Event: Start with the library "c-predefined4".

Theorem: fetch-label-0-case-2
\[ \text{cdr (assoc (cc, cons (',routineerror', 0), make-label-alist (lst, 0)))} = 0 \]

Event: Disable fetch-label-0-case-2.

Theorem: mg-cond-to-p-nat-p-objectp-type-nat
\[ ((\text{length (cond-list)} < ((\exp (2, \text{MG-WORD-SIZE}) - 1) - 1) - 1)) \land \ (p\text{-word-size (state)} = \text{MG-WORD-SIZE})) \rightarrow \ p\text{-objectp-type ('nat, mg-cond-to-p-nat (cc, cond-list), state}) \]

Event: Disable mg-cond-to-p-nat-p-objectp-type-nat.

Theorem: mg-cond-to-p-nat-index-lessp
\[ (\text{length (lst)} < n) \rightarrow ((\text{untag (mg-cond-to-p-nat (c, lst))} < (1 + (1 + n))) = \text{t}) \]

Event: Disable mg-cond-to-p-nat-index-lessp.

Theorem: set-alist-value-map-down-values-length-doesnt-shrink
\[ (\text{mg-vars-list-ok-in-p-state (x, bindings, temp-stk) } \land \text{mg-alistp (x)}) \rightarrow ((\text{length (map-down-values (set-alist-value (name, value, x), bindings, temp-stk))} < \text{length (map-down-values (x, bindings, temp-stk))}) = \text{f}) \]


Theorem: extra-bindings-doesnt-affect-formal-types-preserved
\[ (\text{car (x)} \not\in \text{listcars (y)}) \rightarrow (\text{formal-types-preserved (y, cons (x, z)}) = \text{formal-types-preserved (y, z)}) \]

Theorem: not-simple-identifiers-array-identifiers
\[ (\text{definedp (x, alist)} \land \text{mg-alistp (alist)} \land (\neg \text{simple-identifierp (x, alist)})) \rightarrow \text{array-identifierp (x, alist)} \]

Theorem: mg-meaning-preserves-signatures-match2
\[ \text{plistp (alist)} \]
→ signatures-match (alist,
  mg-alist (mg-meaning (stmt,
    proc-list,
    mg-state (cc, alist, psw),
    n)))))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; ;; PROCEDURE CALLS
;;; ;;
;;; ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

EVENT: Disable make-cond-list.

EVENT: Disable make-call-var-alist.

EVENT: Disable mg-var-ok-in-p-state.

EVENT: Enable proc-call-code.

THEOREM: proc-call-meaning-r-2
(car (stmt) = 'proc-call-mg)
→ (mg-meaning-r (stmt, proc-list, mg-state, n, sizes)
  = if n ≈ 0 then signal-system-error (mg-state, 'timed-out)
    elseif ¬ normal (mg-state) then mg-state
    elseif resources-inadequatep (stmt, proc-list, sizes)
    then signal-system-error (mg-state, 'resource-error)
    else map-call-effects (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)),
    n - 1,
    list (t-size (sizes)
      + data-length (def-locals (fetch-called-def (stmt,
        proc-list)))),
    c-size (sizes)
      + (2
        + length (def-locals (fetch-called-def (stmt,
          proc-list))))),
      proc-list),
    stmt,
    fetch-called-def (stmt,
    proc-list)),
    n - 1,
    list (t-size (sizes)
      + data-length (def-locals (fetch-called-def (stmt,
        proc-list)))),
    c-size (sizes)
      + (2
        + length (def-locals (fetch-called-def (stmt,
Theorem: call-translation-2
\( (\text{car}(\text{stmt}) = \text{proc-call-mg}) \rightarrow (\text{translate}(\text{cinfo}, \text{cond-list}, \text{stmt}, \text{proc-list}) = \text{make-cinfo}(\text{append}(\text{code}(\text{cinfo}), \text{proc-call-code}(\text{cinfo}, \text{stmt}, \text{cond-list}, \text{def-locals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})), \text{length}(\text{def-cond-locals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list}))))), \text{label-alist}(\text{cinfo}), \text{label-cnt}(\text{cinfo}) + (1 + (1 + \text{length}(\text{call-conds}(\text{stmt})))))) \)

Theorem: locals-pointers-bigger0
\( \text{all-cars-unique}(\text{locals}) \rightarrow \text{all-pointers-bigger}(\text{collect-pointers}(\text{map-call-locals}(\text{locals}, n), \text{locals}), n) \)

Theorem: no-p-aliasing-locals
\( ((n \in \mathbb{N}) \land \text{all-cars-unique}(\text{locals})) \rightarrow \text{no-p-aliasing}(\text{map-call-locals}(\text{locals}, n), \text{locals}) \)

Theorem: map-call-formals-all-pointers-smaller3
\( \text{ok-actual-params-list}(\text{actuals}, \text{mg-vars}) \land \text{data-param-lists-match}(\text{actuals}, \text{formals}, \text{mg-vars}) \land \text{all-cars-unique}(\text{formals}) \land \text{mg-alistp}(\text{mg-vars}) \land \text{mg-vars-list-ok-in-p-state}(\text{mg-vars}, \text{bindings}, \text{temp-stk}) \rightarrow \text{all-pointers-smaller}(\text{collect-pointers}(\text{map-call-formals}(\text{formals}, \text{actuals}, \text{bindings}), \text{make-call-param-alist}(\text{formals}, \text{actuals}, \text{mg-vars})), \text{length}(\text{temp-stk})) \)
Theorem: array-alist-element-lengths-match
\[(mg-alistp \(mg-vars\)) \land \text{definedp}(x, mg-vars) \land (\neg \text{simple-mg-type-refp}(\text{cadr}(\text{assoc}(x, mg-vars)))) \rightarrow \text{length}(\text{caddr}(\text{assoc}(x, mg-vars))) = \text{array-length}(\text{cadr}(\text{assoc}(x, mg-vars)))\]


Event: Enable no-p-aliasing.

Theorem: extra-bindings-dont-affect-no-p-aliasing
\[
\text{no-duplicates}(\text{listcars}(\text{append}(\text{bindings1}, \text{lst}))) \rightarrow \text{no-p-aliasing}(\text{append}(\text{bindings1}, \text{bindings2}, \text{lst})) = \text{no-p-aliasing}(\text{bindings2}, \text{lst})
\]

Theorem: extra-bindings-dont-affect-no-p-aliasing2
\[
\text{no-duplicates}(\text{listcars}(\text{append}(\text{bindings2}, \text{lst}))) \rightarrow \text{no-p-aliasing}(\text{append}(\text{bindings1}, \text{bindings2}, \text{lst})) = \text{no-p-aliasing}(\text{bindings1}, \text{lst})
\]

Theorem: actual-pointers-distinct
\[
\text{ok-actual-params-list}(\text{actuals}, mg-vars) \land \text{defined-identifierp}(x, mg-vars) \land \text{data-param-lists-match}(\text{actuals}, \text{formals}, mg-vars) \land \text{ok-mg-formal-data-params-plistp}(\text{formals}) \land \text{all-cars-unique}(mg-vars) \land \text{no-duplicates}(\text{cons}(x, \text{actuals})) \land \text{no-p-aliasing}(\text{bindings}, mg-vars) \land \text{all-cars-unique}(\text{formals}) \land mg-alistp(mg-vars) \rightarrow \not\in\text{untag}(\text{cdr}(\text{assoc}(x, \text{bindings})))\]
\[\not\in \text{collect-pointers}(\text{map-call-formals}(\text{formals}, \text{actuals}, \text{bindings}), \text{make-call-param-alist}(\text{formals}, \text{actuals}, mg-vars))\]

Theorem: actual-pointers-distinct2
\[
\text{ok-actual-params-list}(\text{actuals}, mg-vars) \land \text{defined-identifierp}(x, mg-vars) \land \text{data-param-lists-match}(\text{actuals}, \text{formals}, mg-vars)
\]
\[\begin{aligned}
&\wedge \text{ok-mg-formal-data-params-plistp}(\text{formals}) \\
&\wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-vars, bindings, temp-stk}) \\
&\wedge \text{all-cars-unique}(\text{mg-vars}) \\
&\wedge \text{no-duplicates(cons}(x, \text{actuals})) \\
&\wedge \text{no-p-aliasing}(\text{bindings, mg-vars}) \\
&\wedge \text{all-cars-unique}(\text{formals}) \\
&\wedge \text{mg-alistp}(\text{mg-vars}) \\
&\wedge (\neg\text{simple-mg-type-refp(cadr(assoc}(x, \text{mg-vars})))) \\
\rightarrow \text{disjoint(n-successive-pointers(cdr(assoc}(x, \text{bindings})),} \\
&\text{array-length(cadr(assoc}(x, \text{mg-vars})))) \\
&\text{collect-pointers(map-call-formals(\text{formals, actuals, bindings}),} \\
&\text{make-call-param-alist(\text{formals, actuals, mg-vars}))}
\end{aligned}\]

