EVENT: Start with the library "c-predefined3".

**THEOREM**: arrays-have-non-zerop-lengths  
\[
\text{array-identifierp}(x, \text{alist}) \land \text{mg-alistp}(\text{alist}) \rightarrow (\text{array-length}(\text{cdr}(\text{assoc}(x, \text{alist})))) \neq 0
\]

EVENT: Enable map-down-values-preserves-length.

**THEOREM**: lessp-plus-transitive  
\[
(((x + (w - 1)) < z) \land (y < w)) \rightarrow ((x + y) < z) = \text{t}
\]

EVENT: Disable lessp-plus-transitive.

**THEOREM**: idifference-lessp2  
\[
((x \in \mathbb{N}) \land (x < y)) \rightarrow ((\text{idifference}(y, x) = 0) = \text{f}) \land (\text{idifference}(y, x) \in \mathbb{N})
\]

**THEOREM**: zerop-integerp-trichotomy  
\[
(\text{integerp}(x) \land (x \notin \mathbb{N})) \rightarrow \text{negativep}(x)
\]

EVENT: Disable zerop-integerp-trichotomy.

**THEOREM**: p-object-type-int  
\[
(\text{p-objectp-type}(\text{'int}, \text{tag}(\text{'int}, x), \text{state}) = \text{small-integerp}(x, \text{p-word-size}(\text{state}))) \land (\text{p-objectp-type}(\text{'int}, \text{list}(\text{'int}, x), \text{state}) = \text{small-integerp}(x, \text{p-word-size}(\text{state})))
\]

**THEOREM**: simple-identifierp-options  
\[
(\text{int-identifierp}(x, \text{alist}) \lor \text{boolean-identifierp}(x, \text{alist}) \lor \text{character-identifierp}(x, \text{alist}) \rightarrow \text{simple-identifierp}(x, \text{alist})
\]

**THEOREM**: small-integerp-difference  
\[
(\text{small-integerp}(x, n)
\]

1
\begin{align*}
\land \text{ small-integerp } (y, n) \\
\land (x \not\equiv 0) \\
\land (\neg \text{ negativep } (y))) \\
\rightarrow \text{ small-integerp } (\text{idifference } (x, y), n) \\
\end{align*}

\textbf{Theorem: limits-for-small-integerp} \\
((x < \text{MAXINT}) \land (x \not\equiv 0)) \rightarrow \text{small-integerp } (x, 32)

\textbf{Theorem: simple-typed-identifier-simple-identifierp} \\
\text{simple-typed-identifierp } (x, \text{type}, \text{alist}) \rightarrow \text{simple-identifierp } (x, \text{alist})

\textbf{Theorem: mg-var-ok-array-index-ok3} \\
(\text{mg-vars-list-ok-in-p-state } (\text{mg-vars}, \text{bindings}, \text{temp-stk}) \\
\land \text{mg-alistp } (\text{mg-vars}) \\
\land \text{array-identifierp } (x, \text{mg-vars})) \\
\rightarrow ((\text{untag } (\text{value } (x, \text{bindings}))) \\
+ (\text{array-length } (\text{cadr } (\text{assoc } (x, \text{mg-vars}))) - 1)) \\
< \text{length } (\text{temp-stk}) \\
= \text{t})

\textbf{Theorem: idifference-lessp} \\
((\neg \text{negativep } (y)) \land (\text{idifference } (x, y) \not\equiv 0)) \rightarrow ((y < x) = \text{t})

\textbf{Theorem: nat-p-objectp-reduction} \\
\text{p-objectp-type } (\text{'nat}, \text{tag } (\text{'nat}, x), \text{state}) \\
= \text{small-naturalp } (x, \text{p-word-size } (\text{state}))

\textbf{Theorem: array-index-small-naturalp} \\
((\text{temp-stk-size} < \text{MG-MAX-TEMP-STK-SIZE}) \\
\land ((a + (\text{array-size} - 1)) < \text{temp-stk-size}) \\
\land (\text{index} < \text{array-size})) \\
\rightarrow \text{small-naturalp } (a + \text{index}, 32)

\textbf{Theorem: mg-index-array-args-have-simple-mg-type-refps} \\
((\text{car } (\text{stmt}) = \text{'predefined-proc-call-mg}) \\
\land (\text{call-name } (\text{stmt}) = \text{'mg-index-array}) \\
\land \text{ok-mg-statement } (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\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})) \\
\rightarrow (\text{simple-typed-identifierp } (\text{car } (\text{call-actuals } (\text{stmt}))), \\
\text{array-elmtypen } \text{cadr } (\text{assoc } \text{cadr } (\text{call-actuals } (\text{stmt})), \\
\text{mg-alist } (\text{mg-state}))), \\
\text{mg-alist } (\text{mg-state})) \\
\land \text{array-identifierp } (\text{cadr } (\text{call-actuals } (\text{stmt})), \text{mg-alist } (\text{mg-state})) \\
\land \text{int-identifierp } (\text{caddr } (\text{call-actuals } (\text{stmt})), \text{mg-alist } (\text{mg-state}))
\[\begin{align*}
\land (\text{cadddr (call-actuals (stmt)}) = \text{array-length (cadr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state)))))}
\end{align*}\]

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

**Theorem:** mg-index-array-arg4-small-integerp
\[\begin{align*}
((\text{car (stmt)} = \text{'predefined-proc-call-mg}) \land (\text{call-name (stmt)} = \text{'mg-index-array}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \land \text{ok-mg-statep (mg-state, r-cond-list)} \land \text{signatures-match (mg-alist (mg-state), name-alist)}) \\
\rightarrow \text{small-integerp (cadddr (call-actuals (stmt)), 32)}
\end{align*}\]

**Theorem:** not-zerop-mg-index-array-arg4
\[\begin{align*}
((\text{car (stmt)} = \text{'predefined-proc-call-mg}) \land (\text{call-name (stmt)} = \text{'mg-index-array}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \land \text{ok-mg-statep (mg-state, r-cond-list)} \land \text{signatures-match (mg-alist (mg-state), name-alist)}) \\
\rightarrow ((\text{cadddr (call-actuals (stmt))} \in \mathbb{N}) \land (\text{cadddr (call-actuals (stmt))} \neq 0))
\end{align*}\]

