk} \right),
\text{p-ctrl-stk-size} \left( \text{ctrl-stk} \right) \right) \right) \right)
\land \left( \neg \text{normal} \left( \text{mg-meaning} \left( \text{def-body} \left( \text{fetch-called-def} \left( \text{stmt}, \text{proc-list} \right) \right),
\text{proc-list},
\text{mg-state} \left( \text{cc} \left( \text{mg-state} \right) \right),
\text{make-call-var-alist} \left( \text{mg-alist} \left( \text{mg-state} \right),
\text{stmt},
\text{fetch-called-def} \left( \text{stmt}, \text{proc-list} \right) \right),
\text{mg-psw} \left( \text{mg-state} \right),
n - 1 \right) \right)
\land \left( \text{cc} \left( \text{mg-meaning} \left( \text{def-body} \left( \text{fetch-called-def} \left( \text{stmt}, \text{proc-list} \right) \right),
\text{proc-list},
\text{mg-state} \left( \text{cc} \left( \text{mg-state} \right) \right),
\text{make-call-var-alist} \left( \text{mg-alist} \left( \text{mg-state} \right),
\text{stmt},
\text{fetch-called-def} \left( \text{stmt}, \text{proc-list} \right) \right),
\text{mg-psw} \left( \text{mg-state} \right) \right),
53
\right)
\( n - 1 \rangle \neq \text{'routineerror} \\\n\land (\text{cc (mg-meaning (def-body (fetch-called-def (stmt, proc-list)), proc-list, mg-state (cc (mg-state), make-call-var-alist (mg-alist (mg-state), stmt, fetch-called-def (stmt, proc-list))), mg-psw (mg-state)), n - 1)) \notin \text{def-conds (fetch-called-def (stmt, proc-list)))} \rightarrow (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + length (push-locals-values-code (def-locals (fetch-called-def (stmt, proc-list))))) + length (def-locals (fetch-called-def (stmt, proc-list)))) + length (call-actuals (stmt)) + 5)), ctrl-stk, map-down-values (mg-alist (mg-meaning (stmt, proc-list, mg-state, n)), bindings (top (ctrl-stk)), temp-stk), translate-proc-list (proc-list), '((c-c (nat 1)))}, MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, 'run))= \text{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)),
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))

;; Time for the final proc-call lemma

THEOREM: call-exact-time-schema-normal-case
\[(\text{stmt-time} = (\text{locals-data-length} + \text{locals-length} + \text{actuals-length} + 1 + \text{body-time} + 5 + 1))\]
\[\land (p(\text{initial}, \text{locals-data-length} + \text{locals-length} + \text{actuals-length} + 1 + \text{body-time}) = \text{state2})\]
\[\land (p(\text{state2, 6}) = \text{final}))\]
\[\rightarrow (p(\text{initial, stmt-time}) = \text{final})\]

**Theorem:** call-exact-time-schema-nonnormal-case

\[(\text{stmt-time} = (\text{locals-data-length} + \text{locals-length} + \text{actuals-length} + 1 + \text{body-time} + 5 + 3))\]
\[\land (p(\text{initial}, \text{locals-data-length} + \text{locals-length} + \text{actuals-length} + 1 + \text{body-time} + 5 + 3))\]
\[\land (p(\text{state2, 8}) = \text{final}))\]
\[\rightarrow (p(\text{initial, stmt-time}) = \text{final})\]

(prove-lemma proc-call-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) ’proc-call-mg))

56
(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)
  (LIST (LENGTH TEMP-STK)
   (P-CTRL-STK-SIZE CTRL-STK))))
(IMPLIES
  (AND
   (OK-MG-STATEMENT (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
    (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
    (MAKE-NAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST))
    PROC-LIST)
   (OK-MG-DEF-PLISTP PROC-LIST)
   (OK-TRANSLATION-PARAMETERS
    (MAKE-CINFO NIL
     (CONS
      '(ROUTINEERROR . 0)
      (MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
       0))
     1)
    (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
    (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
    PROC-LIST
    (CONS
     '(DL 0 NIL (NO-OP))
     (CONS
      57
(LIST 'POP* (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
'(RET))

(OK-MG-STATEP (MAKE-CALL-ENVIRONMENT MG-STATE STMT
(FETCH-CALLED-DEF STMT PROC-LIST))
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(COND-SUBSETP (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(EQUAL
(CODE (TRANSLATE-DEF-BODY (ASSOC (CALL-NAME STMT) PROC-LIST)
PROC-LIST))
(APPEND
(CODE
(TRANSLATE
(MAKE-CINFO NIL
(CONS
'(ROUTINEERROR . 0)
(MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
0))
1)
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST))
(CONS
'(DL 0 NIL (NO-OP))
(CONS
(LIST 'POP* (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
'(RET))