**Theorem: no-p-aliasing-formals**

\[
\text{ok-actual-params-list(\text{actuals, mg-vars})} \\
\wedge \text{data-param-lists-match(\text{actuals, formals, mg-vars})} \\
\wedge \text{ok-mg-formal-data-params-plistp(\text{formals})} \\
\wedge \text{mg-vars-list-ok-in-p-state(\text{mg-vars, bindings, temp-stk})} \\
\wedge \text{all-cars-unique(\text{mg-vars})} \\
\wedge \text{no-duplicates(\text{actuals})} \\
\wedge \text{no-p-aliasing}(\text{bindings, mg-vars}) \\
\wedge \text{all-cars-unique(\text{formals})} \\
\wedge \text{mg-alistp(\text{mg-vars})} \\
\rightarrow \text{no-p-aliasing(map-call-formals(\text{formals, actuals, bindings}),} \\
&\text{make-call-param-alist(\text{formals, actuals, mg-vars}))}
\]

**Theorem: actual-params-list-ok-in-mg-alist**

\[
((\text{car}(\text{stmt}) = '\text{proc-call-mg}) \\
\wedge \text{ok-mg-statement(\text{stmt, r-cond-list, name-alist, proc-list})} \\
\wedge \text{ok-mg-statep(\text{mg-state, r-cond-list})} \\
\wedge \text{signatures-match(\text{mg-alist}(\text{mg-state}), \text{name-alist})} \\
\rightarrow \text{ok-actual-params-list(\text{call-actuals(\text{stmt}), mg-alist(\text{mg-state})})}
\]

**Theorem: data-param-lists-match-in-mg-alist**

\[
((\text{car}(\text{stmt}) = '\text{proc-call-mg}) \\
\wedge \text{ok-mg-statement(\text{stmt, r-cond-list, name-alist, proc-list})} \\
\wedge \text{ok-mg-statep(\text{mg-state, r-cond-list})} \\
\wedge \text{signatures-match(\text{mg-alist}(\text{mg-state}), \text{name-alist})} \\
\rightarrow \text{data-param-lists-match(\text{call-actuals(\text{stmt}),} \\
&\text{def-formals(fetch-called-def(\text{stmt, proc-list})),} \\
&\text{mg-alist(\text{mg-state})})}
\]

**Theorem: call-local-names-unique**
[((\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}))
\rightarrow \text{no-duplicates}(\text{append}(\text{listcars}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list}))),
\hspace{1cm}\text{listcars}(\text{def-formals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))))

\textbf{Theorem: no-p-aliasing-in-call-environment}

\begin{align*}
\text{((\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{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}),
\hspace{1cm}\text{bindings}(\text{top}(\text{ctrl-stk})),
\hspace{1cm}\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{all-cars-unique}(\text{mg-alist}(\text{mg-state})) & \\
\rightarrow \text{no-p-aliasing}(\text{make-frame-alist}(\text{fetch-called-def}(\text{stmt}, \text{proc-list}),
\hspace{1cm}\text{stmt},
\hspace{1cm}\text{ctrl-stk},
\hspace{1cm}\text{temp-stk}),
\hspace{1cm}\text{make-call-var-alist}(\text{mg-alist}(\text{mg-state}),
\hspace{1cm}\text{stmt},
\hspace{1cm}\text{fetch-called-def}(\text{stmt}, \text{proc-list}))))
\end{align*}

\textbf{Theorem: call-exact-time-translation-parameters-ok}

\begin{align*}
\text{((\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}) & \\
\rightarrow \text{ok-translation-parameters}(\text{make-cinfo}(\text{nil},
\hspace{1cm}\text{cons}(\text{'routineerror}, 0),
\hspace{1cm}\text{make-label-alist}(\text{make-cond-list}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})),
\hspace{1cm}0)),
\hspace{1cm}1),
\hspace{1cm}\text{make-cond-list}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})),
\hspace{1cm}0))
\end{align*}
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)))))

THEOREM: call-body-rewrite
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))))

DEFINITION:
mg-locals-list-ok-induction-hint (locals, temp-stk)
= if locals ≃ nil then t
  elseif simple-mg-type-refp (cadar (locals))
    then mg-locals-list-ok-induction-hint (cdr (locals),
      cons (mg-to-p-simple-literal (caddar (locals)),
      temp-stk))
  else mg-locals-list-ok-induction-hint (cdr (locals),
    append (reverse (mg-to-p-simple-literal-list (caddar (locals))),
      temp-stk)) endif

THEOREM: definedp-caar
definedp (x, cons (cons (x, y), z))

THEOREM: tag-length-plistp-2
length-plistp (tag (x, y), 2)

THEOREM: mg-locals-list-ok-in-call-environment
(all-cars-unique (locals) ∧ ok-mg-local-data-plistp (locals))
→ mg-vars-list-ok-in-p-state (locals,
append (map-call-locals (locals,
    length (temp-stk)),
    lst),
append (reverse (mg-to-p-local-values (locals)),
    temp-stk))

**Theorem**: simple-formal-ok-for-actual2
(definedp (actual, mg-alist)
 ∧ data-params-match (actual, formal, mg-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist, bindings, temp-stk)
 ∧ simple-mg-type-refp (cadr (formal))) → ok-temp-stk-index (cdr (assoc (actual, bindings)), temp-stk)

**Definition**:
mg-formals-list-ok-induction-hint (formals, actuals) = if formals ≃ nil then t
else mg-formals-list-ok-induction-hint (cdr (formals),
cdr (actuals)) endif

**Theorem**: array-formal-ok-for-actual
(defined-identifierp (actual, mg-alist)
 ∧ data-params-match (actual, formal, mg-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist, bindings, temp-stk)
 ∧ (∼ simple-mg-type-refp (cadr (formal)))) → ok-temp-stk-array-index (cdr (assoc (actual, bindings)),
    append (local-values, temp-stk),
    array-length (cadr (formal)))

**Theorem**: mg-formals-list-ok-in-call-environment0
(all-cars-unique (formals)
 ∧ ok-actual-params-list (actuals, mg-alist)
 ∧ data-param-lists-match (actuals, formals, mg-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist, bindings, temp-stk)) → mg-vars-list-ok-in-p-state (make-call-param-alist (formals, actuals, mg-alist),
    map-call-formals (formals, actuals, bindings),
    append (local-values, temp-stk))

**Theorem**: mg-formals-list-ok-in-call-environment1
(all-cars-unique (formals)
 ∧ mg-alistp (mg-alist)
 ∧ ok-actual-params-list (actuals, mg-alist)
 ∧ ok-mg-formal-data-params-plistp (formals)
 ∧ data-param-lists-match (actuals, formals, mg-alist)
\[ \begin{align*}
\land & \quad \text{mg-vars-list-ok-in-p-state} (\text{mg-alist}, \text{bindings}, \text{temp-stk}) \\
\rightarrow & \quad \text{mg-vars-list-ok-in-p-state} (\text{make-call-param-alist} (\text{formals}, \text{actuals}, \text{mg-alist}), \\
& \quad \text{map-call-formals} (\text{formals}, \text{actuals}, \text{bindings}), \\
& \quad \text{append} (\text{local-values}, \\
& \quad \text{map-down-values} (\text{mg-alist}, \text{bindings}, \text{temp-stk})))
\end{align*} \]