**Theorem:** mg-index-array-steps-1-4
\[\begin{align*}
((n \neq 0) \land (\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))}) \\
\land (\text{car (stmt)} = \text{'predefined-proc-call-mg}) \land (\text{call-name (stmt)} = \text{'mg-index-array}) \land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \land \text{ok-mg-def-plistp (proc-list)} \land \text{ok-mg-statep (mg-state, r-cond-list)} \land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}) = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2})}
\end{align*}\]
\(\wedge\) user-defined-procp \((\text{subr}, \text{proc-list})\)
\(\wedge\) listp \((\text{ctrl-stk})\)
\(\wedge\) all-cars-unique \((\text{mg-alist} (\text{mg-state}))\)
\(\wedge\) signatures-match \((\text{mg-alist} (\text{mg-state}), \text{name-alist})\)
\(\wedge\) mg-vars-list-ok-in-p-state \((\text{mg-alist} (\text{mg-state}),\)
\(\wedge\) bindings \((\text{top} (\text{ctrl-stk})), \text{temp-stk})\)
\(\wedge\) no-p-aliasing \((\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state}))\)
\(\wedge\) normal \((\text{mg-state})\)
\(\rightarrow\) (p-step (p-step (p-step (p-step (map-down (mg-state,\)
\(\wedge\) proc-list, \text{ctrl-stk}, \text{temp-stk},\)
\(\wedge\) tag ('pc,\)
\(\wedge\) cons (subr, \text{length (code (cinfo))}),\)
\(\wedge\) t-cond-list)))\)
\(=\) p-state \((\text{tag} ('\text{pc}, \text{cons (subr, length (code (cinfo)) + 4)}),\)
\(\wedge\) ctrl-stk,\)
\(\wedge\) push \((\text{tag} ('\text{int}, \text{caddr (call-actuals (stmt)})))\),\)
\(\wedge\) push \((\text{value (caddr (call-actuals (stmt)})),\)
\(\wedge\) bindings \((\text{top} (\text{ctrl-stk}))\),\)
\(\wedge\) push \((\text{value (cadr (call-actuals (stmt)})),\)
\(\wedge\) bindings \((\text{top} (\text{ctrl-stk}))\),\)
\(\wedge\) push \((\text{value (car (call-actuals (stmt)})),\)
\(\wedge\) bindings \((\text{top} (\text{ctrl-stk}))\),\)
\(\wedge\) map-down-values \((\text{mg-alist} (\text{mg-state}),\)
\(\wedge\) bindings \((\text{top} (\text{ctrl-stk})), \text{temp-stk})))))\),\)
\(\wedge\) translate-proc-list \((\text{proc-list})\),\)
\(\wedge\) list \((\text{list} ('\text{c-c},\)
\(\wedge\) mg-cond-to-p-nat (cc (mg-state), t-cond-list)))\),\)
\(\wedge\) MG-MAX-CTRL-STK-SIZE,\)
\(\wedge\) MG-MAX-TEMP-STK-SIZE,\)
\(\wedge\) MG-WORD-SIZE,\)
\(\wedge\) 'run))

**Theorem:** mg-index-array-step-5
\((n \neq 0)\)
\(\wedge\) (\(\neg\) resources-inadequatep \((\text{stmt},\)
\(\wedge\) proc-list,\)
\(\wedge\) list \((\text{length (temp-stk)},\)
\(\wedge\) p-ctrl-stk-size \((\text{ctrl-stk})))\)))\)
\(\wedge\) (car \((\text{stmt}) = '\text{predefined-proc-call-mg})
∧ (call-name (stmt) = 'mg-index-array)
∧ 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-proc (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, cons (subr, length (code (cinfo)) + 4)),
   ctrl-stk,
   push (tag ('int, caddr (call-actuals (stmt))),
   push (value (caddr (call-actuals (stmt))),
           bindings (top (ctrl-stk))),
   push (value (cadr (call-actuals (stmt))),
           bindings (top (ctrl-stk))),
   push (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, '(mg-index-array , 0)),
   push (p-frame (cons (cons ('ans,
                             value (car (call-actuals (stmt))),
                             bindings (top (ctrl-stk)))),
                 cons (cons ('a,
                             value (cadr (call-actuals (stmt))),
                             bindings (top (ctrl-stk)))),
                 cons (cons ('i,
                             value (caddr (call-actuals (stmt))))),
                 translate-def-body (assoc (subr, proc-list), proc-list),
                 push (value (caddr (call-actuals (stmt))),
                           bindings (top (ctrl-stk)))),
                 push (value (car (call-actuals (stmt))),
                           bindings (top (ctrl-stk)))),
                 push (value (cadr (call-actuals (stmt))),
                           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-index-array , 0)),
   push (p-frame (cons (cons ('ans,
                             value (car (call-actuals (stmt))),
                             bindings (top (ctrl-stk)))),
                 cons (cons ('a,
                             value (cadr (call-actuals (stmt))),
                             bindings (top (ctrl-stk)))),
                 cons (cons ('i,
                             value (caddr (call-actuals (stmt))))),
                 translate-def-body (assoc (subr, proc-list), proc-list),
                 push (value (caddr (call-actuals (stmt))),
                           bindings (top (ctrl-stk)))),
                 push (value (car (call-actuals (stmt))),
                           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-index-array-steps-6-8
\(((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-index-array} \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-proc}(\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-step}(\text{p-step}(\text{p-state}(\text{tag}(\text{’pc, ’(mg-index-array . 0)))))))),
push (p-frame (cons (cons 'ans,
    value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
    cons (cons 'a,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
    cons (cons 'i,
    value (caddr (call-actuals (stmt))),
    bindings (top (ctrl-stk))))),
    cons (cons ('array-size,
    tag ('int,
    caddr (call-actuals (stmt))))
    '((temp-i
    nat 0))))),
  tag ('pc,
    cons (subr,
    length (code (cinfo))
    + 5))),
  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-index-array . 3)),
push (p-frame (list (cons 'ans,
    value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
    cons ('a,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
    cons ('i,
    value (caddr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
    cons ('array-size,
    tag ('int,
    caddr (call-actuals (stmt))))),

7
THEOREM: mg-index-array-steps-9-12-neg-index
\((n \not\equiv 0)\)
\(\land \) (\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-index-array)
\(\land \) 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 (translate-def-body (assoc (subr, proc-list), proc-list))
\(= \) append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))
\(\land \) user-defined-procp (subr, proc-list)
\(\land \) listp (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)
\(\land \) negativep (untag (mg-to-p-simple.literal (caddr (assoc (caddr (call-actuals (stmt)))))),
\(\text{cons}\ (\text{'temp-i,}
\text{mg-to-p-simple.literal (caddr (assoc (caddr (call-actuals (stmt))),}
\text{mg-alist (mg-state))))),
\text{tag}\ (\text{'pc,}
\text{cons}\ (\text{subr, length (code (cinfo)) + 5)}),
\text{ctrl-stk),}
push (mg-to-p-simple.literal (caddr (assoc (caddr (call-actuals (stmt))),
\text{mg-alist (mg-state))))),
\text{map-down-values (mg-alist (mg-state),}
\text{bindings (top (ctrl-stk)),}
temp-stk)),
\text{translate-proc-list (proc-list),}
\text{list (list (\text{'c-c,}
\text{mg-\text{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))}
\[ \text{p-step (p-step (p-step (p-step (p-state (tag (\text{'pc, }
\text{mg-index-array . 3))},}
\text{push (p-frame (list (cons (\text{'ans,}
\text{value (car (call-actuals (stmt)),}
\text{bindings (top (ctrl-stk))},)
\text{cons (\text{'a,}
\text{value (cadr (call-actuals (stmt)),}
\text{bindings (top (ctrl-stk))},)
\text{cons (\text{'i,}
\text{value (caddr (call-actuals (stmt)),}
\text{bindings (top (ctrl-stk))},)
\text{cons (\text{'array-size,}
\text{tag (\text{'int,}
\text{cadddr (call-actuals (stmt)))},}
\text{cons (\text{'temp-i,}
\text{mg-to-p-simple-literal (caddr (assoc (caddr (mg-alist (mg-state)))))}}
\text{tag (\text{'pc,}
\text{cons (\text{'subr,}
\text{length (code (cinfo)) + 5))},
\text{ctrl-stk)},}
\text{push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))
\text{mg-alist (mg-state))))},
\text{map-down-values (mg-alist (mg-state)),}
\text{bindings (top (ctrl-stk)),}
\text{temp-stk)),}
\text{translate-proc-list (proc-list),}
\text{list (list (\text{'c-c,}
\text{mg-cond-to-p-nat (cc (mg-state),}
\text{t-cond-list))},
\text{MG-MAX-CTRL-STK-SIZE,}
\text{MG-MAX-TEMP-STK-SIZE,}
\text{MG-WORD-SIZE,}
\text{'run))))))}
\]
THEOREM: mg-index-array-steps-9-11-no-error