(USER-DEFINED-PROCP (CALL-NAME STMT)
PROC-LIST)
(PLISTP
(APPEND
(REVERSE
(MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(MAP-DOWN-VALUES (MG-ALIST MG-STATE)
(BINDINGS (TOP CTRL-STK))
TEMP-STK))
)
(Listp
(Cons
(P-FRAME
(MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
STMT CTRL-STK TEMP-STK)
(TAG 'PC
(Cons SUBR
(ADD11
58
(PLUS (LENGTH (CODE CINFO)))
(DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
(LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
(LENGTH (CALL-ACTUALS STMT))))))
CTRL-STK))
(MG-VARS-LIST-OK-IN-P-STATE
 (MG-ALIST (MAKE-CALL-ENVIRONMENT MG-STATE STMT
 (FETCH-CALLED-DEF STMT PROC-LIST)))
 (BINDINGS
 (TOP
 (CONS
 (P-FRAME
 (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
 STMT CTRL-STK TEMP-STK)
 (TAG 'PC
 (CONS SUBR
 (ADD1
 (PLUS (LENGTH (CODE CINFO)))
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
 (LENGTH (CALL-ACTUALS STMT))))))
 CTRL-STK))
) (APPEND
 (REVERSE
 (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
 (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
 (BINDINGS (TOP CTRL-STK))
 TEMP-STK))
 (NO-P-ALIASING
 (BINDINGS
 (TOP
 (CONS
 (P-FRAME
 (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
 STMT CTRL-STK TEMP-STK)
 (TAG 'PC
 (CONS SUBR
 (ADD1
 (PLUS (LENGTH (CODE CINFO)))
 (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
 (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
 (LENGTH (CALL-ACTUALS STMT))))))
 CTRL-STK))
) (MG-ALIST (MAKE-CALL-ENVIRONMENT MG-STATE STMT

(FETCH-CALLED-DEF STMT PROC-LIST))
  (SIGNATURES-MATCH
    (MG-ALIST (MAKE-CALL-ENVIRONMENT MG-STATE STMT
  (FETCH-CALLED-DEF STMT PROC-LIST))
    (MAKE-NAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)))
  (NORMAL (MAKE-CALL-ENVIRONMENT MG-STATE STMT
  (FETCH-CALLED-DEF STMT PROC-LIST)))
  (ALL-CARS-UNIQUE
    (MG-ALIST (MAKE-CALL-ENVIRONMENT MG-STATE STMT
  (FETCH-CALLED-DEF STMT PROC-LIST)))
  (NOT
   (RESOURCE-ERRORP
    (MG-MEANING-R
     (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
     PROC-LIST
     (MAKE-CALL-ENVIRONMENT MG-STATE STMT
     (FETCH-CALLED-DEF STMT PROC-LIST))
     (SUB1 N)
     (LIST
      (LENGTH
       (APPEND
        (REVERSE
         (MG-TO-P-LOCAL-VALUES
          (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))))
        (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
          (BINDINGS (TOP CTRL-STK))
          TEMP-STK)))))
     (P-CTRL-STK-SIZE
      (CONS
       (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
          STMT CTRL-STK TEMP-STK)
        (TAG 'PC
        (CONS SUBR
        (ADD1
         (PLUS
          (LENGTH (CODE CINFO))
          (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))))
          (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
          (LENGTH (CALL-ACTUALS STMT))))))))
      (EQUAL
       (P
        (MAP-DOWN
         ;; state1

