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

EVENT: Enable clock-predefined-proc-call-sequence.

EVENT: Enable clock-predefined-proc-call-body-translation.

EVENT: Enable predefined-proc-call-sequence.

EVENT: Disable mg-to-p-simple-literal.

EVENT: Disable ilessp.

EVENT: Disable not-bool.

EVENT: Disable inegate.

EVENT: Disable fix-small-integer.

EVENT: Disable iplus.

EVENT: Disable idifference.

EVENT: Disable signatures-match-preserves-plistp.

THEOREM: predefined-proc-call-meaning-r-2
\[ \text{car}(stmt) = \text{	exttt{predefined-proc-call-mg}} \]
\[ \rightarrow \ (\text{mg-meaning-r}(stmt, \text{proc-list}, \text{mg-state}, n, sizes)) \]
\[ = \quad \text{if} \ n \\simeq \ 0 \ \text{then} \ \text{signal-system-error}(\text{mg-state}, \text{'timed-out}) \]
\[ \quad \text{elseif} \ \neg \ \text{normal}(\text{mg-state}) \ \text{then} \ \text{mg-state} \]
\[ \quad \text{elseif} \ \text{resources-inadequatep}(stmt, \text{proc-list}, sizes) \]
\[ \quad \text{then} \ \text{signal-system-error}(\text{mg-state}, \text{'resource-error}) \]
\[ \quad \text{else} \ \text{mg-meaning-predefined-proc-call}(stmt, \text{mg-state}) \ \text{endif} \]

THEOREM: predefined-call-translation-2
\[ \text{car}(stmt) = \text{	exttt{predefined-proc-call-mg}} \]
\[ \rightarrow \ (\text{translate}(cinfo, \text{cond-list}, stmt, \text{proc-list})) \]
\[ = \quad \text{add-code}(cinfo, \]
\[ \quad \text{predefined-proc-call-sequence}(stmt, \]
\[ \]
Theorem: unlabel-call
unlabel (cons ('call, x)) = cons ('call, x)

Theorem: lessp-add1-add1-2
((1 + (1 + x)) < 2) = f

Theorem: lessp-add1-add1-add1-3
((1 + (1 + (1 + x))) < 3) = f

;; >> get rid of the extra "d"

Theorem: simple-typed-literalp-mapping-listp
simple-typed-literal (lit, type) → listp (mg-to-p-simple-literal (lit))

Theorem: boolean-mg-to-p-simple-literal
(x ∈ '(true-mg false-mg))
→ (mg-to-p-simple-literal (tag ('boolean-mg, x))
    = tag ('bool,
        if x = 'false-mg then 'f
        else 't endif))

Theorem: boolean-mg-to-p-simple-literal2
mg-to-p-simple-literal (mg-bool (x))
= tag ('bool,
    if ¬ x then 'f
    else 't endif)

Theorem: mg-to-p-simple-literalp-preserves-untag-equality
(simple-typed-literalp (x, type) ∧ simple-typed-literalp (y, type))
→ ((untag (mg-to-p-simple-literal (x))
    = untag (mg-to-p-simple-literal (y))
    = (untag (x) = untag (y)))

Theorem: mg-to-p-simple-literalp-preserves-untag-ilessp
(int-literalp (x) ∧ int-literalp (y))
→ (ilessp (untag (mg-to-p-simple-literal (x)),
    untag (mg-to-p-simple-literal (y)))
    =  ilessp (untag (x), untag (y)))

Theorem: simple-identifiers-have-simple-values
(mg-alistp (alist) ∧ simple-identifierp (x, alist))
→ simple-typed-literalp (caddr (assoc (x, alist)), cadr (assoc (x, alist)))
Theorem: alist-element-type-conversion
\( (\text{mg-alistp} \ (\text{lst}) \land \text{simple-identifierp} \ (x, \text{lst})) \)
\( \rightarrow \ (\text{type} \ (\text{mg-to-p-simple-literal} \ (\text{caddr} \ (\text{assoc} \ (x, \text{lst})))))
= \ \text{if} \ \text{cadr} \ (\text{assoc} \ (x, \text{lst})) = \text{'boolean-mg} \ \text{then} \text{'bool}
\text{else} \text{'int} \text{endif} \)

Theorem: literal-type-conversion
\( \text{simple-typed-literalp} \ (\text{lit}, \text{type}) \)
\( \rightarrow \ (\text{type} \ (\text{mg-to-p-simple-literal} \ (\text{lit})))
= \ \text{if} \ \text{type} = \text{'boolean-mg} \ \text{then} \text{'bool}
\text{else} \text{'int} \text{endif} \)

Theorem: int-literals-mapping
\( \text{int-literalp} \ (x) \)
\( \rightarrow \ ((\text{type} \ (\text{mg-to-p-simple-literal} \ (x))) = \text{'int})
\land \text{listp} \ (\text{mg-to-p-simple-literal} \ (x))
\land \ (\text{cddr} \ (\text{mg-to-p-simple-literal} \ (x)) = \text{nil})
\land \text{small-integerp} \ (\text{untag} \ (\text{mg-to-p-simple-literal} \ (x)), \ 32) \)

Theorem: int-identifiers-have-int-literal-values
\( \text{mg-alistp} \ (\text{mg-vars}) \land \text{int-identifierp} \ (x, \text{mg-vars}) \)
\( \rightarrow \ \text{int-literalp} \ (\text{caddr} \ (\text{assoc} \ (x, \text{mg-vars}))) \)

Theorem: boolean-identifiers-have-boolean-literal-values
\( \text{mg-alistp} \ (\text{mg-vars}) \land \text{boolean-identifierp} \ (x, \text{mg-vars}) \)
\( \rightarrow \ \text{boolean-literalp} \ (\text{caddr} \ (\text{assoc} \ (x, \text{mg-vars}))) \)

Theorem: length-plistp-2-rewrite
\( (\text{cddr} \ (x) = \text{nil}) = \text{length-plistp} \ (x, \ 2) \)

Event: Disable length-plistp-2-rewrite.

;; There is no reason why this couldn’t be changed to defined-identifierp’s
;; rather than simple-identifierp’s

Theorem: simple-identifier-mapping
\( \text{mg-vars-list-ok-in-p-state} \ (\text{mg-vars}, \text{bindings}, \text{temp-stk}) \)
\( \land \ \text{simple-identifierp} \ (b, \text{mg-vars}) \)
\( \land \ (\text{length} \ (\text{temp-stk}) < \text{MG-MAX-TEMP-STK-SIZE}) \)
\( \land \ \text{all-cars-unique} \ (\text{mg-vars}) \)
\( \rightarrow \ ((\text{type} \ (\text{value} \ (b, \text{bindings}))) = \text{'nat})
\land \ (\text{cddr} \ (\text{value} \ (b, \text{bindings})) = \text{nil})
\land \ \text{listp} \ (\text{value} \ (b, \text{bindings}))
\land \ \text{small-naturalp} \ (\text{untag} \ (\text{value} \ (b, \text{bindings})), \ 32) \)
Theorem: simple-identifier-mapping-2
\[(\text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings, temp-stk)} \]
\[\land \text{simple-identifierp (b, mg-alist (mg-state))} \]
\[\land (\text{length (temp-stk)} < \text{MG-MAX-TEMP-STK-SIZE}) \]
\[\land \text{all-cars-unique (mg-alist (mg-state))} \]
\[\rightarrow \left( (\text{type (value (b, bindings))} = \text{'nat}) \right) \]
\[\land \text{(cddr (value (b, bindings))) = nil} \]
\[\land \text{listp (value (b, bindings))} \]
\[\land \text{small-naturalp (untag (value (b, bindings)), 32)} \]

Theorem: simple-identifier-mapping-3
\[(\text{mg-vars-list-ok-in-p-state (mg-alist, bindings, temp-stk)} \]
\[\land \text{definedp (b, mg-alist)} \]
\[\rightarrow \left( \text{(untag (value (b, bindings))) < (1 + length (temp-stk))} = \text{t} \right) \]

Theorem: simple-identifier-nat-p-objectp
\[(\text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings, temp-stk)} \]
\[\land \text{simple-identifierp (b, mg-alist (mg-state))} \]
\[\land (\text{length (temp-stk)} < \text{MG-MAX-TEMP-STK-SIZE}) \]
\[\land \text{all-cars-unique (mg-alist (mg-state))} \]
\[\land (\text{p-word-size (state)} = \text{MG-WORD-SIZE}) \]
\[\rightarrow \text{p-objectp-type ('nat, value (b, bindings), state)} \]

Event: Enable mg-var-ok-in-p-state.

Theorem: array-identifier-nat-p-objectp
\[(\text{mg-vars-list-ok-in-p-state (mg-alist, bindings, temp-stk)} \]
\[\land \text{definedp (b, mg-alist)} \]
\[\land (\text{length (temp-stk)} < \text{MG-MAX-TEMP-STK-SIZE}) \]
\[\land (\text{p-word-size (state)} = \text{MG-WORD-SIZE}) \]
\[\rightarrow \text{p-objectp-type ('nat, value (b, bindings), state)} \]

Theorem: int-literalp-mapping
\[\text{int-literalp (x)} \rightarrow (\text{untag (mg-to-p-simple-literal (x))} = \text{untag (x)}) \]

Theorem: boolean-literalp-mapping
\[\text{boolean-literalp (x)} \rightarrow (\text{untag (mg-to-p-simple-literal (x))} = \text{if untag (x) = 'false-mg then 'f} \]
\[\text{else 't endif)} \]

Theorem: untag-boolean
\[\text{boolean-literalp (x)} \rightarrow (\text{untag (x)} \in \text{'(true-mg false-mg)}) \]
Theorem: int-literalp-value-small
\[ \text{int-literalp} \ (x) \to \text{small-integerp} \ (\text{untag} \ (x), \ 32) \]

Theorem: small-integerp-mapping
\[ \begin{align*} 
\text{small-integerp} \ (x, \ \text{MG-WORD-SIZE}) \\
\to \ (\text{mg-to-p-simple-literal} \ (\text{tag} \ ('\text{int-mg}, \ x)) = \text{tag} \ ('\text{int}, \ x)) 
\end{align*} \]

Theorem: special-conditions-mg-cond-to-p-nat
\[ \begin{align*} 
(\text{mg-cond-to-p-nat} ('\text{routineerror}, \ \text{state}) = '\text{nat} \ 1) \\
\land \ (\text{mg-cond-to-p-nat} ('\text{normal}, \ \text{state}) = '\text{nat} \ 2) 
\end{align*} \]

Theorem: fix-small-integer-identity
\[ \begin{align*} 
\text{small-integerp} \ (x, \ n) \\
\to \ (\text{fix-small-integer} \ (x, \ n) = x) 
\end{align*} \]

Theorem: int-literal-int-objectp
\[ \begin{align*} 
(\text{int-literalp} \ (x) \land (\text{p-word-size} \ (\text{state}) = \text{MG-WORD-SIZE})) \\
\to \ \text{p-objectp-type} \ ('\text{int}, \ \text{mg-to-p-simple-literal} \ (x), \ \text{state}) 
\end{align*} \]

Theorem: untag-int-literal-integerp
\[ \begin{align*} 
\text{int-literalp} \ (x) \\
\to \ \text{integerp} \ (\text{untag} \ (x)) 
\end{align*} \]

Theorem: bool-literal-bool-objectp
\[ \begin{align*} 
\text{p-objectp-type} \ ('\text{bool}, \ '\text{bool t}), \ \text{state}) \\
\land \ \text{p-objectp-type} \ ('\text{bool}, \ '\text{bool f}), \ \text{state}) 
\end{align*} \]

Theorem: bool-literal-bool-objectp2
\[ \begin{align*} 
\text{boolean-literalp} \ (x) \\
\to \ \text{p-objectp-type} \ ('\text{bool}, \ \text{mg-to-p-simple-literal} \ (x), \ \text{state}) 
\end{align*} \]

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; ;;
; ;; EXACT-TIME LEMMA MG-SIMPLE-VARIABLE-ASSIGNMENT
; ;;
; ;;
; ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Theorem: mg-simple-variable-assignment-args-definedp
\[ \begin{align*} 
(\text{ok-mg-statement} \ (\text{stmt}, \ \text{r-cond-list}, \ \text{name-alist}, \ \text{proc-list}) \\
\land \ (\text{car} \ (\text{stmt}) = '\text{predefined-proc-call-mg}) \\
\land \ (\text{call-name} \ (\text{stmt}) = '\text{mg-simple-variable-assignment}) \\
\land \ \text{ok-mg-statep} \ (\text{mg-state}, \ \text{r-cond-list}) \\
\land \ \text{signatures-match} \ (\text{mg-alist} \ (\text{mg-state}), \ \text{name-alist})) \\
\to \ (\text{definedp} \ (\text{car} \ (\text{call-actuals} \ (\text{stmt}))), \ \text{mg-alist} \ (\text{mg-state})) \\
\land \ \text{definedp} \ (\text{cadr} \ (\text{call-actuals} \ (\text{stmt}))), \ \text{mg-alist} \ (\text{mg-state})) 
\end{align*} \]
THEOREM: mg-simple-variable-assignment-args-simple-identifierps
(\( \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list) } \)
\( \land \) \( (\text{car (stmt)} = \text{'predefined-proc-call-mg}) \)
\( \land \) \( (\text{call-name (stmt)} = \text{'mg-simple-variable-assignment}) \)
\( \land \) \( \text{ok-mg-statep (mg-state, r-cond-list)} \)
\( \land \) \( \text{signatures-match (mg-alist (mg-state), name-alist))} \)
\( \rightarrow \) \( (\text{simple-identifierp (\text{car (call-actuals (stmt)}, mg-alist (mg-state)))} \)
\( \land \) \( \text{simple-identifierp (cadr (call-actuals (stmt)), mg-alist (mg-state)))} \)