\((n \neq 0) \land (\neg \text{resources-inadequate}\ (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk}))))\)
\(\land (\text{car} (\text{stmt}) = \text{predefined-proc-call-mg})\)
\(\land (\text{call-name} (\text{stmt}) = \text{mg-index-array})\)
\(\land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})\)
\(\land \text{ok-mg-def-plistp} (\text{proc-list})\)
\(\land \text{ok-mg-statem} (\text{mg-state}, \text{r-cond-list})\)
\(\land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list})))\)
\(\quad = \text{append} (\text{code} (\text{translate} (\text{cinfo, t-cond-list, stmt, proc-list})), \text{code2}))\)
\(\land \text{user-defined-procp} (\text{subr}, \text{proc-list})\)
\(\land \text{listp} (\text{ctrl-stk})\)
\(\land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state}))\)
\(\land \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist})\)
\(\land \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk})\)
\(\land \text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state}))\)
\(\land \text{normal} (\text{mg-state})\)
\(\land (\neg \text{negativep} (\text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})), \text{mg-alist} (\text{mg-state})))))))\))
\(\rightarrow (\text{p-step} (\text{p-step} (\text{p-step} (\text{p-state} (\text{tag} (\text{pc, mg-index-array . 3}))))))\)
mg-alist (mg-state

(tag (\'pc,
    cons (subr,
        length (code (cinfo))
        + 5)),
    ctrl-stk),
push (mg-to-p-simple-literal (caddr (caddr (call-actuals (stmt))),
    mg-alist (mg-state)),
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-index-array . 6)),
push (p-frame (list (cons ('ans,
                value (car (call-actuals (stmt)),
                        bindings (top (ctrl-stk))))),
            cons ('a,
                value (cadr (call-actuals (stmt)),
                        bindings (top (ctrl-stk))))),
            cons ('i,
                value (caddr (call-actuals (stmt)),
                        bindings (top (ctrl-stk))))),
            cons ('array-size,
                tag ('\int,  
                caddr (call-actuals (stmt))))),
            cons ('temp-i,
                mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                mg-alist (mg-state)))))),
    tag (\'pc,
        cons (subr, length (code (cinfo))
        + 5)),
    ctrl-stk),
push (mg-to-p-simple-literal (caddr (caddr (call-actuals (stmt))),
    mg-alist (mg-state))),
push (tag ('\int, caddr (call-actuals (stmt))),
    map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)))),
\[
\text{temp-stk \underline{,} \text{translate-proc-list (proc-list),} \\
\text{list (list ('c-c,} \\
\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))}, \\
\text{MG-MAX-CTRL-STK-SIZE,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{MG-WORD-SIZE,} \\
\text{'run))}
\]

;; This is the (sub-int) instruction
;; The proof of this could be cleaned up substantially

**THEOREM:** mg-index-array-step-12-no-error
\[(n \not\equiv 0) \land \neg \text{resources-inadequate (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))} \land (\text{car (stmt) = 'predefined-proc-call-mg) \land ok-mg-statement (stmt, r-cond-list, name-alist, proc-list) \land ok-mg-def-plistp (proc-list) \land ok-mg-statep (mg-state, r-cond-list) \land (code (translate-def-body (assoc (subr, proc-list), proc-list)) = append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))} \land \text{user-defined-procp (subr, proc-list)} \land \text{listp (ctrl-stk)} \land \text{all-cars-unique (mg-alist (mg-state)) \land signatures-match (mg-alist (mg-state), name-alist) \land mg-vars-list-ok-in-p-state (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)} \land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state)) \land normal (mg-state) \land (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt), mg-alist (mg-state)))))))))} \rightarrow (p-step (p-state (tag ('pc, 'mg-index-array . 6)), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('a, value (cadr (call-actuals (stmt))))),} \]

12
bindings (top (ctrl-stk))),
  cons ('i,
    value (caddr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons ('array-size,
    tag ('int,
      caddr (call-actuals (stmt)))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
      mg-alist (mg-state))))),
  tag ('pc,
    cons (subr, length (code (cinfo))
      + 5)),
  ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
    mg-alist (mg-state))))),
push (tag ('int, caddr (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))
= p-state (tag ('pc, '(mg-index-array . 7)),
push (p-frame (list (cons ('ans,
    value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk))))),
  cons ('a,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons ('i,
    value (caddr (call-actuals (stmt))),
    bindings (top (ctrl-stk)))),
  cons ('array-size,
    tag ('int,
      caddr (call-actuals (stmt)))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
      mg-alist (mg-state))))),
  tag ('pc,
Theorem: mg-index-array-step-13-index-error