\textbf{THEOREM:} \( \text{mg-vars-list-ok-in-call-environment} \)
\(( n \not\approx 0 )\)
\land \quad (\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 \quad (\text{car} (\text{stmt}) = \text{'proc-call-mg})
\land \quad \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list, name-alist, proc-list})
\land \quad \text{ok-mg-def-plistp} (\text{proc-list})
\land \quad \text{ok-translation-parameters} (\text{cinfo, t-cond-list, stmt, proc-list, code2})
\land \quad \text{ok-mg-statep} (\text{mg-state, r-cond-list})
\land \quad \text{cond-subsetp} (\text{r-cond-list}, \text{t-cond-list})
\land \quad (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list}))) = \\
& \quad \text{append} (\text{code} (\text{translate} (\text{cinfo, t-cond-list, stmt, proc-list})))
\land \quad \text{user-defined-procp} (\text{subr, proc-list})
\land \quad \text{plistp} (\text{temp-stk})
\land \quad \text{listp} (\text{ctrl-stk})
\land \quad \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 \quad \text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state}))
\land \quad \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist})
\land \quad \text{normal} (\text{mg-state})
\land \quad \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})))
\rightarrow \quad \text{mg-vars-list-ok-in-p-state} (\text{make-call-var-alist} (\text{mg-alist} (\text{mg-state}), \text{stmt, fetch-called-def} (\text{stmt, proc-list})), \\
& \quad \text{append} (\text{map-call-locals} (\text{def-locals} (\text{fetch-called-def} (\text{stmt, proc-list}))), \\
& \quad \text{length} (\text{temp-stk}), \\
& \quad \text{map-call-formals} (\text{def-formals} (\text{fetch-called-def} (\text{stmt, proc-list})))
\end{align*} \]
Theorem: proc-call-doesnt-halt

\((\text{car}(\text{stmt}) = \text{'proc-call-mg}) \land \text{normal}(\text{mg-state}) \land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \land \text{ok-mg-def-plistp}(\text{proc-list}) \land \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list}) \land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk}) \land \neg \text{resource-errorp}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}, \text{mg-state}, n, \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk})))))\)

\(\rightarrow \neg \text{resource-errorp}(\text{mg-meaning-r}(\text{def-body}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})), \text{proc-list}), \text{make-call-environment}(\text{mg-state}, \text{stmt}, \text{fetch-called-def}(\text{stmt}, \text{proc-list})), n - 1, \text{list}(\text{length}(\text{append}(\text{reverse}(\text{mg-to-p-local-values}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \text{proc-list})))), \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk}))))\)
Theorem: proc-call-doesnt-halt2

\((\text{car}\,(\text{stmt}) = \ \text{proc-call-mg}) \land \ \text{normal}\,(\text{mg-state}) \land \ \text{ok-mg-statement}\,(\text{stmt},\ r\text{-cond-list},\ \text{name-alist},\ \text{proc-list}) \land \ \text{ok-mg-def-plistp}\,(\text{proc-list}) \land \ \text{ok-mg-statep}\,(\text{mg-state},\ r\text{-cond-list}) \land \ \text{mg-vars-list-ok-in-p-state}\,(\text{mg-alist}\,(\text{mg-state}),\ \text{bindings}\,(\text{top}\, (\text{ctrl-stk})),\ \text{temp-stk}) \land \ (\neg \ \text{resource-errorp}\,(\text{mg-meaning-r}\,(\text{stmt},\ \text{proc-list},\ \text{mg-state},\ n,\ \text{list}\,(\text{length}\,(\text{temp-stk}),\ \text{p-ctrl-stk-size}\,(\text{ctrl-stk})))))) \rightarrow (\neg \ \text{resource-errorp}\,(\text{mg-meaning-r}\,(\text{def-body}\,(\text{fetch-called-def}\,(\text{stmt},\ \text{proc-list})),\ \text{proc-list}),\ \text{make-call-environment}\,(\text{mg-state},\ \text{stmt},\ \text{fetch-called-def}\,(\text{stmt},\ \text{proc-list})),\ n - 1,\ \text{list}\,(\text{length}\,(\text{temp-stk})) + \ \text{data-length}\,(\text{def-locals}\,(\text{fetch-called-def}\,(\text{stmt},\ \text{proc-list})))) + \ \text{p-ctrl-stk-size}\,(\text{ctrl-stk}) + 2 + \ \text{length}\,(\text{def-locals}\,(\text{fetch-called-def}\,(\text{stmt},\ \text{proc-list})))) + \ \text{length}\,(\text{def-formals}\,(\text{fetch-called-def}\,(\text{stmt},\ \text{proc-list})))))))))\)

Theorem: proc-call-exact-time-hyps

\((n \neq 0) \land (\neg \ \text{resources-inadequatep}\,(\text{stmt},\ \text{proc-list}))\)
\[(\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{ok-mg-statep (mg-state, r-cond-list}) \land \text{cond-subsetp (r-cond-list, t-cond-list}) \land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list})}) = \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, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}) \rightarrow (\text{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}) \land \text{ok-translation-parameters (make-cinfo (nil, cons ('routineerror , 0), make-label-alist (make-cond-list (fetch-called-def (stmt, proc-list, proc-list), 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)), nil))} \land \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))})}
\[
\text{cons (list ('pop*, data-length (def-locals (fetch-called-def (stmt, proc-list)))), '((ret))))}
\]

∧ 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)))
∧ (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)),
    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, 1 + (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)))))))

∧ 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, 1 + (length (code (cinfo)) + data-length (def-locals (k)) + length (def-locals (fetch-called-def (stmt, proc-list))) + length (call-actuals (stmt))))),

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, 1 + (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)))))),

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))))
∧ (¬ 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))), n − 1, 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, 1 + (length (code (cinfo) + data-length) + length (def-locals (fetch-called-def (stmt, proc-list)) + length (call-actuals (stmt))))))))))
THEOREM: simple-typed.literal-listp
simple-typed.literalp (exp, type) → listp (mg-to-p-simple.literal (exp))

DEFINITION:
push-array-value-induction-hint (array-value, code, temp-stk)
= if array-value ≃ nil then t
else push-array-value-induction-hint (cdr (array-value),
append (code,
list (list (push-constant,
mg-to-p-simple.literal (car (array-value)))))
push (mg-to-p-simple.literal (car (array-value)),
temp-stk)) endif

THEOREM: push-local-array-values-code-effect
((length (temp-stk) + length (array-value)) < MG-MAX-TEMP-STK-SIZE)
∧ simple-mg-type-refp (array-elemtype)
∧ simple-typed.literal-plistp (array-value, array-elemtype)
∧ (cdddr (assoc (subr, proc-list-code))
= append (code,
append (push-local-array-values-code (array-value),
code2))))
→ (p (p-state (tag ('pc, cons (subr, length (code))),
ctrl-stk,
temp-stk,
proc-list-code,
data-segment,
max-ctrl,
MG-MAX-TEMP-STK-SIZE,
word-size,
'run),
length (array-value))
= p-state (tag ('pc,
cons (subr, length (code) + length (array-value))),
ctrl-stk,
append (reverse (mg-to-p-simple.literal-list (array-value)),
temp-stk),
proc-list-code,
data-segment,
max-ctrl,
MG-MAX-TEMP-STK-SIZE,
word-size,
'run))

EVENT: Enable length-cons.
**Definition:**
\[
\textit{locals-values-induction-hint} (\textit{locals}, \textit{code}, \textit{temp-stk}) = \begin{cases} 
\text{true} & \text{if } \textit{locals} \simeq \text{nil} \\
\text{false} & \text{else if } \text{simple-mg-type-refp (cadr (car (\textit{locals})))} \\
\text{true} & \text{then } \textit{locals-values-induction-hint} (\text{cdr (\textit{locals})}, \\
& \text{append (\textit{code},} \\
& \quad \text{list (list (\textit{\textquoteleft push-constant,} \\
& \quad \quad \textit{mg-to-p-simple-literal (caddr (\textit{locals}))}), \\
& \quad \text{push (mg-to-p-simple-literal (caddr (car (\textit{locals}))}, \\
& \quad \textit{temp-stk}))} \\
\text{else } \textit{locals-values-induction-hint} (\text{cdr (\textit{locals})}, \\
& \text{append (\textit{code},} \\
& \quad \text{push-local-array-values-code (caddr (car (\textit{locals}))}, \\
& \quad \text{append (reverse (mg-to-p-simple-literal-list (caddr (car (\textit{locals})))),} \\
& \quad \textit{temp-stk})) \end{cases}
\]