THEOREM: mg-simple-variable-assignment-steps-1-2
(\( (\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))} \)
\( \land \) \( (\text{car (stmt)} = \text{'predefined-proc-call-mg}) \)
\( \land \) \( (\text{call-name (stmt)} = \text{'mg-simple-variable-assignment}) \)
\( \land \) \( \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \)
\( \land \) \( \text{ok-mg-def-plistp (proc-list)} \)
\( \land \) \( \text{ok-mg-statep (mg-state, r-cond-list)} \)
\( \land \) \( \text{code (translate-def-body (assoc (subr, proc-list), proc-list))} \)
\( = \) \( \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))} \)
\( \land \) \( \text{user-defined-procp (subr, proc-list)} \)
\( \land \) \( \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)} \)
\( \land \) \( \text{normal (mg-state)} \)
\( \rightarrow \) \( (\text{p-step (p-step (map-down (mg-state, proc-list, ctrl-stk, temp-stk, tag ('pc, cons (subr, length (code (cinfo))))), t-cond-list)))} \)
\( = \) \( \text{p-state (tag ('pc, cons (subr, length (code (cinfo)) + 2)), ctrl-stk,}
\( \text{push (value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk))),}
\( \text{push (value (car (call-actuals (stmt))), bindings (top (ctrl-stk))),}
\( \text{map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk))}, \)
\( \text{translate-proc-list (proc-list),}
\( \text{list (list ('c-c,}
\( \)) \) \)}}
\[
\text{mg-cond-to-p-nat (cc (mg-state, t-cond-list))},
\]
\[
\text{MG-MAX-CTRL-STK-SIZE,}
\]
\[
\text{MG-MAX-TEMP-STK-SIZE,}
\]
\[
\text{MG-WORD-SIZE,}
\]
\[
\text{'}run\})
\]
\[
;; (\text{call mg-simple-variable-assignment})
\]

**Theorem**: mg-simple-variable-assignment-step-3
\[
(\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}
\]
\[
\land (\text{car (stmt) = 'predefined-proc-call-mg})
\]
\[
\land (\text{call-name (stmt) = 'mg-simple-variable-assignment})
\]
\[
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}
\]
\[
\land \text{ok-mg-def-plistp (proc-list)}
\]
\[
\land \text{ok-mg-statep (mg-state, r-cond-list)}
\]
\[
\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}
\]
\[
= \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}
\]
\[
\land \text{user-defined-procp (subr, proc-list)}
\]
\[
\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}
\]
\[
\land \text{normal (mg-state)}
\]
\[
\rightarrow (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 2)),}
\]
\[
\text{ctrl-stk,}
\]
\[
\text{push (value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk))),}
\]
\[
\text{push (value (car (call-actuals (stmt))), bindings (top (ctrl-stk))),}
\]
\[
\text{map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk))},
\]
\[
\text{translate-proc-list (proc-list),}
\]
\[
\text{list (list ('c-c, mg-cond-to-p-nat (cc (mg-state), t-cond-list))},}
\]
\[
\text{MG-MAX-CTRL-STK-SIZE,}
\]
\[
\text{MG-MAX-TEMP-STK-SIZE,}
\]
\[
\text{MG-WORD-SIZE,}
\]
\[
\text{'}run\})
\]
\[
= \text{p-state (tag ('pc, (mg-simple-variable-assignment . 0))},
\]
Theorem: mg-simple-variable-assignment-step-4

\[ \neg \text{resources-inadequatep}\(\text{stmt}, \text{proc-list}\) \land (\text{car}(\text{stmt}) = \text{predefined-proc-call-mg}) \land (\text{call-name}(\text{stmt}) = \text{mg-simple-variable-assignment}) \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{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})) = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \land \text{user-defined-procp}(\text{subr}, \text{proc-list}) \land \text{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{normal}(\text{mg-state}) \rightarrow \text{p-step}(\text{p-state}(\text{tag}(\text{pc}, \text{push}(\text{p-frame}(\text{list}(\text{cons}(\text{dest}, \text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk})))))), \text{cons}(\text{source}, \text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk})))))), \text{tag}(\text{pc}, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo})) + 3)), \text{ctrl-stk}), \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk}), \text{translate-proc-list}(\text{proc-list}), \text{list}(\text{list}(\text{c-c}, \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}, \text{t-cond-list})))), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}) \]
\[
\text{value (car (call-actuals (\textit{stmt}))),}
\text{bindings (top (\textit{ctrl-stk})))},
\text{\textbf{cons}} ('\textbf{source},}
\text{\textbf{value (cadr (call-actuals (\textit{stmt}))),}
\text{\textbf{bindings (top (\textit{ctrl-stk})))}},
\text{tag ('\textbf{pc},}
\text{\textbf{cons (\textit{subr}, length (code (\textit{cinfo}))}
+ 3))},
\text{\textit{ctrl-stk}),}
\text{map-down-values (mg-alist (\textit{mg-state}),}
\text{\textbf{bindings (top (\textit{ctrl-stk}))),}
\text{\textit{temp-stk}),}
\text{translate-proc-list (\textit{proc-list}),}
\text{list (list ('c-c,}
\text{\textit{mg-cond-to-p-nat (cc (\textit{mg-state}), \textit{t-cond-list}))),}
\text{MG-MAX-CTRL-STK-SIZE,}
\text{MG-MAX-TEMP-STK-SIZE,}
\text{MG-WORD-SIZE,}
\text{\textit{run}))}
\]
\[
= \text{p-state (tag ('\textbf{pc},}
\text{\textbf{\textit{mg-simple-variable-assignment} . 1))},}
\text{push (p-frame (list (cons ('\textbf{dest},}
\text{\textbf{value (car (call-actuals (\textit{stmt}))),}
\text{\textbf{bindings (top (\textit{ctrl-stk})))}},
\text{\textbf{cons ('\textbf{source},}
\text{\textbf{value (cadr (call-actuals (\textit{stmt}))),}
\text{\textbf{bindings (top (\textit{ctrl-stk})))}},
\text{tag ('\textbf{pc},}
\text{\textbf{cons (\textit{subr}, length (code (\textit{cinfo}))}
+ 3))},
\text{\textit{ctrl-stk}),}
\text{push (value (cadr (call-actuals (\textit{stmt}))),}
\text{\textbf{bindings (top (\textit{ctrl-stk})))},}
\text{map-down-values (mg-alist (\textit{mg-state}),}
\text{\textbf{bindings (top (\textit{ctrl-stk}))),}
\text{\textit{temp-stk})),}
\text{translate-proc-list (\textit{proc-list}),}
\text{list (list ('c-c,}
\text{\textit{mg-cond-to-p-nat (cc (\textit{mg-state}), \textit{t-cond-list}))),}
\text{MG-MAX-CTRL-STK-SIZE,}
\text{MG-MAX-TEMP-STK-SIZE,}
\text{MG-WORD-SIZE,}
\text{\textit{run}))}
\]

9
Theorem: mg-simple-variable-assignment-step-5

\[ \neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk}))) \]
\[ \land (\text{car} (\text{stmt}) = \text{'predefined-proc-call-mg}) \]
\[ \land (\text{call-name} (\text{stmt}) = \text{'mg-simple-variable-assignment}) \]
\[ \land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \]
\[ \land \text{ok-mg-def-plistp} (\text{proc-list}) \]
\[ \land \text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list}) \]
\[ \land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list}))) = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \]
\[ \land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \]
\[ \land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \]
\[ \land \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist}) \]
\[ \land \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk}) \]
\[ \land \text{normal} (\text{mg-state}) \]
\[ \rightarrow \text{p-step} (\text{p-state} (\text{tag} ('\text{pc}, '\text{mg-simple-variable-assignment} . 1)), \text{push} (\text{p-frame} (\text{list} (\text{cons} ('\text{dest}, \text{value} (\text{car} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk})))))), \text{cons} ('\text{source}, \text{value} (\text{cadr} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk})))), \text{tag} ('\text{pc}, \text{cons} (\text{subr}, \text{length} (\text{code} (\text{cinfo})) + 3))), \text{ctrl-stk}), \text{push} (\text{value} (\text{cadr} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk})))), \text{map-down-values} (\text{mg-alist} (\text{mg-state}), \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk})), \text{translate-proc-list} (\text{proc-list}), \text{list} (\text{list} ('\text{c-c}, \text{mg-cond-to-p-nat} (\text{cc} (\text{mg-state}), \text{t-cond-list}))), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}, \text{'run})) = \text{p-state} (\text{tag} ('\text{pc}, \text{10}) \right)
'(mg-simple-variable-assignment . 2)),
push (p-frame (list (cons ('dest,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  cons ('source,
    value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  tag ('pc,
    cons (subr, length (code (cinfo))
      + 3)),
  ctrl-stk),
push (rget (untag (value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)),
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))

;; (push-local dest)

**THEOREM:** mg-simple-variable-assignment-step-6

\[\neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk}))) \]
\[\land (\text{car} (\text{stmt}) = \text{'predefined-proc-call-mg}) \]
\[\land (\text{call-name} (\text{stmt}) = \text{'mg-simple-variable-assignment}) \]
\[\land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \]
\[\land \text{ok-mg-def-plistp} (\text{proc-list}) \]
\[\land \text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list}) \]
\[\land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}, \text{proc-list})))
  = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})),
    \text{code2})) \]
\[\land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \]
\[\land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \]
∀ signatures-match (mg-alist (mg-state), name-alist)
∀ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)
∀ normal (mg-state)
→ (p-step (p-state (tag ('pc,
  '(mg-simple-variable-assignment . 2)),
  push (p-frame (list (cons ('dest,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
    cons ('source,
    value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
    tag ('pc,
    cons (subr, length (code (cinfo))
    + 3))),(ctrl-stk),
  push (rget (untag (value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
  map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)),
  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,
  '(mg-simple-variable-assignment . 3)),
  push (p-frame (list (cons ('dest,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
    cons ('source,
    value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))),
    tag ('pc,
    cons (subr, length (code (cinfo))
    + 3))),(ctrl-stk),
push (value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
push (rget (untag (value (cadr (call-actuals (stmt))),
   bindings (top (ctrl-stk)))),
map-down-values (mg-alist (mg-state),
   bindings (top (ctrl-stk)),
   temp-stk)),
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))

;; (deposit-temp-stk)

THEOREM: mg-simple-variable-assignment-step-7
(~ resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))
∧ (car (stmt) = 'predefined-proc-call-mg)
∧ (call-name (stmt) = 'mg-simple-variable-assignment)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-mg-def-plistp (proc-list)
∧ ok-mg-statetp (mg-state, r-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)
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
   bindings (top (ctrl-stk)),
   temp-stk)
∧ normal (mg-state))
→ (p-step (p-state (tag ('pc,
   'mg-simple-variable-assignment . 3)),
push (p-frame (list (cons ('dest,
   value (car (call-actuals (stmt)))),
   push (value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
push (rget (untag (value (cadr (call-actuals (stmt))),
   bindings (top (ctrl-stk)))),
map-down-values (mg-alist (mg-state),
   bindings (top (ctrl-stk)),
   temp-stk)),
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))

;; (deposit-temp-stk)
bindings (top (\textit{ctrl-stk}))),
\texttt{cons (\textquoteleft \texttt{source},}
\texttt{ value (cadr (call-actuals (\texttt{stmt}))),}
\texttt{ bindings (top (\textit{ctrl-stk}))))),
\texttt{tag (\textquoteleft \texttt{pc},}
\texttt{ cons (\texttt{subr}, length (code (\texttt{cinfo}))}
\texttt{ + 3))),}
\texttt{\textit{ctrl-stk}),}
push (value (car (call-actuals (\texttt{stmt}))),
\texttt{ bindings (top (\textit{ctrl-stk}))),}
push (rget (untag (value (cadr (call-actuals (\texttt{stmt}))),
\texttt{ bindings (top (\textit{ctrl-stk}))))),
\texttt{map-down-values (mg-alist (\texttt{mg-state}),}
\texttt{ bindings (top (\textit{ctrl-stk})),}
\texttt{ \textit{temp-stk}))),
\texttt{\texttt{map-down-values (mg-alist (\texttt{mg-state}),}
\texttt{ bindings (top (\textit{ctrl-stk})),}
\texttt{ \textit{temp-stk}))),}
\texttt{translate-proc-list (\texttt{proc-list}),}
\texttt{list (list (\textquoteleft \texttt{c-c},}
\texttt{ mg-cond-to-p-nat (cc (\texttt{mg-state}), \texttt{t-cond-list}))),}
\texttt{MG-MAX-CTRL-STK-SIZE,}
\texttt{MG-MAX-TEMP-STK-SIZE,}
\texttt{MG-WORD-SIZE,}
\texttt{\textquoteleft \texttt{run}}})
\texttt{= p-state (tag (\textquoteleft \texttt{pc},}
\texttt{ \textquoteleft \texttt{(mg-simple-variable-assignment . 4)}),}
push (p-frame (list (cons (\textquoteleft \texttt{dest},
\texttt{ value (car (call-actuals (\texttt{stmt}))),}
\texttt{ bindings (top (\textit{ctrl-stk}))))),
\texttt{cons (\textquoteleft \texttt{source},}
\texttt{ value (cadr (call-actuals (\texttt{stmt}))),}
\texttt{ bindings (top (\textit{ctrl-stk}))))),
tag (\textquoteleft \texttt{pc},}
\texttt{ cons (\texttt{subr}, length (code (\texttt{cinfo}))}
\texttt{ + 3))),}
\texttt{\textit{ctrl-stk}),}
rput (rget (untag (value (cadr (call-actuals (\texttt{stmt}))),
\texttt{ bindings (top (\textit{ctrl-stk}))))),
\texttt{map-down-values (mg-alist (\texttt{mg-state}),}
\texttt{ bindings (top (\textit{ctrl-stk})),}
\texttt{ \textit{temp-stk}))),
\texttt{untag (value (car (call-actuals (\texttt{stmt}))),}
\texttt{ bindings (top (\textit{ctrl-stk})))},
14
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))

;; (ret)

**THEOREM:** mg-simple-variable-assignment-step-8

\((n \neq 0)\)
\(\land (\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}\)
\(\land (\text{car (stmt) = ’predefined-proc-call-mg})\)
\(\land (\text{call-name (stmt) = ’mg-simple-variable-assignment})\)
\(\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land \text{ok-mg-def-plistp (proc-list)}\)
\(\land \text{ok-mg-statep (mg-state, r-cond-list)}\)
\(\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)})
   = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)})\)
\(\land \text{user-defined-procp (subr, proc-list)}\)
\(\land \text{listp (ctrl-stk)}\)
\(\land \text{all-cars-unique (mg-alist (mg-state))}\)
\(\land \text{signatures-match (mg-alist (mg-state), name-alist)}\)
\(\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}\)
\(\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\)
\(\land \text{normal (mg-state)}\)
\(\rightarrow (\text{p-step (p-state (tag (’pc, ’(mg-simple-variable-assignment . 4)),
   push (p-frame (list (cons (’dest, value (car (call-actuals (stmt))),
                                   bindings (top (ctrl-stk))))),
            cons (’source, value (cadr (call-actuals (stmt))))),
        (’run))})),