60
(MAKE-CALL-ENVIRONMENT MG-STATE STMT
   (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(CONS
  (P-FRAME
   (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
    STMT CTRL-STK TEMP-STK)
   (TAG 'PC
    (CONS SUBR (ADD1 (PLUS (LENGTH (CODE CINFO)))
      (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
      (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
      (LENGTH (CALL-ACTUALS STMT)))))
CTRL-STK)
(APPEND
 (REVERSE (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
   (BINDINGS (TOP CTRL-STK))
   TEMP-STK))
(TAG 'PC
 (CONS
 (CALL-NAME STMT)
 (LENGTH (CODE
 (MAKE-CINFO NIL
 (CONS
 ' (ROUTINEERROR . 0)
 (MAKE-LABEL-ALIST
   (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
   0))
  1))))
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
  (CLOCK (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
    PROC-LIST
    (MAKE-CALL-ENVIRONMENT MG-STATE STMT
     (FETCH-CALLED-DEF STMT PROC-LIST))
    (SUB1 N))
  (P-STATE ;; state2
   (TAG 'PC
    (CONS
     (CALL-NAME STMT)
     (IF
(NORMAL (MG-MEANING-R
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(MAKE-CALL-ENVIRONMENT MG-STATE STMT
  (FETCH-CALLED-DEF STMT PROC-LIST))
(SUB1 N)
LIST
.LENGTH
(APPEND
(REVERSE
  (MG-TO-P-LOCAL-VALUES
    (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(MAP-DOWN-VALUES (MG-ALIST MG-STATE)
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)))
(P-CTRL-STK-SIZE
(CONS
  (P-FRAME
    (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
      STMT CTRL-STK TEMP-STK)
    (TAG 'PC
      (CONS SUBR
        (ADD1
          (PLUS
            (LENGTH (CODE CINFO))
            (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
            (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
            (LENGTH (CALL-ACTUALS STMT))))
            CTRL-STK))))
  (LENGTH
    (CODE
      (TRANSLATE
        (MAKE-CINFO NIL
          (CONS
            '(ROUTINEERROR . 0)
            (MAKE-LABEL-ALIST
              (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
              0))
            1)
          (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
          (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
          PROC-LIST))))
          (FIND-LABEL
(FETCH-LABEL
(CC
(MG-MEANING-R
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(MAKE-CALL-ENVIRONMENT MG-STATE STMT
(FETCH-CALLED-DEF STMT PROC-LIST))
(SUB1 N)
(LIST
(LIGHT
(APPEND
(REVERSE
(MG-TO-P-LOCAL-VALUES
(DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
)(MAP-DOWN-VALUES (MG-ALIST MG-STATE)
(BINDINGS (TOP CTRL-STK))
TEMP-STK))))
(P-CTRL-STK-SIZE
(CONS
(P-FRAME
(MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
STMT CTRL-STK TEMP-STK)
(TAG 'PC
(CONS SUBR
(ADD1
(PLUS
(LIGHT (CODE CINFO))
(DATA-LENGTH
(DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
(LIGHT
(DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
(LIGHT
(CALL-ACTUALS STMT))))))
CTRL-STK))))
(LABEL-ALIST
(TRANSLATE
(MAKE-CINFO NIL
(CONS
'(ROUTINEERROR . 0)
(MAKE-LABEL-ALIST
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)
0))
1)
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
63
(PROC-LIST))
  (APPEND
   (CODE
    (TRANSLATE
     (MAKE-CINFO NIL
      (CONS
       '(ROUTINEERROR . 0)
        (MAKE-LABEL-ALIST
         (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
         0))
       1)
       (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
       (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
       PROC-LIST))
     (CONS
      '(DL 0 NIL (NO-OP))
      (CONS
       (LIST 'POP* (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
       '((RET)))))
     (CONS
      (P-FRAME
       (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
         STMT CTRL-STK TEMP-STK)
       (TAG 'PC
        (CONS SUBR
         (ADD1
          (PLUS (LENGTH (CODE CINFO))
            (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
          (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
          (LENGTH (CALL-ACTUALS STMT))))
       CTRL-STK)
       (MAP-DOWN-VALUES
        (MG-ALIST
         (MG-MEANING-R
          (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
          PROC-LIST
          (MAKE-CALL-ENVIRONMENT MG-STATE STMT
           (FETCH-CALLED-DEF STMT PROC-LIST))
           (SUB1 N)
           (LIST
            (LENGTH
             (APPEND
              (REVERSE
               (MG-TO-P-LOCAL-VALUES)"
(DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
(MAP-DOWN-VALUES (MG-ALIST MG-STATE)
  (BINDINGS (TOP CTRL-STK))
  TEMP-STK)))
(P-CTRL-STK-SIZE
(CONS
  (P-FRAME
    (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
      STMT CTRL-STK TEMP-STK)
    (TAG 'PC
      (CONS SUBR
        (ADD1
          (PLUS
            (LENGTH (CODE CINFO)))
            (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
            (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
            (LENGTH (CALL-ACTUALS STMT)))))))
  CTRL-STK))))
(BINDINGS
  (TOP
    (CONS
      (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
          STMT CTRL-STK TEMP-STK)
        (TAG 'PC
          (CONS SUBR
            (ADD1
              (PLUS (LENGTH (CODE CINFO)))
              (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
              (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
              (LENGTH (CALL-ACTUALS STMT)))))))
        CTRL-STK))))
  (APPEND
    (REVERSE
      (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
    (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
      (BINDINGS (TOP CTRL-STK))
      TEMP-STK))))
    (TRANSLATE-PROC-LIST PROC-LIST)
    (LIST
      (LIST 'C-C
        (MG-COND-TO-P-NAT
          (CC
            (MG-MEANING-R
(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
 (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 @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