\[ ((n \not\equiv 0) \land (\neg \text{resources-inadequatep } (\text{stmt},\ \text{proc-list},\ \text{list } (\text{length } (\text{temp-stk}),\ p-\text{ctrl-stk-size } (\text{ctrl-stk})))) \land (\text{car } (\text{stmt}) = \text{'predefined-proc-call-mg}) \land (\text{call-name } (\text{stmt}) = \text{'mg-index-array}) \land \text{ok-mg-statement } (\text{stmt},\ r-\text{cond-list},\ \text{name-alist},\ \text{proc-list}) \land \text{ok-mg-def-plistp } (\text{proc-list}) \land \text{ok-mg-statep } (\text{mg-state},\ r-\text{cond-list}) \land (\text{code } (\text{translate-def-body } (\text{assoc } (\text{subr},\ \text{proc-list}),\ \text{proc-list}))) \land \text{expr-list2}) \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-state},\ \text{proc-list}) \land \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}) \land (\neg \text{negativep } (\text{untag } (\text{mg-to-p-simple-literal } (\text{caddr } (\text{assoc } (\text{caddr } (\text{call-actuals } (\text{stmt})),\ \text{mg-alist } (\text{mg-state})))))))) \land (\text{idifference } (\text{cadddr } (\text{call-actuals } (\text{stmt})))],
\[
\text{ntag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\text{mg-alist} (\text{mg-state})))))) \simeq 0)
\]
\[
\rightarrow (\text{p-step} (\text{p-state} (\text{tag} ('\text{pc}', '\text{mg-index-array} . 7))),
\text{push} (\text{p-frame} (\text{list} (\text{cons} ('\text{ans},
\text{value} (\text{car} (\text{call-actuals} (\text{stmt})),
\text{bindings} (\text{top} (\text{ctrl-stk})))))),
\text{cons} ('\text{a},
\text{value} (\text{cadr} (\text{call-actuals} (\text{stmt})),
\text{bindings} (\text{top} (\text{ctrl-stk})))),
\text{cons} ('\text{i},
\text{value} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\text{bindings} (\text{top} (\text{ctrl-stk})))),
\text{cons} ('\text{array-size},
\text{tag} ('\text{int},
\text{cadddr} (\text{call-actuals} (\text{stmt})))),
\text{cons} ('\text{temp-i},
\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\text{mg-alist} (\text{mg-state}))))),
\text{tag} ('\text{pc},
\text{cons} (\text{subr},
\text{length} (\text{code} (\text{cinfo}))
+ 5))),
\text{ctrl-stk}),
\text{push} (\text{tag} ('\text{int},
\text{idifference} (\text{cadddr} (\text{call-actuals} (\text{stmt}))),
\text{ntag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\text{mg-alist} (\text{mg-state}))))),
\text{map-down-values} (\text{mg-alist} (\text{mg-state}),
\text{bindings} (\text{top} (\text{ctrl-stk})),
\text{temp-stk})),
\text{translate-proc-list} (\text{proc-list}),
\text{list} (\text{list} ('\text{c-c},
\text{mg-cond-to-p-nat} (\text{cc} (\text{mg-state}), \text{t-cond-list}))),
\text{MG-MAX-CTRL-STK-SIZE},
\text{MG-MAX-TEMP-STK-SIZE},
\text{MG-WORD-SIZE},
'\text{run})))
\]
value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons ('array-size,
  tag ('int,
    cadddr (call-actuals (stmt)))))
cons ('temp-i,
  mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
    mg-alist (mg-state))))),
tag ('pc,
  cons (subr, length (code (cinfo))
    + 5)),
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))

\textbf{Theorem:} mg-index-array-steps-14-16-index-error
\((n \not\equiv 0)\)
\land (\neg \text{resources-inadequatep} (stmt,
  proc-list,
  list (length (temp-stk),
    p-ctrl-stk-size (ctrl-stk))))
\land (\text{car} (stmt) = \text{'predicate-proc-call-mg})
\land (\text{call-name} (stmt) = \text{'mg-index-array})
\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} (\text{translate-def-body} (assoc (subr, proc-list), proc-list))
\quad = \quad \text{append} (\text{code} (\text{translate} (cinfo, t-cond-list, stmt, proc-list)),
\quad \text{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))
\text{16}
∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
∧ normal (mg-state)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))))
∧ (idifference (caddr (call-actuals (stmt)),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))))) ≃ 0))
→ (p-step (p-step (p-step (p-state (tag ('pc, 'mg-index-array . 16)),
push (p-frame (list (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk))))),
cons ('a,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))))),
cons ('i,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk))))),
cons ('array-size,
tag ('int,
caddr (call-actuals (stmt))))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))
tag ('pc,
cons (subr,
length (code (cinfo)) + 5))),
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, length (code (cinfo)) + 5)),
ctrl-stk,
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk),
Theorem: mg-index-array-step-13-no-error

\[ ((n \not\equiv 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-index-array}) \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}) \land (\neg \text{negativep}(\text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{caddr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))))) \land (\text{idifference}(\text{caddr}(\text{call-actuals}(\text{stmt}))), \text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{caddr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})))))) \not\equiv 0) \rightarrow (\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, \text{'mg-index-array} . 7))), \text{push}(\text{p-frame}(\text{list}(\text{cons}(\text{'ans}, \text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk}))))), \text{cons}(\text{'a}, \text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk}))))), \text{cons}(\text{'i}, \text{value}(\text{caddr}(\text{call-actuals}(\text{stmt})))),
bindings (top (ctrl-stk))),
cons ('array-size,
tag ('int,
cadddr (call-actuals (stmt))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))),
tag ('pc,
cons (subr, length (code (cinfo))
  + 5)),
ctrl-stk),
push (tag ('int,
idifference (cadddr (call-actuals (stmt))),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))))),
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-index-array . 8)),
push (p-frame (list (cons ('ans,
  value (car (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons ('a,
  value (cadr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons ('i,
  value (caddr (call-actuals (stmt)),
  bindings (top (ctrl-stk))))),
cons ('array-size,
tag ('int,
cadddr (call-actuals (stmt)))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))),
tag ('pc,
cons (subr, length (code (cinfo))
  + 5)),
19
Theorem: \( mg\text{-index-array-steps-14-15-no-error} \)

\[ ((n \neq 0) \land \neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list}, \text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))})) \]
\[ \land \ (\text{car (stmt)} = \text{’predefined-proc-call-mg}) \]
\[ \land \ (\text{call-name (stmt)} = \text{’mg-index-array}) \]
\[ \land \ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \]
\[ \land \ \text{ok-mg-def-plistp (proc-list}) \]
\[ \land \ \text{ok-mg-statep (mg-state, r-cond-list}) \]
\[ \land \ (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}) \]
\[ = \ \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))} \]
\[ \land \ \text{user-defined-proc}\text{p (subr, proc-list)} \]
\[ \land \ \text{listp (ctrl-stk}) \]
\[ \land \ \text{all-cars-unique (mg-alist (mg-state))} \]
\[ \land \ \text{signatures-match (mg-alist (mg-state), name-alist}) \]
\[ \land \ \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state), temp-stk), bindings (top (ctrl-stk))}) \]
\[ \land \ \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \]
\[ \land \ \text{normal (mg-state}) \]
\[ \land \ (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))))}) \]
\[ \land \ (\text{idifference (caddr (call-actuals (stmt)), untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))))} \neq 0) \]
\[ \rightarrow \ (p-step (p-step (p-state (tag (’pc, ’(mg-index-array . 8)), push (p-frame (list (cons (’\text{ans, value (car (call-actuals (stmt))}, bindings (top (ctrl-stk))))),} \]
cons ('a,
  value (cadr (call-actuals (stmt))),
  bindings (top (ctrl-stk)))),
cons ('i,
  value (caddr (call-actuals (stmt))),
  bindings (top (ctrl-stk)))),
cons ('array-size,
  tag ('int,
    cadddr (call-actuals (stmt)))),
cons ('temp-i,
  mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))),
    mg-alist (mg-state)))))

  tag ('pc,
    cons (subr,
      length (code (cinfo))
      +  5))),
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-index-array . 10)),
push (p-frame (list (cons ('ans,
  value (car (call-actuals (stmt))),
    bindings (top (ctrl-stk))))),
  cons ('a,
    value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk))))),
  cons ('i,
    value (caddr (call-actuals (stmt))),
    bindings (top (ctrl-stk))))),
  cons ('array-size,
    tag ('int,
      cadddr (call-actuals (stmt)))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))),
      mg-alist (mg-state)))))