15
bindings (top \(\text{ctrl-stk}\))),

\[
\text{tag ('pc,}
\]
\[
\quad \text{cons (subr, length (code (cinfo)) + 3))},
\]

\(\text{ctrl-stk})\),

\[
\text{rput (rget (untag (value (cadr (call-actuals (stmt)))))},
\]
\[
\quad \text{map-down-values (mg-alist (mg-state),}
\]
\[
\quad \quad \text{bindings (top (ctrl-stk)),}
\]
\[
\quad \quad \text{temp-stk))},
\]
\[
\text{untag (value (car (call-actuals (stmt)))))},
\]
\[
\text{map-down-values (mg-alist (mg-state),}
\]
\[
\quad \text{bindings (top (ctrl-stk)),}
\]
\[
\quad \text{temp-stk))},
\]

\text{translate-proc-list (proc-list)},

\[
\text{list (list ('c-c,}
\]
\[
\quad \text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},
\]
\[
\text{MG-MAX-CTRL-STK-SIZE,}
\]
\[
\text{MG-MAX-TEMP-STK-SIZE,}
\]
\[
\text{MG-WORD-SIZE,}
\]
\[
\text{'run})}
\]

\[
= \text{p-state (tag ('pc,}
\]
\[
\quad \text{cons (subr,}
\]
\[
\quad \quad \text{if normal (mg-meaning-r (stmt,}
\]
\[
\quad \quad \quad \text{proc-list,}
\]
\[
\quad \quad \quad \text{mg-state,}
\]
\[
\quad \quad \quad \text{n,}
\]
\[
\quad \quad \quad \text{list (length (temp-stk),}
\]
\[
\quad \quad \quad \quad \text{p-ctrl-stk-size (ctrl-stk))})}
\]
\[
\text{then length (code (translate (cinfo,}
\]
\[
\quad \text{t-cond-list,}
\]
\[
\quad \text{stmt,}
\]
\[
\quad \text{proc-list))})}
\]
\[
\text{else find-label (fetch-label (cc (mg-meaning-r (stmt,}
\]
\[
\quad \text{proc-list,}
\]
\[
\quad \text{mg-state,}
\]
\[
\quad \text{n,}
\]
\[
\quad \text{list (length (temp-stk),}
\]
\[
\quad \quad \text{p-ctrl-stk-size (ctrl-stk))},
\]
\[
\quad \text{label-alist (translate (cinfo,}
\]
\[
\quad \quad \text{t-cond-list,}
\]
\[
\quad \quad \text{stmt,}
\]
\[
\quad \quad \text{proc-list))}),}
\]
append (code (translate (cinfo, t-cond-list, stmt, proc-list)),

code2)) endif),

cntl-stk,

map-down-values (mg-alist (mg-meaning-r (stmt, proc-list, mg-state, n,

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

bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list ('c-c, mg-conds-to-p-nat (cc (mg-meaning-r (stmt, proc-list, mg-state,

n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))),

t-cond-list)),

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

THEOREM: mg-simple-variable-assignment-exact-time-lemma
((n ≠ 0)
∧ (¬ resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))
∧ (car (stmt) = 'predefined-proc-call-mg)
∧ (call-name (stmt) = 'mg-simple-variable-assignment)
∧ 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))

17
∧ 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))))
→ (p (map-down (mg-state, proc-list, ctrl-stk, temp-stk, tag ('pc, cons (subr, length (code (cinfo)))), t-cond-list),
  clock (stmt, proc-list, mg-state, n))
  = p-state (tag ('pc, cons (subr, cons (subr, if normal (mg-meaning-r (stmt, proc-list, mg-state, n),
  list (length (temp-stk),
  p-ctrl-stk-size (ctrl-stk))))
  then length (code (translate (cinfo, t-cond-list, stmt, proc-list)))
  else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n),
  list (length (temp-stk),
  p-ctrl-stk-size (ctrl-stk)))),
  label-alist (translate (cinfo, t-cond-list, stmt),
  proc-list)))))
append (code (translate (cinr, t-cond-list, stmt, proc-list)),

proc-list)));

code2) endif),

code2)]

ctrl-stk,

map-down-values (mg-alist (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),

bindings (top (ctrl-stk)),

t empirical (top (ctrl-stk)),

translate-proc-list (proc-list),

list (list ('c-c,

mg-cond-to-p-nat (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),

t-cond-list))],

MG-MAX-CTRL-STK-SIZE,

MG-MAX-TEMP-STK-SIZE,

MG-WORD-SIZE,

'run))

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

;;;;

;;;; EXACT-TIME LEMMA MG-SIMPLE-CONSTANT-ASSIGNMENT

;;;;

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

THEOREM: mg-simple-constant-assignment-args-simple-identifierps

(ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
  ∧ (car(stmt) = 'predefined-proc-call-mg)
  ∧ (call-name(stmt) = 'mg-simple-constant-assignment)
  ∧ ok-mg-statep (mg-state, r-cond-list)
  ∧ signatures-match (mg-alist (mg-state), name-alist))
→ simple-identifiertp (car (call-actuals (stmt)), mg-alist (mg-state))

THEOREM: mg-simple-constant-assignment-steps-1-2

19
\((\neg \text{resources-inadequatep}(\text{stmt}, \\
\quad \text{proc-list}, \\
\quad \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk}))))\) \\
\land (\text{car}(\text{stmt}) = ' \text{predefined-proc-call-mg}) \\
\land (\text{call-name}(\text{stmt}) = ' \text{mg-simple-constant-assignment}) \\
\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{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})) \\
\quad = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \\
\quad \text{code2})) \\
\land \text{user-defined-procp}(\text{subr}, \text{proc-list}) \\
\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \\
\quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
\quad \text{temp-stk}) \\
\land \text{normal}(\text{mg-state}) \\
\rightarrow (\text{p-step}(\text{p-step}(\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 \text{t-cond-list})))) \\
\quad = \text{p-state}(\text{tag}('\text{pc}, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo}))) + 2)), \\
\quad \text{ctrl-stk}, \\
\quad \text{push}(\text{mg-to-p-simple-literal}(\text{cadr}(\text{call-actuals}(\text{stmt}))), \\
\quad \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt}))), \\
\quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
\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{list}('\text{c-c}, \\
\quad \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}, \text{t-cond-list}))), \\
\quad \text{MG-MAX-CTRL-STK-SIZE}, \\
\quad \text{MG-MAX-TEMP-STK-SIZE}, \\
\quad \text{MG-WORD-SIZE}, \\
\quad '\text{run}))) \\
\)

;; (call mg-simple-constant-assignment)

**Theorem:** mg-simple-constant-assignment-step-3
\((\neg \text{resources-inadequatep}(\text{stmt}, \\
\quad \text{proc-list}, \\
\quad \text{20})\)
\[\text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))}\]
\[\land (\text{car (stmt) = 'predefined-proc-call-mg})\]
\[\land (\text{call-name (stmt) = 'mg-simple-constant-assignment})\]
\[\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list})\]
\[\land \text{ok-mg-def-plistp (proc-list})\]
\[\land \text{ok-mg-statep (mg-state, r-cond-list})\]
\[\land \text{(code (translate-def-body (assoc (subr, proc-list), proc-list}) = append (code (translate (cinfo, t-cond-list, stmt, proc-list}), code2))}\]
\[\land \text{user-defined-procp (subr, proc-list})\]
\[\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk})\]
\[\land \text{normal (mg-state})\]
\[\rightarrow (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 2), ctrl-stk),}\]
\[\text{push (mg-to-p-simple-literal (cadr (call-actuals (stmt))},\]
\[\text{push (value (car (call-actuals (stmt))},\]
\[\text{bindings (top (ctrl-stk))},\]
\[\text{map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)},\]
\[\text{temp-stk}))},\]
\[\text{translate-proc-list (proc-list),}\]
\[\text{list (list ('c-c,}\]
\[\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},\]
\[\text{MG-MAX-CTRL-STK-SIZE,}\]
\[\text{MG-MAX-TEMP-STK-SIZE,}\]
\[\text{MG-WORD-SIZE,}\]
\[\text{'run})}\]
\[= p-state (tag ('pc,}\]
\[\text{'(mg-simple-constant-assignment . 0))},\]
\[\text{push (p-frame (list (cons ('dest,}\]
\[\text{value (car (call-actuals (stmt))},\]
\[\text{bindings (top (ctrl-stk))}),}\]
\[\text{cons ('source,}\]
\[\text{mg-to-p-simple-literal (cadr (call-actuals (stmt))))),}\]
\[\text{tag ('pc,}\]
\[\text{cons (subr, length (code (cinfo)) + 3)}))},\]
\[\text{ctrl-stk}),}\]
\[\text{map-down-values (mg-alist (mg-state),}\]
\[\text{bindings (top (ctrl-stk)},\]
\[\text{temp-stk}),}\]
\[\text{translate-proc-list (proc-list),}\]
Theorem: mg-simple-constant-assignment-steps-4-5

\[((\neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, list (length (\text{temp-stk}), p-ctrl-stk-size (\text{ctrl-stk})))) \land (\text{car (stmt) = 'predefined-proc-call-mg}) \land (\text{call-name (stmt) = 'mg-simple-constant-assignment}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \land \text{ok-mg-def-plistp (proc-list}) \land (\text{code (translate-def-body (assoc (subr, \text{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{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk}) \land \text{normal (mg-state)}) \to (\text{p-step (p-step (p-state (tag ('pc, ' (mg-simple-constant-assignment , 0))}, push (p-frame (list (cons ('dest, value (car (call-actuals (stmt), bindings (top \text{ctrl-stk}))), cons ('source, mg-to-p-simple-literal (cadr (call-actuals (stmt))))), tag ('pc, cons (subr, length (code (cinfo)) + 3)))), \text{ctrl-stk}), map-down-values (mg-alist (mg-state), bindings (top (\text{ctrl-stk}), temp-stk}), translate-proc-list (proc-list),\text{22})}
Theorem: mg-simple-constant-assignment-step-6

\[
\neg \text{resources-inadequatep}(stmt, \\
\quad \text{proc-list}, \\
\quad \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk})))
\land \ (\text{car}(stmt) = \text{'predefined-proc-call-mg})
\land \ (\text{call-name}(stmt) = \text{'mg-simple-constant-assignment})
\land \ \text{ok-mg-statement}(stmt, \text{r-cond-list}, \text{name-alist}, \text{proc-list})
\land \ \text{ok-mg-def-plistp}(\text{proc-list})
\land \ \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list})
\]
\[\begin{align*}
&\wedge (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)}) \\
&\quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)}) \\
&\wedge \text{user-defined-procp (subr, proc-list)} \\
&\wedge \text{all-cars-unique (mg-alist (mg-state))} \\
&\wedge \text{signatures-match (mg-alist (mg-state), name-alist)} \\
&\wedge \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)} \\
&\wedge \text{normal (mg-state)} \\
&\rightarrow (\text{p-step (p-state (tag ('pc, 'mg-simple-constant-assignment . 2)), push (p-frame (list (cons ('dest, value (car (call-actuals (stmt)), bindings (top (ctrl-stk))))), cons ('source, mg-to-p-simple-literal (cadr (call-actuals (stmt))))), tag ('pc, cons (subr, length (code (cinfo)) + 3))))), ctrl-stk), push (value (car (call-actuals (stmt))), bindings (top (ctrl-stk))), push (mg-to-p-simple-literal (cadr (call-actuals (stmt)))), map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)), translate-proc-list (proc-list), list (list ('c-c, mg-cond-to-p-nat (cc (mg-state), t-cond-list))), MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, 'run)) \\
&= \text{p-state (tag ('pc, 'mg-simple-constant-assignment . 3)), push (p-frame (list (cons ('dest, value (car (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('source, mg-to-p-simple-literal (cadr (call-actuals (stmt))))), tag ('pc, cons (subr, length (code (cinfo)) + 3))},
\end{align*}\]
\(\text{ctrl-stk},\)
\(\text{rput (mg-to-p-simple-literal (cadr (call-actuals (stmt)))},\)
\(\text{untag (value (car (call-actuals (stmt))},\)
\(\text{bindings (top (ctrl-stk))}),\)
\(\text{map-down-values (mg-alist (mg-state),}\)
\(\text{bindings (top (ctrl-stk))},\)
\(\text{temp-stk))},\)
\(\text{translate-proc-list (proc-list),}\)
\(\text{list (list ('c-c,}\)
\(\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},\)
\(\text{MG-MAX-CTRL-STK-SIZE,}\)
\(\text{MG-MAX-TEMP-STK-SIZE,}\)
\(\text{MG-WORD-SIZE,}\)
\(\text{'run))}\)

;; (ret)

**Theorem:** mg-simple-constant-assignment-step-7

\(\((n \not\equiv 0)\)
\(\land (\neg \text{resources-inadequatep (stmt,}\)
\(\quad \text{proc-list,}\)
\(\quad \text{list (length (temp-stk),}\)
\(\quad \quad \text{p-ctrl-stk-size (ctrl-stk)}))\)
\(\land (\text{car (stmt) = 'predefined-proc-call-mg})\)
\(\land (\text{call-name (stmt) = 'mg-simple-constant-assignment})\)
\(\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land \text{ok-mg-def-plistp (proc-list)}\)
\(\land \text{ok-mg-statep (mg-state, r-cond-list)}\)
\(\land \text{listp (ctrl-stk)}\)
\(\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}\)
\(\quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),}\)
\(\quad \quad \text{code2)})\)
\(\land \text{user-defined-procp (subr, proc-list)}\)
\(\land \text{all-cars-unique (mg-alist (mg-state))}\)
\(\land \text{signatures-match (mg-alist (mg-state), name-alist)}\)
\(\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),}\)
\(\quad \text{bindings (top (ctrl-stk)),}\)
\(\quad \text{temp-stk})\)
\(\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\)
\(\land \text{normal (mg-state))}\)
\(\rightarrow (\text{p-step (p-state (tag ('pc,}\)
\(\quad ('\text{mg-simple-constant-assignment . 3})),}\)
\(\quad \text{push (p-frame (list (cons ('dest,})))})\)