67
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 @STMT-TIME
    (CLOCK STMT PROC-LIST MG-STATE N))
  (ADD-ABBREVIATION @BODY-TIME
    (CLOCK (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
      PROC-LIST
      (MAKE-CALL-ENVIRONMENT MG-STATE STMT
        (FETCH-CALLED-DEF STMT PROC-LIST))
      (SUB1 N)))
  (ADD-ABBREVIATION @STATE1
    (MAP-DOWN
      (MAKE-CALL-ENVIRONMENT MG-STATE STMT
        (FETCH-CALLED-DEF STMT PROC-LIST)))
    PROC-LIST
    (CONS
      (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
          STMT CTRL-STK TEMP-STK))
      (TAG 'PC
        (CONS SUBR
          (ADD1
            (PLUS (LENGTH (CODE CINFO))
              (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
              (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
              (LENGTH (CALL-ACTUALS STMT))))))))
CTRL-STK)
(APPEND
  (REVERSE
   (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
  (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
    (BINDINGS (TOP CTRL-STK))
    TEMP-STK))

(TAG 'PC
  (CONS
   (CALL-NAME STMT)
   (LENGTH
    (CODE
     (MAKE-CINFO NIL
      (CONS
       '(ROUTINEERROR . 0)
       (MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
        0))
      1)))))
  (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))

(ADD-ABBREVIATION @STATE2
  (P-STATE
   (TAG 'PC
    (CONS
     (CALL-NAME STMT)
     (IF
      (NORMAL
       (MG-MEANING-R
        (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
        PROC-LIST
        (MAKE-CALL-ENVIRONMENT MG-STATE STMT
         (FETCH-CALLED-DEF STMT PROC-LIST))
        (SUB1 N))
      (LIST
       (LENGTH
        (APPEND
         (REVERSE
          (MG-TO-P-LOCAL-VALUES
           (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
          (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
            (BINDINGS (TOP CTRL-STK))
            TEMP-STK)))))
      (P-CTRL-STK-SIZE
       (CONS
        (P-FRAME
         (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
         69
         1)))))

69
(TAG 'PC
(CONS SUBR
(ADD1
(PLUS
(LENGTH (CODE CINFO))
(DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(LENGTH (CALL-ACTUALS STMT))))))))
(LENGTH
(CODE
(TRANSLATE
(MAKE-CINFO NIL
(CONS
'((ROUTINEERROR . 0)
(MAKE-LABEL-ALIST
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
0))
1)
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST))
(FIND-LABEL
(FETCH-LABEL
(CC
(MG-MEANING-R
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(MAKE-CALL-ENVIRONMENT MG-STATE STMT
(FETCH-CALLED-DEF STMT PROC-LIST))
(SUB1 N)
(LIST
(LENGTH
(APPEND
(REVERSE
(MG-TO-P-LOCAL-VALUES
(DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(MAP-DOWN-VALUES (MG-ALIST MG-STATE)
(BINDINGS (TOP CTRL-STK))
TEMP-STK)))
(P-CTRL-STK-SIZE
(CONS
(P-FRAME

70
(MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
  STMT CTRL-STK TEMP-STK)

(TAG 'PC
  (CONS SUBR
    (ADD1
     (PLUS
      (LENGTH (CODE CINFO))
      (DATA-LENGTH
        (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
      (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
      (LENGTH (CALL-ACTUALS STMT))))
    CTRL-STK)))))

(LABEL-ALIST
  (TRANSLATE
    (MAKE-CINFO NIL
      (CONS
        '(ROUTINEERROR . 0)
         (MAKE-LABEL-ALIST
           (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
       0))
    1)

  (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))

  (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))

  PROC-LIST)))))

(APPEND
  (CODE
    (TRANSLATE
      (MAKE-CINFO NIL
        (CONS
          '(ROUTINEERROR . 0)
          (MAKE-LABEL-ALIST
            (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
         0))
      1)

    (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))

    (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))

    PROC-LIST))

    (CONS
      '(DL 0 NIL (NO-OP)))

    (CONS
      (LIST 'POP*
        (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
      ((RET)))))))