**Theorem:** call-push-locals-values-effect
\[
((\text{length (\textit{temp-stk}) + data-length (\textit{locals}) < MG-MAX-TEMP-STK-SIZE}) \\
\land \text{ok-mg-local-data-plistp (\textit{locals})} \\
\land (\text{cdddr (assoc (\textit{subr}, proc-list-code))} \\
\quad = \text{append (\textit{code}, append (push-locals-values-code (\textit{locals}), code2))))} \\
\rightarrow (\text{p (p-state (tag (\textquoteleft pc, cons (\textit{subr}, length (\textit{code}))},} \\
& \textit{ctrl-stk}, \\
& \textit{temp-stk}, \\
& \textit{proc-list-code}, \\
& \textit{data-segment}, \\
& \textit{max-ctrl}, \\
& \text{MG-MAX-TEMP-STK-SIZE}, \\
& \textit{word-size}, \\
& \textquoteleft run),} \\
& \text{data-length (\textit{locals})} \\
\quad = \text{p-state (tag (\textquoteleft pc,} \\
& \quad \text{cons (\textit{subr}, data-length (\textit{locals}) + length (\textit{code}))},} \\
& \textit{ctrl-stk}, \\
& \text{append (reverse (mg-to-p-local-values (\textit{locals})), temp-stk),} \\
& \textit{proc-list-code}, \\
& \textit{data-segment}, \\
& \textit{max-ctrl}, \\
& \text{MG-MAX-TEMP-STK-SIZE}, \\
& \textit{word-size}, \\
& \textquoteleft run))}
\]

**Definition:**
\[
\textit{locals-addresses-induction-hint} (\textit{locals}, \textit{code}, \textit{temp-stk}, n)
\]
Theorem: call-push-locals-addresses-effect
\[ (\text{length} (\text{temp-stk}) + \text{length} (\text{locals})) < \text{MG-MAX-TEMP-STK-SIZE})
\land (\text{cddr} (\text{assoc} (\text{subr}, \text{proc-list-code})))
\land (n < \text{length} (\text{temp-stk}))
\land (n \neq (\text{data-length} (\text{locals}) - 1))
\rightarrow (\text{p} (\text{p-state} (\text{'pc}, \text{cons} (\text{subr}, \text{length} (\text{code}))))),
\text{ctrl-stk},
\text{temp-stk},
\text{proc-list-code},
\text{data-segment},
\text{max-ctrl},
\text{MG-MAX-TEMP-STK-SIZE},
\text{word-size},
\text{'run},
\text{length} (\text{locals}))
= \text{p-state} (\text{'pc}, \text{cons} (\text{subr}, \text{length} (\text{code}) + \text{length} (\text{locals}))),
\text{ctrl-stk},
\text{append} (\text{reverse} (\text{ascending-local-address-sequence} (\text{locals},
\text{length} (\text{temp-stk}) - n - 1)),
\text{temp-stk}),
\text{length} (\text{locals}))
\begin{align*}
\text{proc-list-code,} \\
\text{data-segment,} \\
\text{max-ctrl,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{word-size,} \\
\text{'run})
\end{align*}

**Definition:**
call-push-actuals-induction-hint \((\text{actuals, code, temp-stk, ctrl-stk})\) 
\[= \begin{cases} 
\text{if } \text{actuals} \simeq \text{nil} \text{ then } \text{t} \\
\text{else call-push-actuals-induction-hint (cdr (actuals),} \\
\text{append (code,} \\
\text{list (list ('push-local,} \\
\text{car (actuals))),} \\
\text{cons (cdr (assoc (car (actuals),} \\
\text{bindings (top (ctrl-stk)))))),} \\
\text{temp-stk),} \\
\text{ctrl-stk) endif} 
\end{cases} \]

**Theorem:** call-push-actuals-effect
\[\begin{align*}
&((\text{length (temp-stk)} + \text{length (actuals)}) < \text{MG-MAX-TEMP-STK-SIZE}) \\
&\wedge (\text{cddr (assoc (subr, proc-list-code))} \\
&= \text{append (code, append (push-actuals-code (actuals), code2)))} \\
\rightarrow (\text{p (p-state (tag ('pc, cons (subr, length (code))),} \\
\text{ctrl-stk,} \\
\text{temp-stk,} \\
\text{proc-list-code,} \\
\text{data-segment,} \\
\text{max-ctrl,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{word-size,} \\
\text{'run),} \\
\text{length (actuals))} \\
&= \text{p-state (tag ('pc, cons (subr, length (code) + length (actuals))),} \\
\text{ctrl-stk,} \\
\text{append (reverse (mg-actuals-to-p-actuals (actuals,} \\
\text{bindings (top (ctrl-stk)))))),} \\
\text{temp-stk),} \\
\text{proc-list-code,} \\
\text{data-segment,} \\
\text{max-ctrl,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{word-size,} \\
\text{'run))} 
\end{align*} \]
THEOREM: call-push-parameters-effect1

\[\begin{align*}
&(((\text{length}(\text{temp-stk})
\quad + \text{data-length}(\text{locals})
\quad + \text{length}(\text{locals})
\quad + \text{length}(\text{actuals}))
\quad < \text{MG-MAX-TEMP-STK-SIZE})
\land \text{ok-mg-local-data-plistp}(\text{locals})
\land (\text{cdddr}(\text{assoc}(\text{subr}, \text{proc-list-code}))
\quad = \text{append}(\text{code},
\quad \text{append}(\text{push-parameters-code}(\text{locals}, \text{actuals}), \text{code2}))))
\rightarrow (\text{p}(\text{p-state}(\text{tag}(\text{'pc}, \text{cons}(\text{subr}, \text{length}(\text{code}))))),
\quad \text{ctrl-stk},
\quad \text{temp-stk},
\quad \text{proc-list-code},
\quad \text{data-segment},
\quad \text{max-ctrl},
\quad \text{MG-MAX-TEMP-STK-SIZE},
\quad \text{word-size},
\quad \text{'run}),
\quad \text{data-length}(\text{locals}) + \text{length}(\text{locals}) + \text{length}(\text{actuals}))
\quad = \text{p-state}(\text{tag}(\text{'pc}, \text{cons}(\text{subr}, \text{length}(\text{code})),
\quad + \text{data-length}(\text{locals})
\quad + \text{length}(\text{locals})
\quad + \text{length}(\text{actuals}))),
\quad \text{ctrl-stk},
\quad \text{append}(\text{reverse}(\text{mg-actuals-to-p-actuals}(\text{actuals},
\quad \text{bindings}(\text{top}(\text{ctrl-stk})))),
\quad \text{append}(\text{reverse}(\text{ascending-local-address-sequence}(\text{locals},
\quad \text{length}(\text{temp-stk})))),
\quad \text{append}(\text{reverse}(\text{mg-to-p-local-values}(\text{locals})),
\quad \text{temp-stk})),
\quad \text{proc-list-code},
\quad \text{data-segment},
\quad \text{max-ctrl},
\quad \text{MG-MAX-TEMP-STK-SIZE},
\quad \text{word-size},
\quad \text{'run}))
\end{align*}\]

THEOREM: call-push-parameters-effect

\[\begin{align*}
&(\text{car}(\text{stmt}) = \text{'proc-call-mg})
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})
\land \text{ok-mg-def-plistp}(\text{proc-list})
\end{align*}\]
\[ \begin{align*}
& \wedge \ \text{ok-mg-statep} \left( \text{mg-state}, \ r\text{-cond-list} \right) \\
& \wedge \ (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \ \text{proc-list}), \ \text{proc-list}))) \\
& \quad = \ \text{append}(\text{code}(\text{translate}(\text{cinfo}, \ t\text{-cond-list}, \ \text{stmt}, \ \text{proc-list})), \ \text{code2})) \\
& \wedge \ \text{user-defined-procp}(\text{subr}, \ \text{proc-list}) \\
& \wedge \ \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}) \\
& \wedge \ (\neg \ \text{resources-inadequatep}(\text{stmt}, \\
& \quad \text{proc-list}, \ \text{temp-stk})) \\
\rightarrow \ (p(\text{map-down}(\text{mg-state}, \\
& \quad \text{proc-list}, \\
& \quad \text{ctrl-stk}, \\
& \quad \text{temp-stk}, \\
& \quad \text{tag}(\text{'pc}, \ \text{cons}(\text{subr}, \ \text{length}(\text{code}(\text{cinfo})))), \\
& \quad t\text{-cond-list}), \\
& \quad \text{data-length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \ \text{proc-list})))) \\
& \quad + \ \text{length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \ \text{proc-list})))) \\
& \quad + \ \text{length}(\text{call-actuals}(\text{stmt}))) \\
& \quad = \ \text{p-state}(\text{tag}(\text{'pc}, \\
& \quad \text{cons}(\text{subr}, \\
& \quad \ \text{length}(\text{code}(\text{cinfo})))) \\
& \quad + \ \text{data-length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \\
& \quad \text{proc-list})))) \\
& \quad + \ \text{length}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \\
& \quad \text{proc-list})))) \\
& \quad + \ \text{length}(\text{call-actuals}(\text{stmt})))) \\
& \quad \text{ctrl-stk}, \\
& \quad \text{append}(\text{reverse}(\text{mg-actuals-to-p-actuals}(\text{call-actuals}(\text{stmt})), \\
& \quad \text{bindings}(\text{top}(\text{ctrl-stk}))))), \\
& \quad \text{append}(\text{reverse}(\text{ascending-local-address-sequence}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \\
& \quad \text{proc-list})), \\
& \quad \text{length}(\text{temp-stk}))))), \\
& \quad \text{append}(\text{reverse}(\text{mg-to-p-local-values}(\text{def-locals}(\text{fetch-called-def}(\text{stmt}, \\
& \quad \text{proc-list})), \\
& \quad \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \\
& \quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \quad \text{temp-stk}))))), \\
& \quad \text{translate-proc-list}(\text{proc-list}), \\
& \quad \text{list}(\text{'c-c}, \\
& \quad \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}, \ t\text{-cond-list}))), \\
& \quad \text{MG-MAX-CTRL-STK-SIZE}, \\
\end{align*} \]
Theorem: call-call-step