25
value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk))),
  cons ('source,
    mg-to-p-simple-literal (cadr (call-actuals (stmt)))),
  tag ('pc,
    cons (subr, length (code (cinfo)) + 3)),
  ctrl-stk),
rput (mg-to-p-simple-literal (cadr (call-actuals (stmt))),
  untag (value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk))),
  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 (subr,
    if normal (mg-meaning-r (stmt,
      proc-list,
      mg-state,
      n,
      list (length (temp-stk),
        p-ctrl-stk-size (ctrl-stk)))))
then length (code (translate (cinfo,
      t-cond-list,
      stmt,
      proc-list)))
else find-label (fetch-label (cc (mg-meaning-r (stmt,
      proc-list,
      mg-state,
      n,
      list (length (temp-stk),
        p-ctrl-stk-size (ctrl-stk))))),
label-alist (translate (cinfo,
      t-cond-list,
      stmt,
      proc-list))),
append (code (translate (cinfo,
Theorem: mg-simple-constant-assignment-exact-time-lemma

\((n \not\equiv 0)\)
\(\wedge\) \(\neg\) resources-inadequatep \((\text{stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), \)
\(\wedge\) \((\text{car (stmt)} = '\text{predefined-proc-call-mg})\)
\(\wedge\) \((\text{call-name (stmt)} = '\text{mg-simple-constant-assignment})\)
\(\wedge\) ok-mg-statement \((\text{stmt, r-cond-list, name-alist, proc-list})\)
\(\wedge\) ok-mg-def-plistp \((\text{proc-list})\)
\(\wedge\) ok-translation-parameters \((\text{cinfo, t-cond-list, stmt, proc-list, code2})\)
\(\wedge\) ok-mg-statep \((\text{mg-state, r-cond-list})\)
\(\wedge\) cond-subsetp \((\text{r-cond-list, t-cond-list})\)
\(\wedge\) \(\text{(code (translate-def-body (assoc (subr, proc-list), proc-list))} =\)
\(\text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}\)
\(\wedge\) user-defined-procp \((\text{subr, proc-list})\)
\[\begin{align*}
&\land \text{plistp (temp-stk)} \\
&\land \text{listp (ctrl-stk)} \\
&\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)} \\
&\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
&\land \text{signatures-match (mg-alist (mg-state), name-alist)} \\
&\land \text{normal (mg-state)} \\
&\land \text{all-cars-unique (mg-alist (mg-state))} \\
&\land (\neg \text{resource-errorp (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}) \\
\rightarrow (p (\text{map-down (mg-state, proc-list, ctrl-stk, temp-stk, tag ('pc, cons (subr, length (code (cinfo)))))}, t-cond-list), \\
\text{clock (stmt, proc-list, mg-state, n)}) \\
= \text{p-state (tag ('pc, cons (subr, \\
\quad \text{if normal (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}) \\
\quad \text{then length (code (translate (cinfo, t-cond-list, stmt, proc-list)))}) \\
\quad \text{else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list)))))}
\end{align*}\]
append (code (translate (cinfo, t-cond-list, stmt, proc-list)),

code2) endif),
ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))),

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

mg-cond-to-p-nat (cc (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),

t-cond-list),

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

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

;;;;;; EXACT-TIME LEMMA MG-SIMPLE-CONSTANT-EQ
;;;;;;
;;;;;

THEOREM: mg-simple-variable-eq-args-simple-identifierps
(ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ (car (stmt) = ’predefined-proc-call-mg)
 ∧ (call-name (stmt) = ’mg-simple-variable-eq)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ signatures-match (mg-alist (mg-state), name-alist))
→ (boolean-identifierp (car (call-actuals (stmt)), mg-alist (mg-state))
 ∧ simple-identifierp (cadr (call-actuals (stmt)), mg-alist (mg-state))

29
\(\wedge \text{simple-identifierp (caddr (call-actuals (stmt)), mg-alist (mg-state))}\)

**THEOREM:** mg-simple-variable-eq-args-definedp

(\text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\wedge\) (\text{car (stmt) = 'predefined-proc-call-mg}\)
\(\wedge\) (\text{call-name (stmt) = 'mg-simple-variable-eq}\)
\(\wedge\) ok-mg-statemp (mg-state, r-cond-list)
\(\wedge\) signatures-match (mg-alist (mg-state), name-alist)
\(\rightarrow\) (\text{definedp (car (call-actuals (stmt)), mg-alist (mg-state))}\)
\(\wedge\) definedp (cadr (call-actuals (stmt)), mg-alist (mg-state))
\(\wedge\) definedp (caddr (call-actuals (stmt)), mg-alist (mg-state)))

**THEOREM:** mg-simple-variable-eq-steps-1-3

\(((n \neq 0)\)
\(\wedge\) (\text{\neg resources-inadequatep (stmt, proc-list},
\quad \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}\)
\(\wedge\) (\text{car (stmt) = 'predefined-proc-call-mg}\)
\(\wedge\) (\text{call-name (stmt) = 'mg-simple-variable-eq}\)
\(\wedge\) ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
\(\wedge\) ok-mg-def-plistp (proc-list)
\(\wedge\) ok-mg-statempep (mg-state, r-cond-list)
\(\wedge\) (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}\)
\(\quad\) = \append (\text{code (translate (cinfo, t-cond-list, stmt, proc-list))}, \text{code2})\)
\(\wedge\) user-defined-procp (subr, proc-list)
\(\wedge\) listp (ctrl-stk)
\(\wedge\) all-cars-unique (mg-alist (mg-state))
\(\wedge\) signatures-match (mg-alist (mg-state), name-alist)
\(\wedge\) mg-vars-list-ok-in-p-state (mg-alist (mg-state),
\quad \text{bindings (top (ctrl-stk)), temp-stk})
\(\wedge\) no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
\(\wedge\) normal (mg-state)
\(\rightarrow\) (p-step (p-step (p-step (map-down (mg-state,
\quad proc-list, ctrl-stk, temp-stk,
\quad tag ('pc, cons (subr, length (code (cinfo))))),
\quad t-cond-list))))\)
\(\quad\) = \p-state (tag ('pc, cons (subr, length (code (cinfo)) + 3)), 30)
\begin{align*}
& \text{\texttt{ctrl-stk}}, \\
& \text{push (value (caddr (call-actuals (\texttt{stmt}))),} \\
& \quad \text{bindings (top \texttt{(ctrl-stk)}),} \\
& \quad \text{push (value (cadr (call-actuals (\texttt{stmt}))),} \\
& \quad \quad \text{bindings (top \texttt{(ctrl-stk)}),} \\
& \quad \quad \text{push (value (car (call-actuals (\texttt{stmt}))),} \\
& \quad \quad \quad \text{bindings (top \texttt{(ctrl-stk)}),} \\
& \quad \quad \quad \text{map-down-values (mg-alist \texttt{(mg-state)},} \\
& \quad \quad \quad \quad \text{bindings (top \texttt{(ctrl-stk)}),} \\
& \quad \quad \quad \quad \text{temp-stk}})))), \\
& \text{translate-proc-list (proc-list),} \\
& \text{list (list ('c-c,} \\
& \quad \text{mg-cond-to-p-nat (cc \texttt{(mg-state), t-cond-list)},} \\
& \text{MG-MAX-CTRL-STK-SIZE,} \\
& \text{MG-MAX-TEMP-STK-SIZE,} \\
& \text{MG-WORD-SIZE,} \\
& \text{'run}))}
\end{align*}

\textbf{Theorem:} mg-simple-variable-eq-step-4
\((n \neq 0)\)
\(\land \quad (\lnot \text{resources-inadequatep (\texttt{stmt},}
\text{proc-list),}
\quad \text{list (length (temp-stk),}
\quad \quad \text{p-ctrl-stk-size (ctrl-stk))))\)
\(\land \quad (\text{car (\texttt{stmt}) = 'predefined-proc-call-mg})\)
\(\land \quad (\text{call-name (\texttt{stmt}) = 'mg-simple-variable-eq})\)
\(\land \quad \text{ok-mg-statement (\texttt{stmt}, r-cond-list, name-alist, proc-list)}\)
\(\land \quad \text{ok-mg-def-plistp (proc-list)}\)
\(\land \quad \text{ok-mg-statem (mg-state, r-cond-list)}\)
\(\land \quad (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}
\quad = \quad \text{append (code (translate (cinfo, t-cond-list, \texttt{stmt}, proc-list)),}
\quad \quad \text{code2)})\)
\(\land \quad \text{user-defined-procp (subr, proc-list)}\)
\(\land \quad \text{listp (ctrl-stk)}\)
\(\land \quad \text{all-cars-unique (mg-alist (mg-state))}\)
\(\land \quad \text{signatures-match (mg-alist (mg-state), name-alist)}\)
\(\land \quad \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),}
\quad \quad \text{bindings (top \texttt{(ctrl-stk)}),}
\quad \quad \text{temp-stk}}))\)
\(\land \quad \text{no-p-aliasing (bindings (top \texttt{(ctrl-stk)}), mg-alist (mg-state))}\)
\(\land \quad \text{normal (mg-state)}\)
\(\rightarrow \quad (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 3)),}
\text{ctrl-stk,}
\quad \text{push (value (caddr (call-actuals (\texttt{stmt})),}
\text{code2))})})\)

31
Theorem: mg-simple-variable-eq-step-5
\[
((n \not= 0) \land \neg \text{resources-inadequatep}(stmt, proc-list),
\]

\[
\text{bindings (top (ctrl-stk))},
\text{push (value (cadr (call-actuals (stmt))},
\text{bindings (top (ctrl-stk))},
\text{push (value (car (call-actuals (stmt))},
\text{bindings (top (ctrl-stk))},
\text{map-down-values (mg-alist (mg-state)},
\text{bindings (top (ctrl-stk))},
\text{temp-stk))})},
\text{translate-proc-list (proc-list),}
\text{list (list ('c-c,}
\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},
\text{MG-MAX-CTRL-STK-SIZE,}
\text{MG-MAX-TEMP-STK-SIZE,}
\text{MG-WORD-SIZE,}
\text{'run})}
\]

\[
= \text{p-state (tag ('pc, ('mg-simple-variable-eq . 0))},
\text{push (p-frame (list (cons ('ans,}
\text{value (car (call-actuals (stmt)}},
\text{bindings (top (ctrl-stk))})},
\text{cons ('x,}
\text{value (cadr (call-actuals (stmt)}},
\text{bindings (top (ctrl-stk))})},
\text{cons ('y,}
\text{value (caddr (call-actuals (stmt)}},
\text{bindings (top (ctrl-stk))})},
\text{tag ('pc,}
\text{cons (subr, length (code (cinfo))}
\text{+ 4))}),
\text{ctrl-stk}),
\text{map-down-values (mg-alist (mg-state)},
\text{bindings (top (ctrl-stk))},
\text{temp-stk)},
\text{translate-proc-list (proc-list),}
\text{list (list ('c-c,}
\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},
\text{MG-MAX-CTRL-STK-SIZE,}
\text{MG-MAX-TEMP-STK-SIZE,}
\text{MG-WORD-SIZE,}
\text{'run})}
\]

THEOREM: mg-simple-variable-eq-step-5
\[
((n \not= 0) \land \neg \text{resources-inadequatep}(stmt, proc-list),
\]

32
list (length (temp-stk),
  p-ctrl-stk-size (ctrl-stk))))
∧ (car (stmt) = 'predefined-proc-call-mg)
∧ (call-name (stmt) = 'mg-simple-variable-eq)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-mg-def-plistp (proc-list)
∧ ok-mg-statep (mg-state, r-cond-list)
∧ (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)
∧ listp (ctrl-stk)
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, (mg-simple-variable-eq . 0)),
     push (p-frame (list (cons ('ans,
                           value (car (call-actuals (stmt)),
                           bindings (top (ctrl-stk))))),
                   cons ('x,
                           value (cadr (call-actuals (stmt)),
                           bindings (top (ctrl-stk))))),
                   cons ('y,
                           value (caddr (call-actuals (stmt)),
                           bindings (top (ctrl-stk)))))),
     tag ('pc,
     cons (subr, length (code (cinfo))
     + 4)),
     ctrl-stk),
     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, (mg-simple-variable-eq . 1)),
  33
push (p-frame (list (cons (’ans,
  value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  cons (’x,
  value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  cons (’y,
  value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  tag (’pc,
  cons (subr, length (code (cinfo))
  + 4)),
  ctrl-stk),
push (value (cadr (call-actuals (stmt))),
  bindings (top (ctrl-stk))),
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: mg-simple-variable-eq-step-6
((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
  proc-list,
  list (length (temp-stk),
  p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = ’predefined-proc-call-mg)
 ∧ (call-name (stmt) = ’mg-simple-variable-eq)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (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)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, 'mg-simple-variable-eq . 1))
, push (p-frame (list (cons ('ans, value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk)))))),
  cons ('x, value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk)))))),
  cons ('y, value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  tag ('pc, cons (subr, length (code (cinfo))
  + 4))),
  ctrl-stk),
push (value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk)))),
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, 'mg-simple-variable-eq . 2)),
push (p-frame (list (cons ('ans, value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk)))))),
  cons ('x, value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk)))))),
  cons ('y, value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  tag ('pc, cons (subr, length (code (cinfo))
  + 4))),
35
\(\text{ctrl-stk},\)
\(\text{push (rget (untag (value (cadr (call-actuals (stmt))))},\)
\(\text{bindings (top (ctrl-stk)))),}\)
\(\text{push (value (cadr (call-actuals (stmt)))),}\)
\(\text{bindings (top (ctrl-stk)))),}\)
\(\text{map-down-values (mg-alist (mg-state),}\)
\(\text{bindings (top (ctrl-stk)),}\)
\(\text{temp-stk))},}\)
\(\text{map-down-values (mg-alist (mg-state),}\)
\(\text{bindings (top (ctrl-stk)),}\)
\(\text{temp-stk))},\)
\(\text{translate-proc-list (proc-list)},\)
\(\text{list (list (c-c,}\)
\(\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))}),\)
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