21
tag ('pc,
  cons (subr, length (code (cinfo))
        + 5)),

  ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                              mg-alist (mg-state)))),
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-index-array-step-16-no-error
  ((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-index-array)
   ∧ 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)
   ∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                          mg-alist (mg-state)))))))

22
\[ \begin{align*}
&\quad \text{idifference (caddr (call-actuals (stmt)),}
\quad \text{untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),}
\quad \text{mg-alist (mg-state))))) \neq 0)}
\rightarrow \quad \text{(p-step (p-state (tag ('pc, 'mg-index-array . 10)),}
\quad \text{push (p-frame (list (cons ('ans,
\quad \text{value (car (call-actuals (stmt)),}
\quad \text{bindings (top (ctrl-stk)))},
\quad \text{cons ('a,
\quad \text{value (cadr (call-actuals (stmt)),}
\quad \text{bindings (top (ctrl-stk)))},
\quad \text{cons ('i,
\quad \text{value (caddr (call-actuals (stmt)),}
\quad \text{bindings (top (ctrl-stk)))},
\quad \text{cons ('array-size,
\quad \text{tag ('int,}
\quad \text{caddr (call-actuals (stmt)))},
\quad \text{cons ('temp-i,
\quad \text{mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),}
\quad \text{mg-alist (mg-state)))))},
\quad \text{tag ('pc,
\quad \text{cons (subr, length (code (cinfo))}
\quad + 5)})),
\quad \text{ctrl-stk),}
\quad \text{push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),}
\quad \text{mg-alist (mg-state)))))},
\quad \text{push (value (cadr (call-actuals (stmt))),}
\quad \text{bindings (top (ctrl-stk)))},
\quad \text{map-down-values (mg-alist (mg-state),}
\quad \text{bindings (top (ctrl-stk)),}
\quad \text{temp-stk))},
\quad \text{translate-proc-list (proc-list),}
\quad \text{list (‘c-c,}
\quad \text{mg-cond-to-p-nat (cc (mg-state), t-cond-list))},
\quad \text{MG-MAX-CTRL-STK-SIZE,}
\quad \text{MG-MAX-TEMP-STK-SIZE,}
\quad \text{MG-WORD-SIZE,}
\quad \text{‘run))}
\end{align*}\]
cons ('i,
    value (caddr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))
cons ('array-size,
    tag ('int,
    caddr (call-actuals (stmt)))))
cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
    mg-alist (mg-state)))))

    tag ('pc,
    cons (subr, length (code (cinfo))
    + 5)),

    ctrl-stk),
push (tag ('nat,
    untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
    mg-alist (mg-state)))))),

    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: non-negative-integerp-small-naturalp
(integerp (x)
∧ (∼ negativep (x))
∧ small-integerp (y, n)
∧ (y ∈ N)
∧ (idifference (y, x) ≠ 0))
→ small-naturalp (x, n)

THEOREM: mg-index-array-step-17-no-error
((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-index-array)
∧ 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)
∧ (~ not (negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))))))
∧ (idifference (caddr (call-actuals (stmt)),
   untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state)))))) ≠ 0))
→ (p-step (p-state (tag ('pc, 'mg-index-array . 11)),
   push (p-frame (list (cons ('ans, value (car (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('a, value (cadr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('i, value (caddr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('array-size, tag ('int, caddr (call-actuals (stmt)))),
   cons ('temp-1, mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))),
   tag ('pc, cons (subr, length (code (cinfo)) + 5)),
   ctrl-stk),
   push (tag ('nat, 
   untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state)))))),
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-index-array . 12)),
push (p-frame (list (cons ('ans,
  value (car (call-actuals (stmt))),
  bindings (top (ctrl-stk))))),
  cons ('a,
  value (cadr (call-actuals (stmt))),
  bindings (top (ctrl-stk))))),
  cons ('i,
  value (caddr (call-actuals (stmt))),
  bindings (top (ctrl-stk))))),
  cons ('array-size,
    tag ('int,
      caddr (call-actuals (stmt)))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
      mg-alist (mg-state)))))),
    tag ('pc,
      cons (subr, length (code (cinfo))
        + 5))),
ctrl-stk),
push (tag ('nat,
  untag (value (cadr (call-actuals (stmt))),
    bindings (top (ctrl-stk))))
  + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
     mg-alist (mg-state))))),
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-index-array-index-lessp-temp-stk-length

\[
(((\text{car} (\text{stmt}) = '\text{predefined-proc-call-mg}) \\
\land (\text{call-name} (\text{stmt}) = '\text{mg-index-array}) \\
\land \text{ok-mg-statement} (\text{stmt}, r\text{-cond-list}, name\text{-alist}, proc\text{-list}) \\
\land \text{ok-mg-statep} (mg\text{-state}, r\text{-cond-list}) \\
\land \text{mg-vars-list-ok-in-p-state} (mg\text{-alist} (mg\text{-state}), \\
\quad \text{bindings} (\text{top} (ctrl\text{-stk})), \\
\quad \text{temp\text{-stk} }) \\
\land \text{signatures-match} (mg\text{-alist} (mg\text{-state}), name\text{-alist}) \\
\land (\neg \text{negativep} (\text{untag} (\text{caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})))), \\
\quad \text{mg\text{-alist} (mg\text{-state})))))) \\
\land (\text{idifference} (\text{caddr} (\text{call-actuals} (\text{stmt})), \\
\quad \text{untag} (\text{mg\text{-to\text{-p\text{-simple\text{-literal}}} (caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})))), \\
\quad \text{mg\text{-alist} (mg\text{-state})))))) \neq 0) \\
\rightarrow (((\text{untag} (\text{value} (\text{cadr} (\text{call-actuals} (\text{stmt}))), \\
\quad \text{bindings} (\text{top} (ctrl\text{-stk})))) \\
\quad + \text{untag} (\text{mg\text{-to\text{-p\text{-simple\text{-literal}}} (caddr} (\text{assoc} (\text{caddr} (\text{call-actuals} (\text{stmt})))), \\
\quad \text{mg\text{-alist} (mg\text{-state})))))) \\
\quad < \text{length} (\text{temp\text{-stk} }) \\
= \text{t})
\]

Theorem: mg-index-array-step-18-no-error

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

\[\rightarrow (\text{p-step (p-state (tag ('pc, '}(mg-index-array . 12)), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('a, value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('i, value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk))))), cons ('array-size, tag ('int, caddr (call-actuals (stmt)))), cons ('temp-i, mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))), mg-alist (mg-state))))), tag ('pc, cons (subr, length (code (cinfo)) + 5))),
\]