\[
(n \not\approx 0) \land (\neg \text{resources-inadequatep}(stmt, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))) \land (\text{car (stmt) = '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{ok-mg-statep}(mg-state, r-cond-list) \land \text{cond-subsetp}(r-cond-list, t-cond-list) \land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)}) = \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), \text{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}(\text{mg-alist (mg-state)}) \land (\neg \text{resource-errorp}(\text{mg-meaning-r}(stmt, \text{proc-list}, mg-state, n, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))) \rightarrow (\text{p-step (p-state (tag ('pc, cons (subr, \text{length (code (cinfo)}) + \text{data-length (def-locals (fetch-called-def (stmt, proc-list)))) + \text{length (def-locals (fetch-called-def (stmt, proc-list)))) + \text{length (call-actuals (stmt))))}, ctrl-stk)})}
append (reverse (mg-actuals-to-p-actuals (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
append (reverse (ascending-local-address-sequence (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),
    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-state), t-cond-list)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, cons (call-name (stmt), 0)),
push (p-frame (append (pairlist (cadr (assoc (call-name (stmt),
    translate-proc-list (proc-list)),
append (ascending-local-address-sequence (def-locals (fetch-called-def (stmt),
    proc-list),
    length (temp-stk),
mg-actuals-to-p-actuals (call-actuals (stmt),
    bindings (top (ctrl-stk))))),
pair-temps-with-initial-values (caddr (assoc (call-name (stmt),
    translate-proc-list (proc-list),
    tag ('pc,
cons (subr,
    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)))),
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-state), t-cond-list)),

MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

**Theorem:** call-steps-to-body

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

\[\rightarrow (\text{\text{p(map-down}(mg-state, proc-list, ctrl-stk, temp-stk, tag ('pc, cons (subr, length (code (cinfo)))), t-cond-list), data-length (def-locals (fetch-called-def (stmt, proc-list))))}\]
+ length (def-locals (fetch-called-def (stmt, proc-list)))
+ length (call-actuals (stmt))
+ 1
= p-state (tag ('pc, cons (call-name (stmt), 0)),
   push (p-frame (append (pairlist (cadr (assoc (call-name (stmt),
      translate-proc-list (proc-list))))),
   append (ascending-local-address-sequence (def-locals (fetch-called-def (stmt, proc-list))),
   length (temp-stk)
   mg-actuals-to-p-actuals (call-actuals (stmt),
   bindings (top (ctrl-stk)))))
   pair-temps-with-initial-values (caddr (assoc (call-name (stmt),
   translate-proc-list (proc-list)),
   tag ('pc, cons (subr,
   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)),
   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-state), t-cond-list))),
   MG-MAX-CTRL-STK-SIZE,
   MG-MAX-TEMP-STK-SIZE,
   MG-WORD-SIZE,
   'run))

**Event**: Disable deposit-alist-value.

**Theorem**: call-local-var-lists-match1

(k ∈ N)
→ (pairlist (listcars (locals), ascending-local-address-sequence (locals, k))
  = map-call-locals (locals, k))

**Theorem**: call-formal-var-lists-match
data-param-lists-match (actuals, formals, alist)
\[ \rightarrow (\text{pairlist} \ (\text{listcars} \ (\text{formals})), \ \text{mg-actuals-to-p-actuals} \ (\text{actuals}, \ bindings)) \]

\[ = \text{map-call-formals} \ (\text{formals}, \ \text{actuals}, \ bindings) \]

**Definition:**

map-down-locals-doesnt-affect-formals-induction-hint (formals, actuals, temp-stk, mg-alist, bindings)
\[ = \begin{cases} 
\text{if} \ \text{formals} \simeq \text{nil} \ \text{then} \ \text{t} \\
\text{else} \ \text{map-down-locals-doesnt-affect-formals-induction-hint} \ (\text{cdr} \ (\text{formals}), \\
\quad \text{cdr} \ (\text{actuals}), \\
\quad \text{deposit-alist-value} \ (\text{list} \ (\text{caar} \ (\text{formals})), \\
\quad \text{cadr} \ (\text{formals}), \\
\quad \text{caddr} \ (\text{assoc} \ (\text{car} \ (\text{actuals}), \\
\quad \text{mg-alist})), \\
\quad \text{cons} \ (\text{cons} \ (\text{car} \ (\text{formals})), \\
\quad \text{cdr} \ (\text{assoc} \ (\text{car} \ (\text{actuals}), \\
\quad \text{bind})), \\
\quad \text{map-call-formals} \ (\text{cdr} \ (\text{formals}), \\
\quad \text{cdr} \ (\text{actuals}), \\
\quad \text{temp-stk}), \\
\quad \text{mg-alist}, \\
\quad \text{bindings}) \end{cases} \]

**Theorem:** map-down-locals-doesnt-affect-formals

all-cars-unique (append (formals, locals))
\[ \rightarrow (\text{map-down-values} \ (\text{make-call-param-alist} \ (\text{formals}, \ \text{actuals}, \ \text{mg-alist}), \\
\quad \text{append} \ (\text{map-call-locals} \ (\text{locals}, \ n), \\
\quad \text{map-call-formals} \ (\text{formals}, \ \text{actuals}, \ bindings)), \\
\quad \text{temp-stk})) \]

\[ = \text{map-down-values} \ (\text{make-call-param-alist} \ (\text{formals}, \\
\quad \text{actuals}, \\
\quad \text{mg-alist}), \\
\quad \text{map-call-formals} \ (\text{formals}, \ \text{actuals}, \ bindings), \\
\quad \text{temp-stk})) \]

**Definition:**

map-down-formals-doesnt-affect-locals-induction-hint (locals, temp-stk, n, lst)
\[ = \begin{cases} 
\text{if} \ \text{locals} \simeq \text{nil} \ \text{then} \ \text{t} \\
\text{elseif} \ \text{simple-mg-type-refp} \ (\text{cadr} \ (\text{car} \ (\text{locals}))) \ \text{then} \ \text{map-down-formals-doesnt-affect-locals-induction-hint} \ (\text{cdr} \ (\text{locals}), \\
\quad \text{deposit-alist-value} \ (\text{car} \ (\text{locals}), \\
\quad \text{temp-stk}) \} 
\end{cases} \]

26
append (map-call-locals (locals n),
 lst),
 temp-stk),

1 + n,
 lst)

else map-down-formals-doesnt-affect-locals-induction-hint (cdr (locals),
 deposit-alist-value (car (locals),
 append (map-call-locals (locals n),
 lst),
 temp-stk),
 array-length (cadr (car (locals))) + n,
 lst) endif

**Theorem**: map-down-formals-doesnt-affect-locals
all-cars-unique (locals)
→  (map-down-values (locals,
 append (map-call-locals (locals n), lst),
 temp-stk)) = map-down-values (locals, map-call-locals (locals n, temp-stk))

**Theorem**: map-down-skips-non-referenced-segment
(mg-alistp (vars) ∧ mg-vars-list-ok-in-p-state (vars bindings temp-stk))
→  (map-down-values (vars bindings append (lst temp-stk))
 = append (lst, map-down-values (vars bindings temp-stk)))