    (CONS
(P-FRAME
  (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
    STMT CTRL-STK TEMP-STK)
  (TAG 'PC
    (CONS SUBR
      (ADD1
        (PLUS (LENGTH (CODE CINFO))
          (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
          (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
          (LENGTH (CALL-ACTUALS STMT))))
      CTRL-STK))
  (MAP-DOWN-VALUES
    (MG-ALIST
      (MG-MEANING-R
        (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
      PROC-LIST
      (MAKE-CALL-ENVIRONMENT MG-STATE STMT
        (FETCH-CALLED-DEF STMT PROC-LIST))
      (SUB1 N)
      (LIST
        (LENGTH
          (APPEND
            (REVERSE
              (MG-TO-P-LOCAL-VALUES
                (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))))
            (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
              (BINDINGS (TOP CTRL-STK))
              TEMP-STK)))))
  (P-CTRL-STK-SIZE
    (CONS
      (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
          STMT CTRL-STK TEMP-STK)
        (TAG 'PC
          (CONS SUBR
            (ADD1
              (PLUS
                (LENGTH (CODE CINFO))
                (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
                (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
                (LENGTH (CALL-ACTUALS STMT))))
            CTRL-STK))))
      (BINDINGS
        (TOP
          72
          72)
(CONS
  (P-FRAME
   (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
     STMT CTRL-STK TEMP-STK))
  (TAG 'PC
   (CONS SUBR
     (ADD1
      (PLUS (LENGTH (CODE CINFO))
        (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
        (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
        (LENGTH (CALL-ACTUALS STMT))))
CTRL-STK)))
(APPEND
  (REVERSE
   (MG-TO-P-LOCAL-VALUES (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
  (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
    (BINDINGS (TOP CTRL-STK))
    TEMP-STK)))
(TRANSLATE-PROC-LIST PROC-LIST)
(LIST
  (LIST 'C-C
    (MG-COND-TO-P-NAT
     (CC
      (MG-MEANING-R
       (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
       PROC-LIST
       (MAKE-CALL-ENVIRONMENT MG-STATE STMT
         (FETCH-CALLED-DEF STMT PROC-LIST))
       (SUB1 N))
    (LIST
     (LENGTH
      (APPEND
       (REVERSE
        (MG-TO-P-LOCAL-VALUES
         (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
        (MAP-DOWN-VALUES (MG-ALIST MG-STATE)
          (BINDINGS (TOP CTRL-STK))
          TEMP-STK))))
     (P-CTRL-STK-SIZE
      (CONS
       (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
          STMT CTRL-STK TEMP-STK))
        (TAG 'PC

(CONS SUBR
  (ADD1
   (PLUS
    (LENGTH (CODE CINFO))
    (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
    (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
    (LENGTH (CALL-ACTUALS STMT)))))
  (CTRL-STK))))
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)))))
(MG-MAX-CTRL-STK-SIZE) (MG-MAX-TEMP-STK-SIZE) (MG-WORD-SIZE)
'RUN))
(ADD-ABBREVIATION @LOCALS-DATA-LENGTH
  (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(ADD-ABBREVIATION @LOCALS-LENGTH
  (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
(ADD-ABBREVIATION @ACTUALS-LENGTH
  (LENGTH (CALL-ACTUALS STMT)))
PROMOTE
(DEMOTE 19)
(DIVE 1 1)
PUSH TOP PROMOTE
(CLAIM (EQUAL (P @INITIAL
  (PLUS @LOCALS-DATA-LENGTH @LOCALS-LENGTH @ACTUALS-LENGTH
    1 @BODY-TIME))
    @STATE2)
  0)
(CLAIM
  (NORMAL
    (MG-MEANING-R
      (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
      PROC-LIST
      (MAKE-CALL-ENVIRONMENT MG-STATE STMT
        (FETCH-CALLED-DEF STMT PROC-LIST))
      (SUB1 N)
      (LIST
        (PLUS (LENGTH TEMP-STK)
          (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
        (PLUS (P-CTRL-STK-SIZE CTRL-STK)
          (PLUS 2
            (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST)))
            (LENGTH (DEF-FORMALS (FETCH-CALLED-DEF STMT
            PROC-LIST))))))))
  0)
(CLAIM (EQUAL @STMT-TIME
  74

74
(PLUS @LOCALS-DATA-LENGTH @LOCALS-LENGTH @ACTUALS-LENGTH 1
   @BODY-TIME 5 1))