\text{THEOREM: mg-simple-variable-eq-step-7}
((n \neq 0)
\land \neg \text{resources-inadequatep (stmt,}\)
\land \text{list (length (temp-stk),}\)
\land \text{p-ctrl-stk-size (ctrl-stk))})
\land (\text{car (stmt) = 'predefined-proc-call-mg})
\land (\text{call-name (stmt) = 'mg-simple-variable-eq})
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list})
\land \text{ok-mg-def-plistp (proc-list)\)
\land \text{ok-mg-statem (mg-state, r-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{user-defined-procp (subr, proc-list)\)
\land \text{listp (ctrl-stk)}\)
\land \text{all-cars-unique (mg-alist (mg-state))\)
\land \text{signatures-match (mg-alist (mg-state), name-alist)\}
\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),}\)
\land \text{bindings (top (ctrl-stk)),}\)
\land \text{temp-stk))\)
\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\)
\land \text{normal (mg-state))}
\rightarrow (\text{p-step (p-state (tag ('pc, '(mg-simple-variable-eq . 2)),}\)
\land \text{push (p-frame (list (cons ('ans,\)

36
value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons (’x,
  value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons (’y,
  value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
tag (’pc,
  cons (subr, length (code (cinfo))
        + 4))),
ctrl-stk),
push (rget (untag (value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
push (value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk)))),
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)),
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, (mgsimple-variable-eq . 3)),
push (p-frame (list (cons (’ans,
  value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  cons (’x,
  value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
  cons (’y,
  value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
tag (’pc,
  cons (subr, length (code (cinfo))
        + 4))),
ctrl-stk),
push (value (caddr (call-actuals (stmt))))
Theorem: mg-simple-variable-eq-step-8
\[ (n \neq 0) \land (\neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list}, \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk})))) \land (\text{car}(\text{stmt}) = \text{predefined-proc-call-mg}) \land (\text{call-name}(\text{stmt}) = \text{mg-simple-variable-eq}) \land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \land \text{ok-mg-def-plistp}(\text{proc-list}) \land \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list}) \land (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})) = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \land \text{user-defined-procp}(\text{subr}, \text{proc-list}) \land \text{listp}(\text{ctrl-stk}) \land \text{all-cars-unique}(\text{mg-alist}(\text{mg-state})) \land \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk}) \land \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state})) \land \text{normal}(\text{mg-state}) \rightarrow \text{(p-step}(\text{p-state}(\text{tag}(\text{pc}, '(\text{mg-simple-variable-eq . 3})), \text{push}(\text{p-frame}(\text{list}(\text{cons}('\text{ans,})))))) \]
value (car (call.actuals (stmt)),
  bindings (top (ctrl-stk)))),
cons ('x,
  value (cadr (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
cons ('y,
  value (caddr (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
tag ('pc,
  cons (subr, length (code (cinfo))
   + 4))),
ctrl-stk),
push (value (caddr (call.actuals (stmt))),
  bindings (top (ctrl-stk))),
push (rget (untag (value (cadr (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
push (value (cadr (call.actuals (stmt))),
  bindings (top (ctrl-stk))),
  map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)),
  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, '(mg-simple-variable-eq . 4)),
push (p-frame (list (cons ('ans,
  value (car (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
  cons ('x,
  value (cadr (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
  cons ('y,
  value (caddr (call.actuals (stmt))),
  bindings (top (ctrl-stk)))),
tag ('pc,
  cons (subr, length (code (cinfo))
   + 4))),
39
Theorem: mg-simple-variable-eq-step-9
\[(n \neq 0)\wedge (\neg \text{resources-inadequatep}(\text{stmt},\text{proc-list}), \text{list} (\text{length}(\text{tempstk}), \text{p-ctrl-stk-size}(\text{ctrl-stk}))))\wedge (\text{car}(\text{stmt}) = \text{'$predefined-proc-call-mg'$})\wedge (\text{call-name}(\text{stmt}) = \text{'$mg-simple-variable-eq'$})\wedge \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})\wedge \text{ok-mg-def-plistp}(\text{proc-list})\wedge \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list})\wedge (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))) = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2}))\wedge \text{user-defined-procp}(\text{subr}, \text{proc-list})\wedge \text{listp}(\text{ctrl-stk})\wedge \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))\wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})\wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk})\wedge \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))\]
\[ \begin{align*}
\wedge \text{normal}(mg-state) \\
\rightarrow (p \text{-step}(p-state \text{ tag } (\text{'pc}' (mg-simple-variable-eq . 4))), \\
\quad \text{push } (p-frame \text{ list } (\text{cons } (\text{'ans}, \\
\quad \quad \quad \text{value } (\text{car } (\text{call-actuals } (stmt))))), \\
\quad \quad \quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{cons } (\text{'x}, \\
\quad \quad \quad \text{value } (\text{cadr } (\text{call-actuals } (stmt))))), \\
\quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{cons } (\text{'y}, \\
\quad \quad \quad \text{value } (\text{caddr } (\text{call-actuals } (stmt))))), \\
\quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{tag } (\text{'pc}, \\
\quad \quad \text{cons } (\text{subr}, \text{length } (\text{code } (cinfo)) \\
\quad \quad \quad + \ 4))), \\
\quad \text{ctrl-stk}), \\
\quad \text{push } (rget (\text{untag } (\text{value } (\text{caddr } (\text{call-actuals } (stmt))))), \\
\quad \quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{map-down-values } (\text{mg-alist } (mg-state), \\
\quad \quad \text{bindings } (\text{top } (temp-stk)))), \\
\quad \text{push } (rget (\text{untag } (\text{value } (\text{cadr } (\text{call-actuals } (stmt))))), \\
\quad \quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{map-down-values } (\text{mg-alist } (mg-state), \\
\quad \quad \text{bindings } (\text{top } (temp-stk)))), \\
\quad \text{map-down-values } (\text{mg-alist } (mg-state), \\
\quad \quad \text{bindings } (\text{top } (ctrl-stk)), \\
\quad \quad \text{temp-stk)))), \\
\quad \text{translate-proc-list } (proc-list), \\
\quad \text{list } (\text{'c-c}, \\
\quad \quad \text{mg-cond-to-p-nat } (cc (mg-state), t-cond-list)), \\
\quad \text{MG-MAX-CTRL-STK-SIZE}, \\
\quad \text{MG-MAX-TEMP-STK-SIZE}, \\
\quad \text{MG-WORD-SIZE}, \\
\quad \text{'run})) \\
= p-state (\text{tag } (\text{'pc}' (mg-simple-variable-eq . 5))), \\
\quad \text{push } (p-frame \text{ list } (\text{cons } (\text{'ans}, \\
\quad \quad \text{value } (\text{car } (\text{call-actuals } (stmt))))), \\
\quad \quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{cons } (\text{'x}, \\
\quad \quad \text{value } (\text{cadr } (\text{call-actuals } (stmt))))), \\
\quad \text{bindings } (\text{top } (ctrl-stk)))), \\
\quad \text{cons } (\text{'y}, \\
\quad \quad \text{value } (\text{caddr } (\text{call-actuals } (stmt))))), \\
\end{align*} \]
Theorem: mg-simple-variable-eq-step-10
((n \neq 0)
\land \lnot \text{resources-inadequatep}(\text{stmt},
\text{proc-list},
\text{list}(\text{length}(\text{temp-stk}),
\text{p-ctrl-stk-size}(\text{ctrl-stk}))))
\land (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg})
\land (\text{call-name}(\text{stmt}) = \text{'mg-simple-variable-eq})
\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{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))
= \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})),
\text{code2})))
\land \text{user-defined-procp}(\text{subr}, \text{proc-list})
\land \text{listp}(\text{ctrl-stk})
\land \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))
\land \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})
\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})
\[\begin{align*}
\land & \text{ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
\land & \text{ normal (mg-state)} \\
\rightarrow & (\text{p-step (p-state (tag ('pc, (mg-simple-variable-eq . 5))), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt)), bindings (top (ctrl-stk))))), cons ('x, value (cadr (call-actuals (stmt)), bindings (top (ctrl-stk))))), cons ('y, value (caddr (call-actuals (stmt)), bindings (top (ctrl-stk)))))), tag ('pc, cons (subr, length (code (cinfo)) + 4))), ctrl-stk), push (tag ('bool, eq-value), map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)), translate-proc-list (proc-list), list (list ('c-c, mg-cond-to-p-nat (cc (mg-state), t-cond-list))), MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, \text{'}run)) \\
& = \text{p-state (tag ('pc, (mg-simple-variable-eq . 6)), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt)), bindings (top (ctrl-stk))))), cons ('x, value (cadr (call-actuals (stmt)), bindings (top (ctrl-stk))))), cons ('y, value (caddr (call-actuals (stmt)), bindings (top (ctrl-stk))))), tag ('pc, cons (subr, length (code (cinfo)) + 4))), ctrl-stk), push (value (car (call-actuals (stmt)), bindings (top (ctrl-stk))))), push (tag ('bool, eq-value)}, \text{'}run)) \\
\end{align*}\]
Theorem: mg-simple-variable-eq-step-11

\[(n \neq 0) \land \neg \text{resources-inadequate}(\text{stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))} \land (\text{car (stmt) = 'predefined-proc-call-mg}) \land (\text{call-name (stmt) = 'mg-simple-variable-eq}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list) \land ok-mg-def-plistp (proc-list)} \land \text{ok-mg-statep (mg-state, r-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{listp (ctrl-stk)} \land \text{all-cars-unique (mg-alist (mg-state))} \land \text{signatures-match (mg-alist (mg-state), name-alist)} \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{normal (mg-state))} \rightarrow (\text{p-step (p-state (tag ('pc, ('mg-simple-variable-eq . 6)), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('x, value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('y, value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk))))})}
tag ('pc,
   cons (subr, length (code (cinfo))
        + 4)),
ctrl-stk),
push (value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
push (tag ('bool, eq-value),
   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, '(mg-simple-variable-eq . 7)),
push (p-frame (list (cons ('ans,
           value (car (call-actuals (stmt))),
           bindings (top (ctrl-stk))))),
   cons ('x,
           value (cadr (call-actuals (stmt))),
           bindings (top (ctrl-stk))))),
   cons ('y,
           value (caddr (call-actuals (stmt))),
           bindings (top (ctrl-stk))))),
tag ('pc,
   cons (subr, length (code (cinfo))
        + 4)),
ctrl-stk),
rput (tag ('bool, eq-value),
untag (value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk)))),
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))

45
THEOREM: mg-bool-ok-boolean
ok-mg-valuep (mg-bool (x), 'boolean-mg)

THEOREM: mg-simple-variable-eq-step-12-true-case
((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
      proc-list,
      list (length (temp-stk),
            p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-simple-variable-eq)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statem (mg-state, r-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)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                               bindings (top (ctrl-stk)),
                               temp-stk)
 ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ∧ normal (mg-state)
 ∧ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))))
      = untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                               mg-alist (mg-state)))))
 → (p-step (p-state (tag ('pc , 'mg-simple-variable-eq . 7)),
push (p-frame (list (cons ('ans,
                            value (car (call-actuals (stmt)),
                                 bindings (top (ctrl-stk))))),
                      cons ('x,
                            value (cadr (call-actuals (stmt)),
                                 bindings (top (ctrl-stk))))),
                      cons ('y,
                            value (caddr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                      tag ('pc,
                           cons (subr, length (code (cinfo))
                                 + 4)))))
 ctrl-stk),
rput (tag ('bool, 't),
    untag (value (car (call-actuals (stmt)));
            bindings (top (ctrl-stk)));
    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 (subr,
            if normal (mg-meaning-r (stmt,
                proc-list,
                mg-state,
                n,
                list (length (temp-stk),
                    p-ctrl-stk-size (ctrl-stk)))))
then length (code (translate (cinfo,
                t-cond-list,
                stmt,
                proc-list))))
else find-label (fetch-label (cc (mg-meaning-r (stmt,
                proc-list,
                mg-state,
                n,
                list (length (temp-stk),
                    p-ctrl-stk-size (ctrl-stk))));
    label-alist (translate (cinfo,
                t-cond-list,
                stmt,
                proc-list)),
    append (code (translate (cinfo,
                t-cond-list,
                stmt,
                proc-list)));
    code2)) endif),
ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (stmt,
                proc-list,
                mg-state),

47
Theorem: mg-simple-variable-eq-step-12-false-case

\((n \not\equiv 0)\)
\(\land \ \neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}\)
\(\land \ (\text{car (stmt)} = \text{‘predefined-proc-call-mg})\)
\(\land \ (\text{call-name (stmt)} = \text{‘mg-simple-variable-eq})\)
\(\land \ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land \ \text{ok-mg-def-plistp (proc-list)}\)
\(\land \ \text{ok-mg-statp (mg-state, r-cond-list)}\)
\(\land \ (\text{code (translate-def-body (assoc (subr, proc-list, proc-list)) = append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))})\)
\(\land \ \text{user-defined-procp (subr, proc-list)}\)
\(\land \ \text{listp (ctrl-stk)}\)
\(\land \ \text{all-cars-unique (mg-alist (mg-state))}\)
\(\land \ \text{signatures-match (mg-alist (mg-state), name-alist)}\)
\(\land \ \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)}\)
\(\land \ \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\)
\(\land \ \text{normal (mg-state)}\)
\(\land \ (\text{untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state))))})\))
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals stmt)), mg-alist mg-state)))))

→ (p-step (p-state (tag ('pc, (mg-simple-variable-eq . 7))),(stmt, mg-state)))))
push (p-frame (list (cons ('ans, value (car (call-actuals stmt)), bindings (top (ctrl-stk))))),
   cons ('x, value (cadr (call-actuals stmt)), bindings (top (ctrl-stk))))),
   cons ('y, value (caddr (call-actuals stmt)), bindings (top (ctrl-stk))))),
tag ('pc,
   cons (subr, length (code (cinfo))
   + 4)),
   ctrl-stk),
rput (tag ('bool, 'f),
   untag (value (car (call-actuals stmt)), bindings (top (ctrl-stk))))),
   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 (subr,
      if normal (mg-meaning-r stmt, proc-list, mg-state, n, list (length temp-stk), p-ctrl-stk-size (ctrl-stk)))
then length (code (translate cinfo, t-cond-list, stmt, proc-list)))
else find-label (fetch-label (cc mg-meaning-r stmt, proc-list, mg-state),