**Definition**:
map-down-locals-induction-hint (locals n temp-stk)
= if locals nil then t
  elseif simple-mg-type-refp (cadr (car (locals)))
  then map-down-locals-induction-hint (cdr (locals),
  1 + n,
  cons (mg-to-p-simple-literal (caddar (locals)),
  temp-stk))

  else map-down-locals-induction-hint (cdr (locals),
  array-length (cadr (car (locals))) + n,
  append (reverse (mg-to-p-simple-literal-list (caddr (car (locals)))),
  temp-stk)) endif

**Theorem**: map-down-locals-equals-reverse-values
((n length (temp-stk))
 ∧ all-cars-unique (locals)
\[ \begin{align*} &\land \text{ok-mg-local-data-plistp}(\text{locals}) \\
&\land \text{plistp}(\text{temp-stk}) \\
\rightarrow & \text{(map-down-values}(\text{locals}, \\
&\quad \text{map-call-locals}(\text{locals}, n), \\
&\quad \text{append}(\text{reverse}(\text{mg-to-p-local-values}(\text{locals})), \\
&\quad \text{temp-stk}))) \\
&= \text{append}(\text{reverse}(\text{mg-to-p-local-values}(\text{locals})), \text{temp-stk})) \end{align*} \]

**Theorem:** map-down-again-preserves-values

\[ \begin{align*} &\land \text{mg-vars-list-ok-in-p-state}(\text{mg-vars}, \text{bindings}, \text{temp-stk}) \\
&\land \text{ok-actual-params-list}(\text{actuals}, \text{mg-vars}) \\
&\land \text{data-param-lists-match}(\text{actuals}, \text{formals}, \text{mg-vars}) \\
&\land \text{all-cars-unique}(\text{mg-vars}) \\
&\land \text{plistp}(\text{temp-stk}) \\
&\land \text{no-p-aliasing}(\text{bindings}, \text{mg-vars}) \\
&\land \text{all-cars-unique}(\text{formals}) \\
&\land \text{mg-alistp}(\text{mg-vars}) \\
\rightarrow & \text{(map-down-values}(\text{make-call-param-alist}(\text{formals}, \text{actuals}, \text{mg-vars}), \\
&\quad \text{map-call-formals}(\text{formals}, \text{actuals}, \text{bindings}), \\
&\quad \text{map-down-values}(\text{mg-vars}, \text{bindings}, \text{temp-stk}))) \\
&= \text{map-down-values}(\text{mg-vars}, \text{bindings}, \text{temp-stk})) \end{align*} \]

**Event:** Enable deposit-temp.

**Theorem:** array-formal-ok-for-actual2

\[ \begin{align*} &\land \text{definedp}(\text{actual}, \text{mg-alist}) \\
&\land \text{data-params-match}(\text{actual}, \text{formal}, \text{mg-alist}) \\
&\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}, \text{bindings}, \text{temp-stk}) \\
&\land \text{(}\neg \text{simple-mg-type-refp}(\text{cadr}(\text{formal}))) \end{align*} \]

\[ \rightarrow \text{ok-temp-stk-array-index}(\text{cdr}(\text{assoc}(\text{actual}, \text{bindings})), \\
\text{map-down-values}(\text{mg-alist}, \text{bindings}, \text{temp-stk}), \\
\text{array-length}(\text{cadr}(\text{formal}))) \]

**Event:** Disable array-formal-ok-for-actual2.

**Theorem:** call-environment-mg-vars-list-ok1

\[ \begin{align*} &\land \text{no-p-aliasing}(\text{bindings}, \text{mg-vars}) \\
&\land \text{mg-alistp}(\text{mg-vars}) \\
&\land \text{all-cars-unique}(\text{formals}) \\
&\land \text{ok-actual-params-list}(\text{actuals}, \text{mg-vars}) \\
&\land \text{ok-mg-formal-data-params-plistp}(\text{formals}) \\
&\land \text{data-param-lists-match}(\text{actuals}, \text{formals}, \text{mg-vars}) \\
&\land \text{mg-vars-list-ok-in-p-state}(\text{mg-vars}, \text{bindings}, \text{temp-stk}) \\
\rightarrow & \text{mg-vars-list-ok-in-p-state}(\text{make-call-param-alist}(\text{formals}, \\
\text{actuals}, \text{mg-vars})) \end{align*} \]
Theorem: map-down-preserves-references

\((n \not= 0)\)
\(\land\) (\neg \text{resources-inadequatep} \(\text{stmt},\)
\(\text{proc-list},\)
list (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} \(\text{r-cond-list}, \text{t-cond-list}\)
\(\land\) (code (translate-def-body (assoc \(\text{subr}, \text{proc-list}\), \text{proc-list}))
= append (code (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{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,
list (length (\text{temp-stk}),
\text{p-ctrl-stk-size} (\text{ctrl-stk})))))
\rightarrow\) (append (reverse (\text{mg-to-p-local-values} \(\text{def-locals} \(\text{fetch-called-def} \(\text{stmt},\)
\text{proc-list})\)));
\text{map-down-values} \(\text{mg-alist} \(\text{mg-state}\),
\text{bindings} (\text{top} \(\text{ctrl-stk}\)),
\text{temp-stk}))
= \text{map-down-values} \(\text{make-call-var-alist} \(\text{mg-alist} \(\text{mg-state}\),
\text{stmt},\)
\text{fetch-called-def} \(\text{stmt},\)
THEOREM: call-step-initial-equals-state1
((n \not\equiv 0)
\land (\lnot \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}
\land (\text{car (stmt)} = \text{\textquotesingle 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{ok-mg-statep (mg-state, r-cond-list)}
\land \text{cond-subsetp (r-cond-list, t-cond-list)}
\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}
\quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)})
\land \text{user-defined-procp (subr, proc-list)}
\land \text{plistp (temp-stk)}
\land \text{listp (ctrl-stk)}
\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}
\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}
\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 (\lnot \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))})
\rightarrow (\text{p-state (tag ('pc, cons (call-name (stmt), 0))}, \ldots)
push (p-frame (append (pairlist (cadr (assoc (call-name stmt),
   translate-proc-list (proc-list)))),
   append (ascending-local-address-sequence (def-locals (fetch-called-def
   length (temp-stk)),
   mg-actuals-to-p-actuals (call-actuals stmt,
   bindings (top (ctrl-stk))))),
   pair-temps-with-initial-values (caddr (assoc (call-name stmt),
   translate-proc-list (proc-list))
   tag ('pc,
   cons (subr,
   cons (subr,
   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)),
   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-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run)
= 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,
1 + (length (code (cinfo))
+ data-length (def-locals (fetch-called-def (stmt,
;; This lemma characterizes the behavior all of the way through the call-body

(prove-lemma call-step-initial-to-state2 (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)
    (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-statet 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)))

;; This lemma characterizes the behavior all of the way through the call-body
(USER-DEFINED-PROCP SUBR PROC-LIST)
(PLISTP TEMP-STK)
(LISTP CTRL-STK)
(MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE))
(BINDINGS (TOP CTRL-STK))
TEMP-STK)
  (NO-P-ALIASING (BINDINGS (TOP CTRL-STK))
  (MG-ALIST MG-STATE))
  (SIGNATURES-MATCH (MG-ALIST MG-STATE)
  NAME-ALIST)
  (NORMAL MG-STATE)
  (ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
  (NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)
  (P-CTRL-STK-SIZE CTRL-STK))))
  (EQUAL
   (P
    (MAP-DOWN ;; state1
      (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

33
(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))))))
                      1) )))
                0)))
            0)))
          (SUB1 N))
        1)))
      0)))
    0) )
(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
(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}

35
(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))
      (MAKE-CINFO NIL
        (CONS
          '(ROUTINEERROR . 0)
          (MAKE-LABEL-ALIST
            (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
            0))
        (MAKE-CINFO NIL
          (CONS
            '(ROUTINEERROR . 0)
            (MAKE-LABEL-ALIST
              (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
              0))
          (MAKE-CINFO NIL
            (CONS
              '(ROUTINEERROR . 0)
              (MAKE-LABEL-ALIST
                (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
                0))
            (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)))))))
  (BINDINGS
    (TOP
    (CONS
      (P-FRAME
        (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
STMT CTRL-STK TEMP-STK)
        (TAG 'PC
        (CONS SUBR

37
(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))
EQUAL
(P (MAP-DOWN MG-STATE PROC-LIST CTRL-STK TEMP-STK
(TAG 'PC
(CONS SUBR (LENGTH (CODE CINFO))))
T-COND-LIST)
  (plus (data-length (def-locals (fetch-called-def stmt proc-list)))
(length (def-locals (fetch-called-def stmt proc-list)))
(length (call-actuals stmt))
1
(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
  (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