0)
(CLAIM (EQUAL (P @STATE2 6) @FINAL) 0)
(DEMOTE 20 22 23)
(GENERALIZE ((@ACTUALS-LENGTH ACTUALS-LENGTH)
   (@LOCALS-LENGTH LOCALS-LENGTH)
   (@LOCALS-DATA-LENGTH LOCALS-DATA-LENGTH)
   (@STATE2 STATE2)
   (@STATE1 STATE1)
   (@BODY-TIME BODY-TIME)
   (@STMT-TIME STMT-TIME)
   (@FINAL FINAL)
   (@INITIAL INITIAL)))
DROP
(USE-LEMMA CALL-EXACT-TIME-SCHEMA-NORMAL-CASE)
DEMOTE
(S-PROP AND OR NOT IMPLIES FIX ZEROP IFF NLISTP)
(CONTRADICT 23)
(DIVE 1)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-0-UNWINDING-LEMMA)
(DIVE 1 1 1 1 1)
(REWRITE CALL-STATE2-STEP1-EFFECT)
UP
(REWRITE CALL-STATE2-STEP2-EFFECT)
UP
(REWRITE CALL-STATE2-STEP3-EFFECT)
UP
(REWRITE CALL-STATE2-STEP4-EFFECT)
UP
(REWRITE CALL-STATE2-STEP5-EFFECT)
UP
(REWRITE CALL-STATE2-STEP6-EFFECT-NORMAL-BODY-EQUALS-FINAL)
TOP S-PROP
(DEMOTE 21)
(DIVE 1 1)
(REWRITE MG-MEANING-EQUIVALENCE)
TOP S
(DEMOT 16)
DROP PROVE
(DIVE 1)
(REWRITE PROC-CALL-DOESNT-HALT2)
TOP S S
(CONTRADICT 22)
(DIVE 1)
X
(= (CAR STMT) 'PROC-CALL-MG 0)
S
(DIVE 2 2 2 2 2 2 1)
(= * T 0)
TOP DROP PROVE
(DEMOTE 21)
(DIVE 1 1)
(REWRITE MG-MEANING-EQUIVALENCE)
TOP S
(DIVE 1)
(REWRITE PROC-CALL-DOESNT-HALT2)
TOP S
(CLAIM (EQUAL @STMT-TIME
          (PLUS @LOCALS-DATA-LENGTH @LOCALS-LENGTH @ACTUALS-LENGTH 1
                @BODY-TIME 5 3))
      0)
(DEMOTE 21)
(DIVE 1 1 1)
(REWRITE MG-MEANING-EQUIVALENCE)
TOP PROMOTE
(CLAIM (EQUAL (P @STATE2 8) @FINAL) 0)
(DEMOTE 20 21 23)
(GENERALIZE ((@ACTUALS-LENGTH ACTUALS-LENGTH)
              (@LOCALS-LENGTH LOCALS-LENGTH)
              (@LOCALS-DATA-LENGTH LOCALS-DATA-LENGTH)
              (@STATE2 STATE2)
              (@STATE1 STATE1)
              (@BODY-TIME BODY-TIME)
              (@STMT-TIME STMT-TIME)
              (@FINAL FINAL)
              (@INITIAL INITIAL)))
DROP
(USE-LEMMA CALL-EXACT-TIME-SCHEMA-NONNORMAL-CASE)
DENOTE
(S-PROP AND OR NOT IMPLIES FIX ZEROP IFF NLISTP)
(CONTRADICT 23)
(DROP 19 20 21 23)
(DIVE 1)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-ADD1-3)
(REWRITE P-O-UNWINDING-LEMMA)
(DIVE 1 1 1 1 1 1 1)
(REWRITE CALL-STATE2-STEP1-EFFECT)
UP
(REWRITE CALL-STATE2-STEP2-EFFECT)
UP
(REWRITE CALL-STATE2-STEP3-EFFECT)
UP
(REWRITE CALL-STATE2-STEP4-EFFECT)
UP
(REWRITE CALL-STATE2-STEP5-EFFECT)
UP
(CLAIM
(EQUAL
(DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(MG-STATE (CC MG-STATE)
(MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
STMT
(FETCH-CALLED-DEF STMT PROC-LIST))
(MG-PSW MG-STATE))
(SUB1 N)))
'ROUTINEERROR)
O)
(REWRITE CALL-STATE2-STEP6-EFFECT-ROUTINEERROR-BODY)
UP
(REWRITE CALL-STATE2-STEP7-EFFECT-ROUTINEERROR-BODY)
UP
(REWRITE CALL-STATE2-STEP8-EFFECT-ROUTINEERROR-BODY-EQUALS-FINAL)
UP S-PROP
(CLAIM
(MEMBER

77
(CC
(MG-MEANING
 (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
PROC-LIST
(MG-STATE (CC MG-STATE)
 (MAKE-CALL-VAR-ALIST (MG-ALIST MG-STATE)
 STMT
 (FETCH-CALLED-DEF STMT PROC-LIST))
(MG-PSW MG-STATE))
 (SUB1 N)))
(DEF-CONDS (FETCH-CALLED-DEF STMT PROC-LIST)))
0)
(REWRITE CALL-STATE2-STEP6-EFFECT-CALL-CONDS-BODY)
UP
(REWRITE CALL-STATE2-STEP7-EFFECT-CALL-CONDS-BODY)
UP
(REWRITE CALL-STATE2-STEP8-EFFECT-CALL-CONDS-BODY-EQUALS-FINAL)
UP S-PROP PROVE PROVE
(DEMOTE 16 19)
DROP PROVE
(REWRITE CALL-STATE2-STEP6-EFFECT-LOCAL-CONDS-BODY)
UP
(REWRITE CALL-STATE2-STEP7-EFFECT-LOCAL-CONDS-BODY)
UP
(REWRITE CALL-STATE2-STEP8-EFFECT-LOCAL-CONDS-BODY-EQUALS-FINAL)
TOP S-PROP
(DEMOTE 16 19)
DROP PROVE
(DEMOTE 16 19)
DROP PROVE
(DEMOTE 16 19)
DROP PROVE
(DIVE 1)
(REWRITE PROC-CALL-DOESNT-HALT2)
TOP S
(CONTRADICT 22)
(DROP 19 20 22)
(DIVE 1)
X
(= (CAR STMT) 'PROC-CALL-MG 0)
S
(DIVE 2 2 2 2 2 2 1)
(= * F 0)
TOP S