49
\[ n, \]
\[ \text{list (length (temp-stk),} \]
\[ \text{p-ctrl-stk-size (ctrl-stk))}, \]
\[ \text{label-alist (translate (cinfo,} \]
\[ \text{t-cond-list,} \]
\[ \text{stmt,} \]
\[ \text{proc-list))}, \]
\[ \text{append (code (translate (cinfo,} \]
\[ \text{t-cond-list,} \]
\[ \text{stmt,} \]
\[ \text{proc-list))}, \]
\[ \text{code2})} \text{ endif}),} \]
\[ \text{ctrl-stk,} \]
\[ \text{map-down-values (mg-alist (mg-meaning-r (stmt,} \]
\[ \text{proc-list,} \]
\[ \text{mg-state,} \]
\[ n,} \]
\[ \text{list (length (temp-stk),} \]
\[ \text{p-ctrl-stk-size (ctrl-stk))},} \]
\[ \text{bindings (top (ctrl-stk))}, \]
\[ \text{temp-stk)}, \]
\[ \text{translate-proc-list (proc-list),} \]
\[ \text{list (list ('c-c,} \]
\[ \text{mg-cond-to-p-nat (cc (mg-meaning-r (stmt,} \]
\[ \text{proc-list,} \]
\[ \text{mg-state,} \]
\[ n,} \]
\[ \text{list (length (temp-stk),} \]
\[ \text{p-ctrl-stk-size (ctrl-stk))},} \]
\[ \text{t-cond-list))},} \]
\[ \text{MG-MAX-CTRL-STK-SIZE,} \]
\[ \text{MG-MAX-TEMP-STK-SIZE,} \]
\[ \text{MG-WORD-SIZE,} \]
\[ \text{'run})} \]

**Theorem:** mg-simple-variable-eq-exact-time-lemma
\[ ((n \not\approx 0) \]
\[ \land (\neg \text{resources-inadequatep (stmt,} \]
\[ \text{proc-list,} \]
\[ \text{list (length (temp-stk),} \]
\[ \text{p-ctrl-stk-size (ctrl-stk))})) \]
\[ \land (\text{car (stmt) = 'predefined-proc-call-mg}) \]
\[ \land (\text{call-name (stmt) = 'mg-simple-variable-eq}) \]
\[ \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \]
∧ ok-mg-def-plistp (proc-list)
∧ ok-mg-statep (mg-state, r-cond-list)
∧ (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)
∧ listp (ctrl-stk)
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
   bindings (top (ctrl-stk)),
   temp-stk)
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state))
→ (p (map-down (mg-state,
   proc-list,
   ctrl-stk,
   temp-stk,
   tag ('pc, cons (subr, length (code (cinfo)))),
   t-cond-list),
   clock (stmt, proc-list, mg-state, n))
   = p-state (tag ('pc,
      cons (subr,
         if normal (mg-meaning-r (stmt,
             proc-list,
             mg-state,
             n,
             list (length (temp-stk),
               p-ctrl-stk-size (ctrl-stk))))
         then length (code (translate (cinfo,
             t-cond-list,
             stmt,
             proc-list)))
         else find-label (fetch-label (cc (mg-meaning-r (stmt,
             proc-list,
             mg-state,
             n,
             list (length (temp-stk),
               p-ctrl-stk-size (ctrl-stk))))),
             label-alist (translate (cinfo,
               t-cond-list,
               stmt,
               proc-list)))),
   append (code (translate (cinfo,
\textbf{Theorem}: \texttt{mg-simple-constant-eq-args-simple-identifierps}

\begin{verbatim}
(\text{ok-mg-statement} (\textit{stmt}, r-cond-list, name-alist, proc-list)
\land (\text{car} (\textit{stmt}) = \texttt{\textquotesingle predefined-proc-call-mg\textquotesingle})
\land (\text{call-name} (\textit{stmt}) = \texttt{\textquotesingle mg-simple-constant-eq\textquotesingle})
\land \text{ok-mg-statep} (mg-state, r-cond-list)
\land \text{signatures-match} (mg-alist (mg-state), name-alist))
\rightarrow (\text{boolean-identifierp} (\text{car} (\text{call-actuals} (\textit{stmt}))),
mg-alist (mg-state))
\land \text{simple-identifierp} (\text{cadr} (\text{call-actuals} (\textit{stmt})),
mg-alist (mg-state))
\land \text{simple-typed-literalp} (\text{caddr} (\text{call-actuals} (\textit{stmt})),
mg-alist (mg-state)))
\end{verbatim}

52
Theorem: mg-simple-constant-eq-args-definedp

\[
\begin{align*}
\text{(ok-mg-statement} \; (\text{stmt, r-cond-list, name-alist, proc-list}) & \wedge
\text{(car} \; \text{stmt} \; = \text{'predefined-proc-call-mg}) \\
& \wedge
\text{(call-name} \; \text{stmt} \; = \text{'mg-simple-constant-eq}) \\
& \wedge
\text{ok-mg-statep} \; (\text{mg-state, r-cond-list}) \\
& \wedge
\text{signatures-match} \; (\text{mg-alist} \; (\text{mg-state}), \text{name-alist})) \\
\rightarrow
& \; \text{(definedp} \; (\text{car} \; \text{call-actuals} \; (\text{stmt})), \text{mg-alist} \; (\text{mg-state})) \\
& \wedge
\text{definedp} \; (\text{cadr} \; \text{call-actuals} \; (\text{stmt})), \text{mg-alist} \; (\text{mg-state}))
\end{align*}
\]

Theorem: mg-simple-constant-eq-steps-1-3

\[
\begin{align*}
((n \neq 0) & \wedge
\text{(\neg resources-inadequatep} \; (\text{stmt, proc-list, list (length} \; \text{temp-stk)}), \\
& \text{p-ctrl-stk-size} \; (\text{ctrl-stk})))) \\
& \wedge
\text{(car} \; \text{stmt} \; = \text{'predefined-proc-call-mg}) \\
& \wedge
\text{(call-name} \; \text{stmt} \; = \text{'mg-simple-constant-eq}) \\
& \wedge
\text{ok-mg-statement} \; (\text{stmt, r-cond-list, name-alist, proc-list}) \\
& \wedge
\text{ok-mg-def-plistp} \; (\text{proc-list}) \\
& \wedge
\text{ok-mg-statep} \; (\text{mg-state, r-cond-list}) \\
& \wedge
\text{(code} \; \text{translate-def-body} \; (\text{assoc} \; (\text{subr, proc-list}), \text{proc-list})) \\
& \; = \; \text{append} \; (\text{code} \; \text{translate} \; (\text{cinfo, t-cond-list, stmt, proc-list}), code2)) \\
& \wedge
\text{user-defined-procp} \; (\text{subr, proc-list}) \\
& \wedge
\text{listp} \; (\text{ctrl-stk}) \\
& \wedge
\text{all-cars-unique} \; (\text{mg-alist} \; (\text{mg-state})) \\
& \wedge
\text{signatures-match} \; (\text{mg-alist} \; (\text{mg-state}), \text{name-alist}) \\
& \wedge
\text{mg-vars-list-ok-in-p-state} \; (\text{mg-alist} \; (\text{mg-state}), \\
& \text{bindings} \; (\text{top} \; \text{ctrl-stk})), \text{temp-stk}) \\
& \wedge
\text{no-p-aliasing} \; (\text{bindings} \; (\text{top} \; \text{ctrl-stk})), \text{mg-alist} \; (\text{mg-state})) \\
& \wedge
\text{normal} \; (\text{mg-state})) \\
\rightarrow
& \; \text{(p-step} \; \text{(p-step} \; \text{(p-step} \; \text{map-down} \; (\text{mg-state, proc-list, ctrl-stk, temp-stk,}} \\
& \text{tag} \; (\text{'pc, cons} \; (\text{subr, length} \; \text{code} \; (\text{cinfo}))))), \\
& \text{t-cond-list)))) \\
& \; = \; \text{p-state} \; (\text{tag} \; (\text{'pc, cons} \; (\text{subr, length} \; \text{code} \; (\text{cinfo}) \; + \; 3)), \\
& \text{ctrl-stk})
\end{align*}
\]

53
THEOREM: mg-simple-constant-eq-step-4

\[
((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{car} \ (\text{stmt}) = \ '\text{predefined-proc-call-mg}')
\land \ (\text{call-name} \ (\text{stmt}) = \ '\text{mg-simple-constant-eq}')
\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{code} \ (\text{translate-def-body} \ (\text{assoc} \ (\text{subr}, \ \text{proc-list}), \ \text{proc-list})))
\quad = \ \text{append} \ (\text{code} \ (\text{translate} \ (\text{cinfo}, \ \text{t-cond-list}, \ \text{stmt}, \ \text{proc-list})), \ \text{code2}))
\land \ \text{user-defined-procp} \ (\text{subr}, \ \text{proc-list})
\land \ \text{listp} \ (\text{ctrl-stk})
\land \ \text{all-cars-unique} \ (\text{mg-alist} \ (\text{mg-state}))
\land \ \text{signatures-match} \ (\text{mg-alist} \ (\text{mg-state}), \ \text{name-alist})
\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{normal} \ (\text{mg-state})
\rightarrow \ (\text{p-step} \ (\text{p-state} \ (\text{tag} \ ('\text{pc}, \ \text{cons} \ (\text{subr}, \ \text{length} \ (\text{code} \ (\text{cinfo}))) + 3)), \ \text{ctrl-stk},
\quad \text{push} \ (\text{mg-to-p-simple-literal} \ (\text{caddr} \ \text{call-actuals} \ (\text{stmt}))),
\quad \text{push} \ (\text{value} \ (\text{cdr} \ \text{call-actuals} \ (\text{stmt})),
\quad \text{bindings} \ (\text{top} \ (\text{ctrl-stk}))))
\[p-state\left(\text{tag}\left(\text{\textquote{pc}}, \text{\textquote{mg-simple-constant-eq \ . 0}}\right),
\text{push}\left(\text{p-frame}\left(\text{list}\left(\text{cons}\left(\text{\textquote{ans}},
\text{value}\left(\text{car}\left(\text{call-actuals}\left(\text{stmt}\right)\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right)\right)\right),
\text{cons}\left(\text{\textquote{x}},
\text{value}\left(\text{cadr}\left(\text{call-actuals}\left(\text{stmt}\right)\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right)\right)\right),
\text{cons}\left(\text{\textquote{y}},
\text{mg-to-p-simple-literal}\left(\text{caddr}\left(\text{call-actuals}\left(\text{stmt}\right)\right)\right)\right)\right),
\text{tag}\left(\text{\textquote{pc}},
\text{cons}\left(\text{subr}, \text{length}\left(\text{code}\left(\text{cinfo}\right)\right) + 4\right)\right),
\text{ctrl-stk}\right),
\text{map-down-values}\left(\text{mg-alist}\left(\text{mg-state}\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right),
\text{temp-stk}\right)\right),
\text{translate-proc-list}\left(\text{proc-list}\right),
\text{list}\left(\text{\textquote{c-c},
\text{mg-cond-to-p-nat}\left(\text{cc}\left(\text{mg-state}, \text{t-cond-list}\right)\right)\right),
\text{MG-MAX-CTRL-STK-SIZE},
\text{MG-MAX-TEMP-STK-SIZE},
\text{MG-WORD-SIZE},
\text{\textquote{run}}\right)\right)\]

\[= \ p-state\left(\text{tag}\left(\text{\textquote{pc}}, \text{\textquote{mg-simple-constant-eq . 0}}\right),
\text{push}\left(\text{p-frame}\left(\text{list}\left(\text{cons}\left(\text{\textquote{ans}},
\text{value}\left(\text{car}\left(\text{call-actuals}\left(\text{stmt}\right)\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right)\right)\right),
\text{cons}\left(\text{\textquote{x}},
\text{value}\left(\text{cadr}\left(\text{call-actuals}\left(\text{stmt}\right)\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right)\right)\right),
\text{cons}\left(\text{\textquote{y}},
\text{mg-to-p-simple-literal}\left(\text{caddr}\left(\text{call-actuals}\left(\text{stmt}\right)\right)\right)\right)\right),
\text{tag}\left(\text{\textquote{pc}},
\text{cons}\left(\text{subr}, \text{length}\left(\text{code}\left(\text{cinfo}\right)\right) + 4\right)\right),
\text{ctrl-stk}\right),
\text{map-down-values}\left(\text{mg-alist}\left(\text{mg-state}\right),
\text{bindings}\left(\text{top}\left(\text{ctrl-stk}\right)\right),
\text{temp-stk}\right)\right),
\text{translate-proc-list}\left(\text{proc-list}\right),
\text{list}\left(\text{\textquote{c-c},
\text{mg-cond-to-p-nat}\left(\text{cc}\left(\text{mg-state}, \text{t-cond-list}\right)\right)\right),
\text{MG-MAX-CTRL-STK-SIZE},
\text{MG-MAX-TEMP-STK-SIZE},
\text{MG-WORD-SIZE},
\text{\textquote{run}}\right)\right)\]