40
(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)

41
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))))))
           (BINDINGS
            (TOP
             (CONS
              (P-FRAME

42
(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
        43)
(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)))
((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 @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

CONSOLEERROR . 0)
(MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)
  0))
))))
(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)
  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

45
(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
          (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))))))))
    (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

47
(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-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))))
(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))))))))
      (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 @BODY-TIME
  (CLOCK (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))
  PROC-LIST

49
Event: Enable make-call-environment.

(prove-lemma call-state2-step1-effect (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)
      (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)))
  (make-call-environment mg-state stmt
    (fetch-called-def stmt proc-list))
  (add-abbreviation @time-to-state1
    (plus (data-length (def-locals (fetch-called-def stmt proc-list)))
      (length (def-locals (fetch-called-def stmt proc-list)))
      (length (call-actuals stmt))
      1))
  (promote
    (dive 1 2)
    (= (plus (plus (data-length (def-locals (fetch-called-def stmt proc-list)))
                  (length (def-locals (fetch-called-def stmt proc-list)))
                  (length (call-actuals stmt))
                  1)
       @body-time))
  (up
    (rewrite p-plus-lemma)
    (dive 1)
    (rewrite call-steps-to-body)
    (rewrite call-step-initial-equals-state1)
    (up up
      (demote 19)
      s-prop)))

50
(USER-DEFINED-PROCP SUBR PROC-LIST)
(PLISTP TEMP-STK)
(LISTP CTRL-STK)
(MG-VARS-LIST-OK-IN-P-STATE (MG-ALIST MG-STATE)
BINDINGS (TOP CTRL-STK))
TEMP-STK)
(NO-P-ALIASING (BINDINGS (TOP CTRL-STK))
(MG-ALIST MG-STATE))
(SIGNATURES-MATCH (MG-ALIST MG-STATE)
NAME-ALIST)
(NORMAL MG-STATE)
(ALL-CARS-UNIQUE (MG-ALIST MG-STATE))
(NOT (RESOURCE-ERRORP (MG-MEANING-R STMT PROC-LIST MG-STATE N)
(LIST (LENGTH TEMP-STK)
(P-CTRL-STK-SIZE CTRL-STK))))
(equal
(p-step
 (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

51
(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
                (MAKE-FRAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST)
                STMT CTRL-STK TEMP-STK))))))

52
(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))))))))
(BINDINGS
  (TOP
  (CONS
  (P-FRAME

54
(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))))
   (TRANSimize-PROC-LIST PROC-LIST)
   (LIST
    (LIST 'C-C
      (MG-CON4-TO-P-NAT
       (CC
        (MG-MEANIG-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))
   (P-STATE (TAG 'PC
    (CONS
     (CALL-NAME STMT)
     (PLUS
      (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))
         1))))
        (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
          (MAP-DOWN-VALUES
           (MG-ALIST
            (MG-MEANING-R
             (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))))

56
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)))))
  (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
  (MG-MAX-CTRL-STK-SIZE) (MG-MAX-TEMP-STK-SIZE) (MG-WORD-SIZE)
'RUN)))))
((INSTRUCTIONS
  (ENABLE LENGTH-CONS)
  PROMOTE
  (DIVE 1)
  X
  (S LEMMAS)
  (DIVE 1 1 1)
  (= *
   (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))
  0))

58
(REWRITE TRANSLATE-DEF-BODY-REWRITE
 (($CINFO
   (MAKE-CINFO NIL
     (CONS
       (CONS 'ROUTINEERROR 0)
       (MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
         0))
     1))
   ($T-COND-LIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))
 ($CODE2
   (LIST '(DL 0 NIL (NO-OP))
     (LIST 'POP*
       (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
     '(RET))
   ($STMT (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST))))))

(REWRITE GET-LENGTH-CAR)
(S LEMMAS)
UP X UP X
(S LEMMAS)
X
(S LEMMAS)
(DIVE 1 2 2 1)
('= *
 (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))))
)

(REWRITE LENGTH-MG-TO-P-LOCAL-VALUES)
NX
(REWRITE MAP-DOWN-VALUES-PRESERVES-LENGTH)
Theorem: call-body-mg-vars-list-ok1
\[(n \not\equiv 0) \land \neg \text{resources-inadequatep}(stmt, proc-list),\]
list (length (temp-stk),
    p-ctrl-stk-size (ctrl-stk)))
∧ (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)
∧ (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)))))
→ mg-vars-list-ok-in-p-state (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)),
    n − 1, list (data-length (def-locals (fetch-called-def (stmt,
    proc-list)),
    + length (temp-stk),
    2
    + length (def-locals (fetch-called-def (stmt, proc-list))),
    + length (def-formals (fetch-called-def (stmt,
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))) ;; (pop* (data-length locals))

;; (pop* (data-length locals))

(prove-lemma call-state2-step2-effect (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)
   (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-translation 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)))

62
(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
             (CALL-NAME STMT)
             (PLUS
              (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)))
                 1))))
(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)
   (MAP-DOWN-VALUES
    (MG-ALIST
     63
(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)))))
 (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
 (CALL-NAME STMT)
 (PLUS
 (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)))
 2)))))
 (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)
  (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

66
(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))
(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)))
       (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
       (LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
       (LENGTH (CALL-ACTUALS STMT))))))))
  (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))))))
  (MG-MAX-CTRL-STK-SIZE) (MG-MAX-TEMP-STK-SIZE) (MG-WORD-SIZE)
'RUN))
((INSTRUCTIONS PROMOTE
  (DIVE 1)
  X
  (S LEMMAS)
  (DIVE 1 1 2)
(REWRITE TRANSLATE-DEF-BODY-REWRITE
   (($CINFO
      (MAKE-CINFO NIL
       (CONS
        (CONS 'ROUTINEERROR 0)
        (MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)) 0))
       1))
      ($T-COND-LIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)))
      ($CODE2
       (LIST '(DL 0 NIL (NO-OP))
        (LIST 'POP*
         (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
        '(RET))
       ($STMT (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST)))))
   (REWRITE GET-LENGTH-PLUS)
   (S LEMMAS)
   UP X UP X
   (S LEMMAS)
   (DIVE 1)
   X UP S
   (S LEMMAS)
   UP S-PROP
   (DIVE 1)
   (DIVE 1)
   (REWRITE MAP-DOWN-VALUES-PRESERVES-LENGTH)
   (S LEMMAS)
   (DIVE 1)
   (REWRITE LENGTH-MG-TO-P-LOCAL-VALUES)
   UP UP
   (= * F)
   UP S
   (REWRITE CALLED-DEF-FORMALS-OK)
   (REWRITE CALL-BODY-MG-VARS-LIST-OK1)
   (DIVE 1 1)
   (REWRITE MG-MEANING-EQUIVALENCE)
   UP UP
   (REWRITE MG-MEANING-PRESERVES-MG-ALISTP
    (($R-COND-LIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)))
     ($NAME-ALIST (MAKE-NAME-ALIST (FETCH-CALLED-DEF STMT PROC-LIST))))
    (DIVE 1)
    (= *
     (MAKE-CALL-ENVIRONMENT MG-STATE STMT))
(FETCH-CALLED-DEF STMT PROC-LIST))

0)

UP
(REWRITE PROC-CALL-EXACT-TIME-HYPS)
S PROVE
(REWRITE CALL-EXACT-TIME-HYPS1)
(DIVE 1)
S UP
(REWRITE CALL-SIGNATURES-MATCH2)
(DIVE 1)
(REWRITE MORE-RESOURCES-PRESERVES-NOT-RESOURCE-ERRORP
((T-SIZE1
  (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)))))
(C-SIZE1
(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)))))

UP S
(S LEMMAS)
(DIVE 1 1 1)
(REWRITE LENGTH-MG-TO-P-LOCAL-VALUES)
NX
(REWRITE MAP-DOWN-VALUES-PRESERVES-LENGTH)
TOP DROP PROVE
(REWRITE OK-MG-STATEP-MG-ALIST-MG-ALISTP)
(REWRITE CALLED-DEF-FORMALS-OK)
(DIVE 1 1)
X
(S LEMMAS)
(S-PROP P-FRAME-SIZE)
(S LEMMAS)
UP TOP DROP PROVE
(DIVE 1 1 3)
(= *
(MAKE-CALL-ENVIRONMENT MG-STATE STMT
 (FETCH-CALLED-DEF STMT PROC-LIST))
0)
TOP
(DIVE 1)
(REWRITE PROC-CALL-DOESNT-HALT)
TOP S
(S-PROP MAKE-CALL-ENVIRONMENT)
(PROVE (ENABLE OK-MG-STATEMENT OK-PROC-CALL))
(REWRITE PROC-CALL-EXACT-TIME-HYPS)
(DIVE 1 1)
X TOP
(PROVE (ENABLE FETCH-CALLED-DEF FETCH-DEF)))))