78
Event: Make the library "c-proc-call2".
Index

all-cars-unique, 2–10, 17, 20, 24, 27, 29, 34, 38, 43, 47, 50, 53
all-pointers-bigger, 1
array-length, 3, 4
body-condition-member-make-cond-list, 31
body-condition-not-leave, 32
call-actuals, 6, 8, 9, 11, 18, 26–29, 36, 39, 41, 44, 49, 51, 52, 54
call-add1-lc-not-in-code, 20
call-condgs, 19, 21, 25, 33, 35, 37, 40, 42, 45, 48
call-condgs-index-lesssp, 33
call-def-cond-label-find-labelp, 33
call-exact-time-schema-nonnormalex-case, 56
call-exact-time-schema-normal-case, 56
call-lc-not-in-code, 23
call-state2-step5-effect, 17
call-state2-step6-effect-call-conds-body, 34
call-state2-step6-effect-local-conds-body, 47
call-state2-step6-effect-normal -body-equals-final, 20
call-state2-step6-effect-routinee-rror-body, 24
call-state2-step7-effect-call-conds-body, 38
call-state2-step7-effect-local-conds-body, 50
call-state2-step7-effect-routinee-rror-body, 26
call-state2-step8-effect-call-conds-body-equals-final, 42
call-state2-step8-effect-local-conds-body-equals-final, 52
call-state2-step8-effect-routinee-rror-body-equals-final, 28
code, 10, 17, 19–30, 34, 36, 38, 39, 41, 43–55
collect-pointers, 1
cond-case-jump-label-list, 19, 21, 25, 35, 48
cond-conversion, 33, 38, 42
cond-subsetp, 10, 17, 20, 24, 26, 28, 34, 38, 43, 47, 50, 53
convert-condition1, 42
convert-condition1-index-equivalence, 42
copy-out-params, 3, 4, 6
copy-out-params-restriction, 3
copy-out-params-restriction-cons, 3
data-length, 10, 17
data-param-lists-match, 5, 7
def-body, 6, 7, 9, 10, 18, 19, 21, 22, 24, 25, 27, 29, 31, 32, 34–37, 39–45, 47–54
def-cond-localgs, 19, 21, 25, 36, 48
def-condgs, 33–35, 37, 39–42, 44, 45, 48, 51, 54
def-formals, 6–9, 11
def-localgs, 7–11, 17, 18, 26–29, 36, 39, 41, 44, 49, 51, 52, 54
deposit-alist-value, 2, 4
deposit-array-value, 1, 3, 4
deposit-array-value-deposit-alist-value-commute2, 4
deposit-array-value-doesn’t-affected-map-down-values, 4
deposit-temp, 3, 4
deposit-temp-deposit-array-value
-compute6, 3

drop-formals-induction-hint, 1, 2
drop-locals-induction-hint, 2

extra-binding-doesnt-affect-copy-out-params, 4

fetch-called-def, 6–11, 17–19, 21, 22, 24–29, 31–45, 47–54
fetch-label, 22, 30, 42, 46, 55
find-def-conds-induction-hint, 32
find-def-conds-label1, 33
find-label, 19, 21, 23, 25, 30, 33, 36, 46, 48, 55
find-labelp, 20, 23, 33, 34
find-labelp-def-conds, 33
formal-types-preserved, 5, 6
formals-meaning-signature, 6

get, 19, 21, 25, 33, 36–38, 40, 42, 45, 46, 48
get-indexed-jump-instruction, 42
get-indexed-pop-global-instruction, 38
get-indexed-push-constant-instruction, 33
get-member-label-cnt-list, 46

index, 33, 34, 37, 40–42, 44–46

label-alist, 22, 30, 46, 55
label-cnt, 19–21, 23, 25, 34–36, 48
label-cnt-list, 19, 21, 25, 36, 46, 48
leave-not-in-make-cond-list, 32
length, 1, 3, 4, 8–11, 17–36, 38, 39, 41–55
listcars, 2–7, 9
locals-alist-mg-vars-ok, 8
locals-alist-mg-vars-ok0, 8
locals-meaning-signature, 7