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

\(\text{push (tag ('nat, untag (value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk)))) + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))), mg-alist (mg-state))))), map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk))), translate-proc-list (proc-list), list (list ('c-c, mg-cond-to-p-nat (cc (mg-state), t-cond-list))), MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, 'run))\)

\[= \text{p-state (tag ('pc, '}(mg-index-array . 13)),}\]
push (p-frame (list (cons ('ans,
               value (car (call-actuals (stmt)),
               bindings (top (ctrl-stk))))),
cons ('a,
               value (cadr (call-actuals (stmt)),
               bindings (top (ctrl-stk))))),
cons ('i,
               value (caddr (call-actuals (stmt)),
               bindings (top (ctrl-stk))))),
cons ('array-size,
               tag ('int,
               caddr (call-actuals (stmt))))),
cons ('temp-i,
               mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                  mg-alist (mg-state))))),
               tag ('pc,
               cons (subr, length (code (cinfo))
                    + 5)),
               ctrl-stk),
push (rget (untag (value (cadr (call-actuals (stmt)),
               bindings (top (ctrl-stk))))
               + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                  mg-alist (mg-state))))),
               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))

**Theorem:** mg-index-array-step-19-no-error

\[ ((n \neq 0) \land \neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))} \land \text{car (stmt) = 'predefined-proc-call-mg} \]
∧ (call-name (stmt) = 'mg-index-array)
∧ 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state)))))))
∧ (idifference (caddr (call-actuals (stmt))),
   untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))) ≠ 0))
→ (p-step (p-state (tag ('pc, 'mg-index-array . 13)),
   push (p-frame (list (cons ('ans,
   value (car (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('a,
   value (cadr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('i,
   value (caddr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('array-size,
   tag ('int,
   caddr (call-actuals (stmt))),
   cons ('temp-i,
   mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))),
   tag ('pc,
   cons (subr, length (code (CINFO))
   + 5))),
   ctrl-stk),
   push (rget (untag (value (cadr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))
   + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))),
   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)
   (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state)))))))
   (idifference (caddr (call-actuals (stmt))),
   untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
   mg-alist (mg-state))))))) ≠ 0))
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-index-array . 14)),
   push (p-frame (list (cons (’ans,
      value (car (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons (’a,
      value (cadr (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons (’i,
      value (caddr (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons (’array-size,
      tag (’int,
         caddr (call-actuals (stmt))))),
   cons (’temp-i,
      mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
         mg-alist (mg-state))))))},
tag (’pc,
   cons (subr, length (code (cinfo))
      + 5)),
ctrl-stk),
push (value (car (call-actuals (stmt)),
   bindings (top (ctrl-stk)))),
push (rget (untag (value (cadr (call-actuals (stmt)),
   bindings (top (ctrl-stk)))))
   + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
         mg-alist (mg-state))))),
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)),
map-down-values (mg-alist (mg-state),
   bindings (top (ctrl-stk)),
   temp-stk)),
Theorem: mg-index-array-step-20-no-error

\((n \not\equiv 0)\)
\(\land\) \((-\text{resources-inadequatep}(stmt,\)
\(\text{proc-list},\)
\(\text{list}(\text{length}(\text{temp-stk}),\)
\(\text{p-ctrl-stk-size}(\text{ctrl-stk})))))\)
\(\land\) \((\text{car}(stmt) = \text{predefined-proc-call-mg})\)
\(\land\) \((\text{call-name}(stmt) = \text{mg-index-array})\)
\(\land\) \((\text{ok-mg-statement}(stmt, r-cond-list, name-alist, proc-list))\)
\(\land\) \((\text{ok-mg-def-plistp}(\text{proc-list}))\)
\(\land\) \((\text{ok-mg-statep}(\text{mg-state}, r-cond-list))\)
\(\land\) \((\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))\)
\(=\) \(\text{append}(\text{code}(\text{translate}(\text{cinfo}, t-cond-list, stmt, proc-list)),\)
\(\text{code2})))\)
\(\land\) \((\text{user-defined-procp}(\text{subr}, \text{proc-list}))\)
\(\land\) \((\text{listp}(\text{ctrl-stk}))\)
\(\land\) \((\text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))))\)
\(\land\) \((\text{signatures-match}(\text{mg-alist}(\text{mg-state}), name-alist))\)
\(\land\) \((\text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}),\)
\(\text{bindings}(\text{top}(\text{ctrl-stk})),\)
\(\text{temp-stk})))\)
\(\land\) \((\text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))))\)
\(\land\) \((\text{normal}(\text{mg-state})))\)
\(\land\) \((\neg\text{negativep}(\text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{caddr}(\text{call-actuals}(stmt)),\)
\(\text{mg-alist}(\text{mg-state}))))))))\)
\(\land\) \((\text{idifference}(\text{caddr}(\text{call-actuals}(stmt)),\)
\(\text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{caddr}(\text{call-actuals}(stmt)),\)
\(\text{mg-alist}(\text{mg-state})))))) \not\equiv 0)\))
\(\rightarrow\) \((\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, \text{'mg-index-array . 14})),\)
\(\text{push}(\text{p-frame}(\text{list}(\text{cons}(\text{'ans},\)
\(\text{value}(\text{car}(\text{call-actuals}(stmt)),\)
\(\text{bindings}(\text{top}(\text{ctrl-stk}))))),\)
\(\text{cons}(\text{'a},\)
\(\text{value}(\text{cadr}(\text{call-actuals}(stmt))))),\)
\(\text{list}(\text{'c-c}),\)
\(\text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}, t-cond-list)))\),\)
\(\text{MG-MAX-CTRL-STK-SIZE},\)
\(\text{MG-MAX-TEMP-STK-SIZE},\)
\(\text{MG-WORD-SIZE},\)
\(\text{'run})))\)
\(\text{bindings}(\text{top}(\text{ctrl-stk})),\)
\(\text{temp-stk})))\),
\(\text{translate-proc-list}(\text{proc-list}),\)
\(\text{list}(\text{list}(\text{'c-c}),\)
\(\text{mg-max-ctrl-stk-size},\)
\(\text{mg-max-temp-stk-size},\)
\(\text{mg-word-size},\)
\(\text{'run})\)\)
bindings (top (ctrl-stk))), cons ('i, value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk))),
cons ('array-size, tag ('int, caddr (call-actuals (stmt)))), cons ('temp-i, 
mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))), mg-alist (mg-state))))),
tag ('pc, cons (subr, length (code (cinfo))
  + 5)),
ctrl-stk), push (value (car (call-actuals (stmt))), bindings (top (ctrl-stk))),
push (rget (untag (value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk)))))
  + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))),
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-index-array . 15)), push (p-frame (list (cons ('ans, value (car (call-actuals (stmt))), bindings (top (ctrl-stk))))),
cons ('a, value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk))))),
cons ('i, value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk)))),
cons ('array-size,
tag ('int, caddr (call-actuals (stmt))),
    cons ('temp-i, mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))),
    tag ('pc, cons (subr, length (code (cinfo)) + 5)),
    ctrl-stk),
    rput (rget (untag (value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk)))))
    + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))),
    map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk),
    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))