;; (ret)

;; (ret)

(prove-lemma call-state2-step3-effect (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)
 (OK-MG-STATEMENT STMT R-COND-LIST NAME-ALIST PROC-LIST)
 (OK-MG-DEF-PLISTP PROC-LIST)
 (OK-TRANSLATION-PARAMETERS CINFO T-COND-LIST STMT PROC-LIST CODE2)
 (OK-MG-STATEP MG-STATE R-COND-LIST)
 (COND-SUBSETP R-COND-LIST T-COND-LIST)
 (EQUAL (CODE (TRANSLATE-DEF-BODY (ASSOC SUBR PROC-LIST)
 PROC-LIST))
 (APPEND (CODE (TRANSLATE CINFO T-COND-LIST STMT PROC-LIST))
 CODE2))
 (USER-DEFINED-PROC SUBR PROC-LIST)

70
(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
     (CALL-NAME STMT)
     (PLUS
     (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))))
     2))))
     (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))
     71
(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

72
(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)))))
    (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
    (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))))
   74
(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)))))
 ((INSTRUCTIONS PROMOTE
 (DIVE 1)
 X
 (S LEMMAS)
 (DIVE 1 1 2)
 (REWRITE TRANSLATE-DEF-BODY-REWRITE
 (((CINFO
 (MAKE-CINFO NIL
 (CONS
 (CONS 'ROUTINEERROR 0)
 (MAKE-LABEL-ALIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST))
 0)))
 1))
 ($T-COND-LIST (MAKE-COND-LIST (FETCH-CALLED-DEF STMT PROC-LIST)))
 ($CODE2
 (LIST
 '(DL 0 NIL (NO-OP))
 (LIST 'POP* (DATA-LENGTH (DEF-LOCALS (FETCH-CALLED-DEF STMT PROC-LIST))))
 '(RET)))
 ($STMT (DEF-BODY (FETCH-CALLED-DEF STMT PROC-LIST)))))))
 UP
 (REWRITE GET-LENGTH-PLUS)
 (S LEMMAS)
 UP X UP X
(S LEMMAS)
X
(S LEMMAS)
UP prove
(PROVE (ENABLE OK-MG-STATEMENT OK-PROC-CALL))
(DIVE 1 1)
X TOP
(PROVE (ENABLE FETCH-CALLED-DEF FETCH-DEF)))))

EVENT: Make the library "c-proc-call1".
Index

actual-params-list-ok-in-mg-alist, 5
actual-pointers-distinct, 4
actual-pointers-distinct2, 4
all-cars-unique, 3–9, 12, 15, 22, 24, 26–30, 61
all-pointers-bigger, 3
all-pointers-smaller, 3
array-alist-element-lengths-match, 4
array-formal-ok-for-actual, 8
array-formal-ok-for-actual2, 28
array-identifierp, 1
array-length, 4, 5, 8, 18, 27, 28
ascending-local-address-sequence, 18, 20, 21, 23, 25, 31
bindings, 6, 9–15, 19–25, 29–32, 61, 62
c-size, 2
call-actuals, 5, 10, 11, 14, 15, 21–23, 25, 31, 32, 62
call-body-mg-vars-list-ok1, 60
call-body-rewrite, 7
call-call-step, 22
call-conds, 3
call-environment-mg-vars-list-ok1, 28
call-exact-time-translation-parameters-ok, 6
call-formal-vars-lists-match, 26
call-local-names-unique, 6
call-local-vars-list-match1, 25
call-name, 7, 13, 23, 25, 30–32
call-push-actuals-effect, 19
call-push-actuals-induction-hint, 19
call-push-locals-addresses-effect, 18
call-push-locals-values-effect, 17
call-push-parameters-effect, 20
call-push-parameters-effect1, 20
call-step-initial-equals-state1, 30
call-steps-to-body, 24
call-translation-2, 3
cc, 21, 23–25, 31, 61
code, 3, 7, 9, 10, 12–15, 21–25, 29–32, 61
collect-pointers, 3–5
cond-subsetp, 9, 12, 13, 22, 24, 29, 30, 61
data-length, 2, 7, 11, 13–15, 17, 18, 20–25, 31, 32, 61
data-param-lists-match, 3–5, 8, 26, 28
data-param-lists-match-in-mg-alist, 5
data-params-match, 8, 28
def-body, 2, 7, 10–13, 15, 61
def-cond-locals, 3
def-formals, 3, 5, 6, 9, 11, 62
def-locals, 2, 3, 6, 7, 9–11, 13–15, 21–25, 29–32, 61, 62
defined-identifierp, 4, 8
definedp, 1, 4, 7, 8, 28
definedp-caar, 7
deposit-alist-value, 26, 27
disjoint, 5
exp, 1
extra-bindings-doesnt-affect-parameters-preserved, 1
extra-bindings-dont-affect-no-parameter-p
aliasing, 4
-aliasing2, 4
fetch-label-0-case-2, 1
formal-types-preserved, 1
label-alist, 3
no-p-aliasing-formals, 5
no-p-aliasing-in-call-environment, 6
no-p-aliasing-locals, 3
normal, 2, 9–12, 15, 22, 24, 29, 30, 61
not-simple-identifiers-array-identifiers, 1
ok-actual-params-list, 3–5, 8, 28
ok-mg-def-plistp, 6, 9–12, 20, 22, 24, 29, 30, 61
ok-mg-formal-data-params-plistp, 4, 5, 8, 28
ok-mg-local-data-plistp, 7, 17, 18, 20, 28
ok-mg-statement, 5, 6, 9–12, 20, 22, 24, 29, 30, 61
ok-mg-statep, 5, 6, 9–13, 21, 22, 24, 29, 30, 61
ok-temp-stk-array-index, 8, 28
ok-temp-stk-index, 8
ok-translation-parameters, 6, 7, 9, 12, 13, 22, 24, 29, 30, 61
p, 16–21, 25
p-ctrl-stk-size, 9–12, 15, 21, 22, 24, 29, 30, 61, 62
p-frame, 11, 14, 15, 23, 25, 31, 32
p-objectp-type, 1
p-state, 16–20, 22–25, 31
p-step, 23
p-word-size, 1
pair-temps-with-initial-values, 23, 25, 31
plistp, 1, 6, 9, 12, 13, 22, 24, 28–30, 61
proc-call-code, 3
proc-call-doesnt-halt, 10
proc-call-doesnt-halt2, 11
proc-call-exact-time-hyps, 11
proc-call-meaning-r-2, 2
push, 16–18, 23, 25, 31
push-actuals-code, 19
push-array-value-induction-hint, 16
push-local-array-values-code, 16, 17
push-local-array-values-code-effect, 16
push-locals-addresses-code, 18
push-locals-values-code, 17
push-parameters-code, 20
resource-errorp, 10–12, 15, 22, 24, 29, 30, 61
resources-inadequatep, 2, 9, 12, 21, 22, 24, 29, 30, 61
reverse, 7, 8, 10, 13–21, 23, 25, 27–32, 62
set-alist-value, 1
set-alist-value-map-down-values-length-doesnt-shrink, 1
signal-system-error, 2
signatures-match, 2, 5, 6, 9, 12, 15, 22, 24, 29, 30, 61
simple-formal-ok-for-actual2, 8
simple-identifiersp, 1
simple-mg-type-refp, 4, 5, 7, 8, 16–18, 26–28
simple-typed-literal-listp, 16
simple-typed-literal-plistp, 16
simple-typed-literalp, 16
t-size, 2
tag, 7, 11, 14–25, 30–32
tag-length-plistp-2, 7
top, 6, 9–15, 19–25, 29–32, 61, 62
translate, 3, 7, 9, 12, 13, 21, 22, 24, 29, 30, 61
translate-def-body, 7, 9, 12, 13, 21, 22, 24, 29, 30, 61
translate-proc-list, 21, 23, 25, 31
untag, 1, 4
user-defined-procp, 6, 9, 12, 13, 21, 22, 24, 29, 30, 61