**Theorem:** mg-simple-constant-eq-step-5

\[
\left(\left( n \neq 0 \right) \land \left( \neg \text{resources-inadequatep}\left( \text{stmt}, \text{proc-list}, \text{list}\left(\text{length}\left(\text{temp-stk}\right), \text{p-ctrl-stk-size}\left(\text{ctrl-stk}\right)\right)\right)\right)\right)
\land \left( \text{\textquote{car}}\left(\text{\textquote{stmt}}\right) = \text{\textquote{predefined-proc-call-mg}}\right)
\land \left( \text{\textquote{call-name}}\left(\text{\textquote{stmt}}\right) = \text{\textquote{mg-simple-constant-eq}}\right)
\]
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-mg-def-plistp (proc-list)
∧ ok-mg-statep (mg-state, r-cond-list)
∧ (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)
∧ listp (ctrl-stk)
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                           bindings (top (ctrl-stk)),
                                           temp-stk)
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, 'mg-simple-constant-eq . 0)),
         push (p-frame (list (cons ('ans,
                                  value (car (call-actuals (stmt))),
                                  bindings (top (ctrl-stk))))),
            cons ('x,
              value (cadr (call-actuals (stmt))),
              bindings (top (ctrl-stk))))),
         cons ('y,
           mg-to-p-simple-literal (caddr (call-actuals (stmt))))),
         tag ('pc,
           cons (subr, length (code (cinfo))
           + 4)),
         ctrl-stk),
         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, 'mg-simple-constant-eq . 1)),
         push (p-frame (list (cons ('ans,
                                  value (car (call-actuals (stmt))),
                                  bindings (top (ctrl-stk))))),
            cons ('x,
              value (cadr (call-actuals (stmt)))),
Theorem: mg-simple-constant-eq-step-6
\[(n \not= 0)\]
\[\land \neg \text{resources-inadequatep}(\text{stmt},\text{proc-list})\]
\[\land (\text{car}(\text{stmt}) = \text{predefined-proc-call-mg})\]
\[\land (\text{call-name}(\text{stmt}) = \text{mg-simple-constant-eq})\]
\[\land \text{ok-mg-statement}(\text{stmt},\text{r-cond-list},\text{name-alist},\text{proc-list})\]
\[\land \text{ok-mg-def-plistp}(\text{proc-list})\]
\[\land \text{ok-mg-statep}(\text{mg-state},\text{r-cond-list})\]
\[\land (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr},\text{proc-list}),\text{proc-list}))) = \text{append}(\text{code}(\text{translate}(\text{cinfo},\text{t-cond-list},\text{stmt},\text{proc-list})),\text{code2}))\]
\[\land \text{user-defined-procp}(\text{subr},\text{proc-list})\]
\[\land \text{listp}(\text{ctrl-stk})\]
\[\land \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))\]
\[\land \text{signatures-match}(\text{mg-alist}(\text{mg-state}),\text{name-alist})\]
\[\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}),\text{bindings}(\text{top}(\text{ctrl-stk})),\text{temp-stk})\]
\[\land \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})),\text{mg-alist}(\text{mg-state}))\]
\[\land \text{normal}(\text{mg-state})\]
\[\rightarrow (\text{p-step}(\text{p-state}(\text{tag}(\text{pc},\text{mg-simple-constant-eq . 1}))))\]
push (p-frame (list (cons (ans,
    value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons (x,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons (y,
    mg-to-p-simple-literal (caddr (call-actuals (stmt)))),
  tag (pc,
    cons (subr, length (code (cinfo))
        + 4)),
  ctrl-stk),
push (value (cadr (call-actuals (stmt))),
  bindings (top (ctrl-stk))),
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)),
translate-proc-list (proc-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, (mg-simple-constant-eq . 2)),
push (p-frame (list (cons (ans,
    value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons (x,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons (y,
    mg-to-p-simple-literal (caddr (call-actuals (stmt)))),
  tag (pc,
    cons (subr, length (code (cinfo))
        + 4)),
  ctrl-stk),
push (rget (untag (value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
push (value (cadr (call-actuals (stmt))),
  bindings (top (ctrl-stk))),
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk)),

58
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: mg-simple-constant-eq-step-7
((n \neq 0)
 ∧ (\neg resources-inadequatep (stmt,
                     proc-list,
                     list (length (temp-stk),
                           p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = ’predefined-proc-call-mg)
 ∧ (call-name (stmt) = ’mg-simple-constant-eq)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (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)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                        bindings (top (ctrl-stk)),
                                        temp-stk))
 ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ∧ normal (mg-state))
→ (p-step (p-state (tag (’pc, ’(mg-simple-constant-eq . 2)),
                  push (p-frame (list (cons (’ans,
                                    value (car (call-actuals (stmt)),
                                    bindings (top (ctrl-stk))))),
                               cons (’x,
                                     value (cadr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                               cons (’y,
                                     mg-to-p-simple-literal (caddr (call-actuals (stmt)))))),
              tag (’pc,

59
\[
\begin{align*}
&\text{cons (\textit{subr}, length (code (\textit{cinfo}))} \\
&\quad \quad \quad \quad \quad \quad \quad + \ 4)), \\
&\text{\textit{ctrl-stk}),} \\
&\text{push (rget (untag (value (cadr (call-actuals (\textit{stmt})))),} \\
&\quad \quad \quad \text{bindings (top (\textit{ctrl-stk})))),} \\
&\text{push (value (cadr (call-actuals (\textit{stmt})))),} \\
&\text{bindings (top (\textit{ctrl-stk}))},} \\
&\text{map-down-values (mg-alist (\textit{mg-state}),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})),} \\
&\quad \textit{temp-stk})),} \\
&\text{map-down-values (mg-alist (\textit{mg-state}),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})),} \\
&\quad \textit{temp-stk})),} \\
&\text{translate-proc-list (\textit{proc-list}),} \\
&\text{list (list ('c-c,} \\
&\quad \text{mg-cond-to-p-nat (cc (\textit{mg-state}), t-cond-list)),} \\
&\quad \text{mg-max-ctrl-stk-size,} \\
&\quad \text{mg-max-temp-stk-size,} \\
&\quad \text{mg-word-size,} \\
&\quad '\text{run})} \\
&= \text{p-state (tag ('pc, ' (\textit{mg-simple-constant-eq} . 3)),} \\
&\text{push (p-frame (list (cons ('\textit{ans},} \\
&\quad \text{value (car (call-actuals (\textit{stmt})))),} \\
&\quad \text{bindings (top (\textit{ctrl-stk}))))),} \\
&\quad \text{cons ('\textit{x},} \\
&\quad \text{value (cadr (call-actuals (\textit{stmt})))),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})))},} \\
&\quad \text{cons ('\textit{y},} \\
&\quad \text{mg-to-p-simple-literal (caddr (call-actuals (\textit{stmt}))))),} \\
&\text{tag ('pc,} \\
&\quad \text{cons (\textit{subr}, length (code (\textit{cinfo}))} \\
&\quad \quad \quad \quad \quad \quad \quad + \ 4))}, \\
&\text{\textit{ctrl-stk}),} \\
&\text{push (mg-to-p-simple-literal (caddr (call-actuals (\textit{stmt})))),} \\
&\text{push (rget (untag (value (cadr (call-actuals (\textit{stmt})))),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})))),} \\
&\text{push (value (cadr (call-actuals (\textit{stmt})))),} \\
&\text{bindings (top (\textit{ctrl-stk}))},} \\
&\text{map-down-values (mg-alist (\textit{mg-state}),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})),} \\
&\quad \textit{temp-stk})),} \\
&\text{map-down-values (mg-alist (\textit{mg-state}),} \\
&\quad \text{bindings (top (\textit{ctrl-stk})),} \\
&\quad \textit{temp-stk})),}
\end{align*}
\]
Theorem: mg-simple-constant-eq-step-8

\((n \neq 0)\)
\(\land\) (\(\neg\) resources-inadequatep \((stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))\)
\(\land\) (car \((stmt)\) = 'predefined-proc-call-mg)
\(\land\) (call-name \((stmt)\) = 'mg-simple-constant-eq)
\(\land\) ok-mg-statement \((stmt, r-cond-list, name-alist, proc-list)\)
\(\land\) ok-mg-def-plistp \((proc-list)\)
\(\land\) ok-mg-statep \((mg-state, r-cond-list)\)
\(\land\) (code \((\text{translate-def-body} \,(\text{assoc} \,(\text{subr}, proc-list), proc-list))\))
\(\land\) user-defined-procp \((\text{subr, proc-list})\)
\(\land\) listp \((\text{ctrl-stk})\)
\(\land\) all-cars-unique \((mg-alist \,(mg-state))\)
\(\land\) signatures-match \((mg-alist \,(mg-state), name-alist)\)
\(\land\) mg-vars-list-ok-in-p-state \((mg-alist \,(mg-state)), bindings \,(top \,(ctrl-stk)), temp-stk)\)
\(\land\) no-p-aliasing \((bindings \,(top \,(ctrl-stk)), mg-alist \,(mg-state))\)
\(\land\) normal \((mg-state)\)
\(\rightarrow\) (p-step \((p-state \,(\text{tag} \,(\text{\textquote Single}}} \text{\textquote Constant}}} \text{\textquote Eq}} . 3)), push \((p-frame \,(list \,(\text{cons} \,(\text{\textquote Ans}}, value \,(\text{\textquote Car} \,(\text{\textquote Call}}-\text{\textquote Actuals} \,(\text{\textquote Stmnt})), bindings \,(top \,(ctrl-stk)))))), \text{\textquote Cons} \,(\text{\textquote \textquote X}}, value \,(\text{\textquote Cadr} \,(\text{\textquote Call}}-\text{\textquote Actuals} \,(\text{\textquote Stmnt})), bindings \,(top \,(ctrl-stk)))))), \text{\textquote Cons} \,(\text{\textquote Y}}, mg-to-p-simple-literal \,(\text{\textquote Caddr} \,(\text{\textquote Call}}-\text{\textquote Actuals} \,(\text{\textquote Stmnt})))))\), tag \((\text{\textquote Pc}}, \text{\textquote Cons} \,(\text{\textquote Subr}, length \,(\text{\textquote Code} \,(\text{\textquote Cinfo})) + 4))\), ctrl-stk),
push (mg-to-p-simple-literal (caddr (call-actuals (stmt)))),
push (rget (untag (value (cadr (call-actuals (stmt))))),
  bindings (top (ctrl-stk))));
push (value (cadr (call-actuals (stmt)));
  bindings (top (ctrl-stk)));
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk));
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, (mg-simple-constant-eq . 4)),
push (p-frame (list (cons ('ans,
  value (car (call-actuals (stmt)));
  bindings (top (ctrl-stk))))),
  cons ('x,
  value (cadr (call-actuals (stmt)))
  bindings (top (ctrl-stk))));
  cons ('y,
  mg-to-p-simple-literal (caddr (call-actuals (stmt))));),
tag ('pc,
  cons (subr, length (code (cinfo))
  + 4)),

ctrl-stk),
push (tag ('bool,
  if untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state))))))
  = untag (mg-to-p-simple-literal (caddr (call-actuals (stmt))))
  then 't
  else 'f endif),
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,
Theorem: mg-simple-constant-eq-step-9

\((n \neq 0)\) \\
\(\land \) (resources-inadequate \((stmt, proc-list, \) \\
list \(\text{length}(temp-stk), \) \\
p-ctrl-stk-size \((ctrl-stk)\))) \\
\(\land\) (car \((stmt) = \text{'predefined-proc-call-mg}) \\
\(\land\) (call-name \((stmt) = \text{'mg-simple-constant-eq}) \\
\(\land\) ok-mg-statement \((stmt, r-cond-list, name-alist, proc-list)\) \\
\(\land\) ok-mg-def-plistp \((proc-list)\) \\
\(\land\) ok-mg-statep \((mg-state, r-cond-list)\) \\
\(\land\) (code \((\text{translate-def-body} (\text{assoc} (\text{subr}, proc-list), proc-list))\) \\
\(=\) append \((\text{code} \((\text{translate-def-body} (\text{assoc} (\text{subr}, proc-list), proc-list))\), \) \\
\text{code2})\) \\
\(\land\) user-defined-procp \((\text{subr}, proc-list)\) \\
\(\land\) listp \((\text{ctrl-stk})\) \\
\(\land\) all-cars-unique \((\text{mg-alist} \((mg-state)\))\) \\
\(\land\) signatures-match \((\text{mg-alist} \((mg-state)\), name-alist)\) \\
\(\land\) mg-vars-list-ok-in-p-state \((\text{mg-alist} \((mg-state)\), \) \\
\text{bindings} \((\text{top} (\text{ctrl-stk})), \) \\
\text{temp-stk})\) \\
\(\land\) no-p-aliasing \((\text{bindings} \((\text{top} (\text{ctrl-stk})), \) mg-alist \((mg-state)\))\) \\
\(\land\) normal \((mg-state)\) \\
\(\rightarrow\) (p-step (p-state \((\text{tag} \((\text{'pc}, \text{'mg-simple-constant-eq}, 4)\), \) \\
push \((\text{p-frame} \((\text{list} \((\text{cons} \((\text{'ans}, \) \\
value \((\text{car} \((\text{call-actuals} \((stmt)\))), \) \\
\text{bindings} \((\text{top} (\text{ctrl-stk}))))\), \) \\
\text{cons} \((\text{'x}, \) \\
value \((\text{cadr} \((\text{call-actuals} \((stmt)\))), \) \\
\text{bindings} \((\text{top} (\text{ctrl-stk}))))\), \) \\
\text{cons} \((\text{'y}, \) \\
mg-to-p-simple-literal \((\text{caddr} \((\text{call-actuals} \((stmt)\))))\), \) \\
\text{tag} \((\text{'pc}, \) \\
\text{cons} \((\text{subr}, \) \\
\text{length} \((\text{code} \((\text{cinfo})\)) \) \\
+ 4)\)), \) \\
\text{ctrl-stk})\), \\
push \((\text{tag} \((\text{'bool}, \text{eq-value}), \) \\
\text{map-down-values} \((\text{mg-alist} \((\text{mg-state})\), \) \\
\text{bindings} \((\text{top} (\text{ctrl-stk})), \) \\
\text{temp-stk}))\), \) 