THEOREM: mg-index-array-steps-21-22-no-error
((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-index-array)
 ∧ 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
  mg-alist (mg-state)))))))
∧ (idifference (caddr (call-actuals (stmt)),
  untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
  mg-alist (mg-state)))))) ≠ 0))
→ (p-step (p-step (p-state (tag ('pc, 'mg-index-array . 15)),
  push (p-frame (list (cons ('ans,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))),
  cons ('a,
    value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))),
  cons ('i,
    value (caddr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))),
  cons ('array-size,
    tag ('int,
    caddr (call-actuals (stmt))))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
      mg-alist (mg-state))))),
  tag ('pc,
    cons (subr,
      length (code (cinfo))
      + 5))),
  ctrl-stk),
  rput (rget (untag (value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))
  + untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt),
    mg-alist (mg-state)))))))
  map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)),
  untag (value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk)))))
  map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)))
Theorem: mg-index-array-push-cc

\[
((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{\texttt{\texttt{ predefined-proc-call-mg}}})
\land (\text{call-name}(stmt) = \text{\texttt{mg-index-array}})
\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}(\text{translate-def-body}(assoc(subr, proc-list), proc-list)))
\begin{align*}
&= \text{append} \left( \text{code} \left( \text{translate} \left( \text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list} \right), \text{code2} \right) \right) \\
&\land \text{user-defined-procp} \left( \text{subr}, \text{proc-list} \right) \\
&\land \text{listp} \left( \text{ctrl-stk} \right) \\
&\land \text{all-cars-unique} \left( \text{mg-alist} \left( \text{mg-state} \right) \right) \\
&\land \text{signatures-match} \left( \text{mg-alist} \left( \text{mg-state} \right), \text{name-alist} \right) \\
&\land \text{normal} \left( \text{mg-state} \right) \\
\rightarrow \quad & \left( \text{p-step} \left( \text{p-state} \left( \text{tag} \left( \text{'pc}, \text{cons} \left( \text{subr}, \text{length} \left( \text{code} \left( \text{cinfo} \right) \right) + 5 \right) \right), \text{ctrl-stk}, \text{temp-stk}, \text{translate-proc-list} \left( \text{proc-list} \right), \text{list} \left( \text{list} \left( \text{'c-c, cc-value} \right) \right), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}, \text{'run} \right) \right) \\
&= \text{p-state} \left( \text{tag} \left( \text{'pc}, \text{cons} \left( \text{subr}, \text{length} \left( \text{code} \left( \text{cinfo} \right) \right) + 6 \right) \right), \text{ctrl-stk}, \text{push} \left( \text{cc-value}, \text{temp-stk} \right), \text{translate-proc-list} \left( \text{proc-list} \right), \text{list} \left( \text{list} \left( \text{'c-c, cc-value} \right) \right), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}, \text{'run} \right) \\
\end{align*}

\text{Theorem: mg-index-array-sub1-cc} \\
\left( \left( n \not\equiv 0 \right) \right) \\
\land \quad \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) \\
\land \quad \left( \text{car} \left( \text{stmt} \right) = \text{'predefined-proc-call-mg} \right) \\
\land \quad \left( \text{call-name} \left( \text{stmt} \right) = \text{'mg-index-array} \right) \\
\land \quad \text{ok-mg-statement} \left( \text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list} \right) \\
\land \quad \text{ok-mg-def-plistp} \left( \text{proc-list} \right) \\
\land \quad \text{ok-mg-statep} \left( \text{mg-state}, \text{r-cond-list} \right) \\
\land \quad \left( \text{code} \left( \text{translate-def-body} \left( \text{assoc} \left( \text{subr}, \text{proc-list}, \text{proc-list} \right) \right) = \text{append} \left( \text{code} \left( \text{translate} \left( \text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list} \right), \text{code2} \right) \right) \right) \right) \\
\land \quad \text{user-defined-procp} \left( \text{subr}, \text{proc-list} \right) \\
\land \quad \text{listp} \left( \text{ctrl-stk} \right) \\
\land \quad \text{all-cars-unique} \left( \text{mg-alist} \left( \text{mg-state} \right) \right) \\
\land \quad \text{signatures-match} \left( \text{mg-alist} \left( \text{mg-state} \right), \text{name-alist} \right)
\[ \begin{align*}
&\wedge \text{normal}(mg-state) \\
&\wedge (cc-value \in \text{list}(\text{'nat 1}, \text{'nat 2})) \\
\rightarrow \quad \text{(p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)), ctrl-stk,}} \\
&\quad \quad \text{push (cc-value, temp-stk),}} \\
&\quad \quad \text{translate-proc-list (proc-list),}} \\
&\quad \quad \text{list (list ('}c-c, cc-value)),}} \\
&\quad \quad \text{MG-MAX-CTRL-STK-SIZE,}} \\
&\quad \quad \text{MG-MAX-TEMP-STK-SIZE,}} \\
&\quad \quad \text{MG-WORD-SIZE,}} \\
&\quad \quad \text{'run))) \\
&= \quad \text{p-state (tag ('pc, cons (subr, length (code (cinfo)) + 7)),}} \\
&\quad \quad \text{ctrl-stk,}} \\
&\quad \quad \text{push (tag ('nat, untag (cc-value) - 1), temp-stk),}} \\
&\quad \quad \text{translate-proc-list (proc-list),}} \\
&\quad \quad \text{list (list ('}c-c, cc-value)),}} \\
&\quad \quad \text{MG-MAX-CTRL-STK-SIZE,}} \\
&\quad \quad \text{MG-MAX-TEMP-STK-SIZE,}} \\
&\quad \quad \text{MG-WORD-SIZE,}} \\
&\quad \quad \text{'run))}
\end{align*} \]