make-call-param-alist, 6–8
make-call-var-alist, 6, 7, 9, 10, 18, 19, 21, 22, 24, 25, 27, 29, 31, 32, 35–37, 39–45, 47–54
make-cond-list, 18, 19, 21, 22, 25, 26, 31, 32, 35, 36, 38, 40, 42, 45, 48–50, 52
map-call-formals, 2, 3, 6–9, 11
map-call-locals, 2, 3, 8, 9, 11
map-down-copy-facts, 5
map-down-copy-facts2, 5
map-down-copy-facts3, 5
map-down-copy-out-params-induction-hint, 4, 5
map-down-copy-out-params-relation-new, 5
map-down-drop-locals-restriction, 2
map-down-values, 1–4, 6, 11, 18, 19, 22, 23, 25–28, 30, 36, 37, 40, 41, 44, 46, 49, 51, 52, 54, 55
map-down-values-drop-formals-restriction, 2
mg-alist, 6–11, 17–32, 34–55
mg-alistp, 1, 3, 4, 6, 7
mg-cond-to-p-nat, 18–23, 25, 26, 31, 33, 35–38, 40, 42, 45, 46, 48–50, 52, 55
mg-cond-to-p-nat-normal, 20
mg-cond-to-p-nat-routineerror, 23
mg-max-ctrl-stk-size, 18, 19, 22, 23, 26, 28, 30, 31, 36, 38, 40, 42, 45, 46, 49, 50, 52, 54, 55
mg-meaning, 7, 9, 11, 18, 19, 21, 22, 24–29, 31, 32, 35–37, 39–45, 47–54
mg-meaning-r, 10, 11, 17, 21–24, 27, 29–32, 34, 38, 43, 45–47, 50, 53–55
mg-psw, 6, 7, 9, 10, 18, 19, 21, 22,
24, 25, 27, 29, 31, 32, 35–37, 39–45, 47–54
mg-state, 6, 7, 9, 10, 18, 19, 21, 22, 24, 25, 27, 29, 31, 32, 35–37, 39–45, 47–54
mg-to-p-local-values, 8, 11
mg-vars-list-ok-in-p-state, 1–4, 6–8, 10, 17, 20, 24, 27, 29, 31, 32, 34, 38, 43, 47, 50, 53
mg-word-size, 18, 20, 22, 23, 26, 28, 30, 31, 36, 38, 40, 42, 45, 46, 49, 50, 52, 54, 55
no-duplicates, 2, 6
no-p-aliasing, 3, 4, 6, 9, 10, 17, 20, 24, 27, 29, 34, 38, 43, 47, 50, 53
no-p-aliasing-in-call-alists-ne w, 8
nonnormal-cond-conversion-not-normal, 42
normal, 10, 17, 20–22, 24, 27, 29–32, 34, 35, 38, 39, 43, 45, 47, 50, 51, 53, 54
ok-actual-params-list, 5, 7
ok-mg-def-plistp, 6–8, 10, 17, 20, 23, 24, 26, 28, 31–34, 38, 43, 47, 50, 53
ok-mg-formal-data-params-plistp, 6, 7
ok-mg-local-data-plistp, 8
ok-mg-statement, 6–8, 10, 17, 20, 23, 24, 26, 28, 31–34, 38, 42, 47, 50, 53
ok-mg-statep, 7, 8, 10, 17, 20, 24, 26, 28, 31–34, 38, 43, 47, 50, 53
ok-temp-stk-array-index, 1
ok-temp-stk-index, 1
ok-translation-parameters, 10, 17, 20, 23, 24, 26, 28, 33, 34, 38, 43, 47, 50, 53
p, 56
p-state, 18, 20, 22, 23, 26, 28, 30, 31, 36, 38, 41, 42, 45, 46, 49, 50, 52, 54, 55
p-step, 18, 22, 26, 28, 30, 36, 41, 45, 49, 52, 54
param-alist-mg-vars-ok, 7
param-alist-mg-vars-ok0, 7
plistp, 8, 10, 17, 20, 24, 27, 29, 34, 38, 43, 47, 50, 53
popn, 1, 11
popn-deposit-array-value, 1
popn-deposit-induction-hint, 1
popnlocals, 1
popn-locals1, 1
popn-rput, 1
push, 18, 26, 27, 37, 40, 49, 51
push-locals-values-code, 28, 29, 41, 44, 52, 54
push-parameters-code, 26, 27, 36, 39, 49, 51
put, 42, 45
resource-errorp, 10, 17, 21, 24, 27, 29, 31, 32, 34, 38, 43, 47, 50, 53
resources-inadequatep, 10, 17, 20, 24, 26, 28, 34, 38, 42, 47, 50, 53
restrict, 2, 3, 5–7, 9
restrict-cons, 3
ret-temp-stk-equals-final-temp-stk, 9
reverse, 8, 11
rput, 1
set-alist-value, 5
signatures-match, 7–10, 17, 20, 24, 27, 29, 31, 32, 34, 38, 43, 47, 50, 53
simple-mg-type-refp, 3, 4

82