63
Theorem: mg-simple-constant-eq-step-10

\( ((n \not\equiv 0) \land \neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))})) \land (\text{car (stmt)} = '\text{predefined-proc-call-mg}) \land (\text{call-name (stmt)} = '\text{mg-simple-constant-eq}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \land \text{ok-mg-def-plistp (proc-list}) \)
∧ ok-mg-statep (mg-state, r-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)
∧ listp (ctrl-stk)
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, (mg-simple-constant-eq . 5)),
     push (p-frame (list (cons ('ans,
                        value (car (call-actuals (stmt)),
                          bindings (top (ctrl-stk))))),
                          cons ('x,)
                          value (cadr (call-actuals (stmt)),
                          bindings (top (ctrl-stk))))),
                          cons ('y, mg-to-p-simple-literal (caddr (call-actuals (stmt))))),
     tag ('pc,
     cons (subr, length (code (cinfo))
     + 4)),
     ctrl-stk),
     push (value (car (call-actuals (stmt))),
     bindings (top (ctrl-stk))))),
     push (tag ('bool, eq-value),
     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, (mg-simple-constant-eq . 6)),
push (p-frame (list (cons ('ans,
     value (car (call-actuals (stmt)),
     bindings (top (ctrl-stk))))),
     cons ('x,
Theorem: mg-simple-constant-eq-step-11-true-case
\[ (n \not= 0) \land \neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list}, \text{list}(\text{length}(\text{temp-stk}), \text{p-ctrl-stk-size}(\text{ctrl-stk}))) \land (\text{car}(\text{stmt}) = \text{predefined-proc-call-mg}) \land (\text{call-name}(\text{stmt}) = \text{mg-simple-constant-eq}) \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{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})) = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \land \text{user-defined-procp}(\text{subr}, \text{proc-list}) \land \text{listp}(\text{ctrl-stk}) \land \text{all-cars-unique}(\text{mg-alist}(\text{mg-state})) \land \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \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}))
\begin{align*}
& \land \text{normal}(\text{mg-state}) \\
& \land \text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))) \\
& = \text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{call-actuals}(\text{stmt})))) \\
\rightarrow & (\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, (\text{mg-simple-constant-eq} . 6))), \\
& \quad \text{push}(\text{p-frame}(\text{list}(\begin{align*}
& \begin{cases}
& \text{'ans} , \\
& \text{value}(\text{car}(\text{call-actuals}(\text{stmt}))), \\
& \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \text{cons}(\text{'x}, \\
& \text{value}(\text{cadr}(\text{call-actuals}(\text{stmt}))), \\
& \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \text{cons}(\text{'y}, \\
& \text{mg-to-p-simple-literal}(\text{caddr}(\text{call-actuals}(\text{stmt})))), \\
& \text{tag}(\text{'pc}, \\
& \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo}))) \\
& + 4)), \\
& \text{ctrl-stk}), \\
& \text{rput}(\text{tag}(\text{'bool}, \text{'t}), \\
& \text{untag}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt}))), \\
& \text{bindings}(\text{top}(\text{ctrl-stk})))), \\
& \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \\
& \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \text{temp-stk})), \\
& \text{translate-proc-list}(\text{proc-list}), \\
& \text{list}(\text{list}(\begin{align*}
& \text{'c-c}, \\
& \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}), \text{t-cond-list}))), \\
& \text{MG-MAX-CTRL-STK-SIZE}, \\
& \text{MG-MAX-TEMP-STK-SIZE}, \\
& \text{MG-WORD-SIZE}, \\
& \text{'run}))) \\
& = \text{p-state}(\text{tag}(\text{'pc}, \\
& \text{cons}(\text{subr}, \\
& \begin{cases}
& \text{if}\ \text{normal}(\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})))) \\
& \text{then}\ \text{length}(\text{code}(\text{translate}(\text{cinfo}, \\
& \text{t-cond-list}, \text{stmt}, \text{proc-list})))) \\
& \text{else}\ \text{find-label}(\text{fetch-label}(\text{cc}(\text{mg-meaning-r}(\text{stmt}, \text{proc-list}))))
\end{cases}))
\end{align*}))
\end{align*}
mg-state,
n,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk))),
label-alist (translate (cinfo,
t-cond-list,
stmt,
proc-list)),
append (code (translate (cinfo,
t-cond-list,
stmt,
proc-list)),

code2)) endif),

ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (stmt,
proc-list,
mg-state,
n,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk)))),

bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list (c-c,
mg-cond-to-p-nat (cc (mg-meaning-r (stmt,
proc-list,
mg-state,
n,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk))))),

t-cond-list)),

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

THEOREM: mg-simple-constant-eq-step-11-false-case
((n ≠ 0)
∧ (¬ resources-inadequatep (stmt,
proc-list,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk))))
∧ (car (stmt) = 'predefined-proc-call-mg)
∧ (call-name (stmt) = 'mg-simple-constant-eq)

68
\[\begin{align*}
\wedge & \quad \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\wedge & \quad \text{ok-mg-def-plistp} (\text{proc-list}) \\
\wedge & \quad \text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list}) \\
\wedge & \quad (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list}))) \\
& \quad = \quad \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code}2)) \\
\wedge & \quad \text{user-defined-procp} (\text{subr}, \text{proc-list}) \\
\wedge & \quad \text{listp} (\text{ctrl-stk}) \\
\wedge & \quad \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \\
\wedge & \quad \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist}) \\
\wedge & \quad \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk}) \\
\wedge & \quad \text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state})) \\
\wedge & \quad \text{normal} (\text{mg-state}) \\
\wedge & \quad (\text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{cadr} (\text{call-actuals} (\text{stmt})), \text{mg-alist} (\text{mg-state})))))) \\
& \neq \quad (\text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{call-actuals} (\text{stmt})))))) \\
\rightarrow & \quad (\text{p-step} (\text{p-state} (\text{tag} (\text{pc}, \text{mg-simple-constant-eq} . 6))), \\
& \quad \text{push} (\text{p-frame} (\text{list} (\text{cons} (\text{ans}, \\
& \quad \text{value} (\text{call-actuals} (\text{stmt})), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})))), \\
& \quad \text{cons} (\text{x}, \\
& \quad \text{value} (\text{cadr} (\text{call-actuals} (\text{stmt}))), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})))), \\
& \quad \text{cons} (\text{y}, \\
& \quad \text{mg-to-p-simple-literal} (\text{caddr} (\text{call-actuals} (\text{stmt}))))), \\
& \quad \text{tag} (\text{pc}, \\
& \quad \text{cons} (\text{subr}, \text{length} (\text{code} (\text{cinfo}))) \\
& \quad + \text{4})), \text{ctrl-stk}), \\
& \quad \text{rput} (\text{tag} (\text{bool}, \text{f}), \\
& \quad \text{untag} (\text{value} (\text{call-actuals} (\text{stmt})), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})))), \\
& \quad \text{map-down-values} (\text{mg-alist} (\text{mg-state}), \\
& \quad \text{bindings} (\text{top} (\text{ctrl-stk})), \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}), \text{t-cond-list})), \text{mg-max-ctrl-stk-size}, \\
& \quad \text{mg-max-temp-stk-size}, \text{mg-word-size}, \text{run})) \end{align*}\]
\[ \text{p-state (tag ('pc, cons (\text{subr},}
\]
\[ \quad \text{if normal (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))}
\]
\[ \quad \text{then length (code (translate (cinfo, t-cond-list, stmt, proc-list)))}
\]
\[ \quad \text{else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list))},
\]
\[ \quad \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list))},
\]
\[ \quad \quad \text{code2)) endif}),}
\]
\[ \text{ctrl-stk, map-down-values (mg-alist (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), bindings (top (ctrl-stk)), temp-stk), translate-proc-list (proc-list), list (list ('c-c, mg-cond-to-p-nat (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))),}
\]
\[ 70 \]
THEOREM: mg-simple-constant-eq-exact-time-lemma

\((n \not\equiv 0)\) \\
\land (\neg \\text{resources-inadequate}(stmt, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))) ) \\
\land (\text{car (stmt) = 'predefined-proc-call-mg}) \\
\land (\text{call-name (stmt) = 'mg-simple-constant-eq}) \\
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
\land \text{ok-mg-def-plistp (proc-list)} \\
\land \text{ok-mg-statep (mg-state, r-cond-list)} \\
\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)) = append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}) \\
\land \text{user-defined-procp (subr, proc-list)} \\
\land \text{listp (ctrl-stk)} \\
\land \text{all-cars-unique (mg-alist (mg-state))} \\
\land \text{signatures-match (mg-alist (mg-state), name-alist)} \\
\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{normal (mg-state))} \\
\rightarrow (p (\text{map-down (mg-state, proc-list, ctrl-stk, temp-stk, tag ('pc, cons (subr, length (code (cinfo))))), t-cond-list)}, \\
\text{clock (stmt, proc-list, mg-state, n)}) \\
= \text{p-state (tag ('pc, cons (subr, } \\
\begin{align*}
\text{if normal (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}\)
\end{align*}
then length (code (translate (cinfo, t-cond-list, stmt, proc-list)))

else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list))), append (code (translate (cinfo, t-cond-list, stmt, proc-list))), code2) endif)

ctrl-stk, map-down-values (mg-alist (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), bindings (top (ctrl-stk)), temp-stk), translate-proc-list (proc-list), list (list ('c-c, mg-cond-to-p-nat (cc (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), t-cond-list))}, MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, 'run))

EVENT: Make the library "c-predefined1".
Index

add-code, 2
alist-element-type-conversion, 3
all-cars-unique, 3, 4, 10, 11, 13, 15,
18, 24, 25, 28, 30, 31, 33,
34, 36, 38, 40, 42, 44, 46,
48, 51, 53, 54, 56, 57, 59,
61, 63, 65, 66, 69, 71
array-identifier-nat-p-objectp, 4

bindings, 6–72
bool-literal-bool-objectp, 5
bool-literal-bool-objectp2, 5
boolean-identifierp, 3, 29, 52
boolean-identifiers-have-boolean
literal-values, 3
boolean-literalp, 3–5
boolean-literalp-mapping, 4
boolean-mg-to-p-simple-literal, 2
boolean-mg-to-p-simple-literal2, 2
call-actuals, 5–16, 19–26, 29–49, 52–
67, 69
call-name, 5–8, 10, 11, 13, 15, 17,
19–23, 25, 27, 29–31, 33,
34, 36, 38, 40, 42, 44, 46,
48, 50, 52–55, 57, 59, 61,
63, 64, 66, 68, 71
cc, 7–29, 31–45, 47–52, 54–62, 64–
70, 72
clock, 18, 28, 51, 71
code, 6–72
cond-subsetp, 17, 27
definedp, 4, 5, 30, 53
fetch-label, 16, 19, 26, 28, 47, 50,
51, 68, 70, 72
find-label, 17, 19, 27, 29, 47, 50, 52,
68, 70, 72
fix-small-integer, 5
fix-small-integer-identity, 5
ilessp, 2
int-identifierp, 3
int-identifiers-have-int-literal
l-values, 3
int-literal-int-objectp, 5
int-literalp, 2–5
int-literalp-mapping, 4
int-literalp-value-small, 5
int-literals-mapping, 3
integerp, 5

label-alist, 2, 16, 19, 26, 28, 47, 50,
51, 68, 70, 72
length, 3, 4, 6–72
length-plistp, 3
length-plistp-2-rewrite, 3
lessp-add1-add1-2, 2
lessp-add1-add1-add1-3, 2

literal-type-conversion, 3
map-down, 6, 18, 20, 28, 30, 51, 53,
71
map-down-values, 6–17, 19–27, 29,
31–45, 47–50, 52, 54–60, 62–
70, 72
mg-alist, 4–72
mg-alistp, 2, 3
mg-bool, 46
mg-bool-ok-boolean, 46
mg-cond-to-p-nat, 5, 7–17, 19–27,
29, 31–45, 47–50, 52, 54–62, 64–
69, 71, 72
mg-max-ctrl-stk-size, 7–17, 19–27, 29,
31–45, 47–50, 52, 54–62, 64–
69, 71, 72
mg-max-temp-stk-size, 3, 4, 7–17,
19–27, 29, 31–45, 47–50, 52,
54–69, 71, 72
mg-meaning-predefined-proc-call, 1
mg-meaning-r, 1, 16–19, 26–29, 47–
52, 67, 68, 70–72
33, 34, 36, 38, 40, 42, 44, 46, 48, 51, 53, 54, 56, 57, 59, 61, 63, 64, 66, 69, 71
ok-mg-statement, 5–8, 10, 11, 13, 15, 17, 19–23, 25, 27, 29–31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52–54, 56, 57, 59, 61, 63, 64, 66, 69, 71
ok-mg-statep, 5–8, 10, 11, 13, 15, 17, 19–23, 25, 27, 29–31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52–54, 56, 57, 59, 61, 63, 64, 66, 69, 71
ok-mg-valuep, 46
ok-translation-parameters, 17, 27

p, 18, 28, 51, 71
p-ctrl-stk-size, 6–8, 10, 11, 13, 15–23, 25–31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 51–54, 56, 57, 59, 61, 63, 64, 66–68, 70–72
p-frame, 8–12, 14, 16, 21–24, 26, 32–35, 37, 39, 41–43, 45, 46, 49, 55–58, 60–67, 69
p-objectp-type, 4, 5
p-state, 7–17, 19–27, 29, 31–45, 47–50, 52, 54–69, 71, 72
p-step, 6, 7, 9, 10, 12, 14, 16, 20, 21, 23, 24, 26, 30, 32, 33, 35, 37, 39, 41, 43, 45, 47, 49, 53, 55, 56, 58, 60, 62, 64, 65, 67, 69
p-word-size, 4, 5
 plistp, 18, 28
predefined-call-translation-2, 1
predefined-call-meaning-r-2, 1
predefined-proc-call-sequence, 2
push, 6–14, 16, 20–26, 31–46, 49, 54–67, 69
resource-errorp, 18, 28
resources-inadequatep, 1, 6–8, 10, 11, 13, 15, 17, 20–23, 25, 27, 30, 31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 50, 53–55, 57, 59, 61, 63, 64, 66, 68, 71
rget, 11–14, 16, 36–41, 58, 60, 62
rput, 15, 16, 25, 26, 45, 47, 49, 66, 67, 69
signal-system-error, 1
signatures-match, 5, 6, 10, 12, 13, 15, 18, 19, 24, 25, 28–31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 51–54, 56, 57, 59, 61, 63, 65, 66, 69, 71
simple-identifier-mapping, 3
simple-identifier-mapping-2, 4
simple-identifier-mapping-3, 4
simple-identifier-natp-objectp, 4
simple-identifierp, 2–4, 6, 19, 29, 30, 52
simple-identifiers-have-simple-values, 2
simple-typed-literalp, 2, 3, 53
simple-typed-literalp-mapping-li stp, 2
small-integerp, 3, 5
small-integerp-mapping, 5
small-naturalp, 3, 4
special-conditions-mg-cond-to-p -nat, 5
tag, 2, 5–39, 41–47, 49–72
top, 6–72
translate, 1, 6–8, 10, 11, 13, 15–22, 24–31, 33, 34, 36, 38, 40, 42, 44, 46–54, 56, 57, 59, 61, 63, 65–72
translate-def-body, 6–8, 10, 11, 13, 15, 17, 20–22, 24, 25, 27, 30, 31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 51, 53, 54,
56, 57, 59, 61, 63, 65, 66, 69, 71
translate-proc-list, 6–17, 19–27, 29, 31–45, 47–50, 52, 54–62, 64–70, 72
type, 3, 4

unlabel, 2
unlabel-call, 2
untag, 2–5, 11–14, 16, 25, 26, 36–42, 45–49, 58, 60, 62, 66, 67, 69
untag-boolean, 4
untag-int-literal-integerp, 5
user-defined-procp, 6–8, 10, 11, 13, 15, 18, 20–22, 24, 25, 27, 30, 31, 33, 34, 36, 38, 40, 42, 44, 46, 48, 51, 53, 54, 56, 57, 59, 61, 63, 65, 66, 69, 71

value, 3, 4, 6–16, 20–26, 31–47, 49, 54–67, 69