**THEOREM**: mg-index-array-last-step-error-case
\[ \begin{align*}
&((n \neq 0) \\
&\wedge (\neg \text{resources-inadequatep (stmt,}} \\
&\quad \quad \text{proc-list,}} \\
&\quad \quad \text{list (length (temp-stk),}} \\
&\quad \quad \text{p-ctrl-stk-size (ctrl-stk))})) \\
\wedge (\text{car (stmt) = 'predefined-proc-call-mg}) \\
\wedge (\text{call-name (stmt) = 'mg-index-array}) \\
\wedge \quad \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
\wedge \quad \text{ok-mg-def-plistp (proc-list)} \\
\wedge \quad \text{ok-mg-statep (mg-state, r-cond-list)} \\
\wedge (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))}} \\
&\quad \quad = \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),}} \\
&\quad \quad \text{code2))} \\
\wedge \quad \text{user-defined-procp (subr, proc-list)} \\
\wedge \quad \text{listp (ctrl-stk)} \\
\wedge \quad \text{all-cars-unique (mg-alist (mg-state))} \\
\wedge \quad \text{signatures-match (mg-alist (mg-state), name-alist)} \\
\wedge \quad \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),}} \\
&\quad \quad \text{bindings (top (ctrl-stk)),}} \\
&\quad \quad \text{temp-stk)} \\
\wedge \quad \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
\wedge \quad \text{normal (mg-state)}
\end{align*} \]
\(\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state))))))}\)
\(\lor \text{idifference (caddr (call-actuals (stmt))), untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)), mg-alist (mg-state)))))) \approx 0)\)
\(\rightarrow (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo))) + 7)), ctrl-stk, push (tag ('nat, untag ('(nat 1)) - 1), map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)), translate-proc-list (proc-list), '((c-c (nat 1)))), MG-MAX-CTRL-STK-SIZE, MG-MAX-TEMP-STK-SIZE, MG-WORD-SIZE, 'run))\)
\(= \text{p-state (tag ('pc, cons (subr, if normal (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))) then length (code (translate (cinfo, t-cond-list, stmt, proc-list)))) else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list)))), append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)) endif),}\)
**Theorem:** simple-typed-identifier-type-equivalence

\[ \text{simple-typed-identifierp} \ (b, \ type, \ alist) \rightarrow ((\text{cadr} \ (\text{assoc} \ (b, \ alist))) = \text{type} = \text{t}) \]

**Theorem:** mg-index-array-step-25-no-error

\[ ((n \not\approx 0) \land (\neg \text{resources-inadequatep} \ (stmt, \ proc-list, \ \text{list} \ (\text{length} \ (temp-stk), \ p-ctrl-stk-size \ (ctrl-stk)))) \]

\[ \land (\text{car} \ (stmt) = '\text{predefined-proc-call-mg}') \]
\[ \land (\text{call-name} \ (stmt) = '\text{mg-index-array}') \]
\[ \land \text{ok-mg-statement} \ (stmt, \ r-cond-list, \ name-alist, \ proc-list) \]
\[ \land \text{ok-mg-def-plistp} \ (proc-list) \]
\[ \land \text{ok-mg-statep} \ (mg-state, \ r-cond-list) \]
\[ \land (\text{code} \ (\text{translate-def-body} \ (\text{assoc} \ (subr, \ proc-list), \ proc-list)) = \text{append} \ (\text{code} \ (\text{translate} \ (cinfo, \ t-cond-list, \ stmt, \ proc-list)), \ code2)) \]
\[ \land \text{user-defined-procp} \ (subr, \ proc-list) \]
\[ \land \text{listp} \ (ctrl-stk) \]
\[ \land \text{all-cars-unique} \ (\text{mg-alist} \ (mg-state)) \]
\[\begin{align*}
\wedge & \text{signatures-match \( (mg-alist \ (mg-state), \ name-alist) \)} \\
\wedge & \text{mg-vars-list-ok-in-p-state \( (mg-alist \ (mg-state), \)} \\
& \text{\hspace{1em} \text{bindings \( (top \ (ctrl-stk)) \), temp-stk) \)} \\
\wedge & \text{no-p-aliasing \( (bindings \ (top \ (ctrl-stk)), \ mg-alist \ (mg-state)) \)} \\
\wedge & \text{normal \( (mg-state) \)} \\
\wedge & \text{\neg \text{negativep \( (untag \ (mg-to-p-simple-literal \ (caddr \ (assoc \ (caddr \ (call-actuals \ (stmt)), \ mg-alist \ (mg-state)))))) \)} \\
\wedge & \text{(idifference \( (caddr (call-actuals \ (stmt)), \)} \\
& \text{untag \( (mg-to-p-simple-literal \ (caddr \ (assoc \ (caddr \ (call-actuals \ (stmt)), \ mg-alist \ (mg-state)))))) \neq 0 \)} \\
\rightarrow & \text{(p-step \( (p-state \ (tag \ (pc, \ cons \ (subr, \ length \ (code \ (cinfo)) \ + \ 7)), \)} \\
& \text{ctrl-stk, \)} \\
& \text{\hspace{1em} push \( (tag \ (nat, \)} \\
& \text{\hspace{2em} untag \( (mg-cond-to-p-nat \ (cc \ (mg-state), \)} \\
& \text{\hspace{3em} t-cond-list) \) \) \)} \\
& \text{\hspace{2em} rput \( (rget \ (untag \ (value \ (cadr \ (call-actuals \ (stmt)), \)} \\
& \text{\hspace{3em} bindings \( (top \ (ctrl-stk))) \) \)} \\
& \text{\hspace{2em} \hspace{1em} + \hspace{1em} untag \( (mg-to-p-simple-literal \ (caddr \ (assoc \ (caddr \ (call-actuals \ (stmt)), \)} \\
& \text{\hspace{3em} mg-alist \( (mg-state)))))) \)} \\
& \text{map-down-values \( (mg-alist \ (mg-state), \)} \\
& \text{\hspace{1em} bindings \( (top \ (ctrl-stk)), \)} \\
& \text{\hspace{2em} temp-stk) \)} \\
& \text{untag \( (value \ (car \ (call-actuals \ (stmt)), \)} \\
& \text{bindings \( (top \ (ctrl-stk))) \) \)} \\
& \text{map-down-values \( (mg-alist \ (mg-state), \)} \\
& \text{\hspace{1em} bindings \( (top \ (ctrl-stk)), \)} \\
& \text{\hspace{2em} temp-stk) \)} \\
& \text{translate-proc-list \( (proc-list), \)} \\
& \text{list \( (\mathrm{\text{\texttt{\`\texttt{c-c}}, \}} \\
& \text{\hspace{1em} mg-cond-to-p-nat \ (cc \ (mg-state), \ t-cond-list恤)), \)} \\
& \text{MG-MAX-CTRL-STK-SIZE, \)} \\
& \text{MG-MAX-TEMP-STK-SIZE, \)} \\
& \text{MG-WORD-SIZE, \)} \\
& \text{\texttt{\`run} \)) \)} \\
& \text{p-state \( (tag \ (pc, \)} \\
& \text{\hspace{1em} cons \( (subr, \)} \\
& \text{\hspace{2em} if \ normal \( (mg-meaning-r \ (stmt, \)} \\
& \text{\hspace{3em} proc-list, \)} \\
& \text{\hspace{4em} mg-state, \)} \\
& \text{\hspace{5em} n, \)} \\
& \text{\hspace{2em} list \( (length \ (temp-stk), \)} \\
& \text{\hspace{3em} p-ctrl-stk-size \( (ctrl-stk))) \)) \)} \\
& \text{then \ length \( (code \ (translate \ (cinfo, \)} \\
& \text{41}}

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

43
Theorem: mg-array-element-assignment-args-have-simple-mg-type-refps

\((\text{car} \ (\text{stmt}) = \text{\'predefined-proc-call-mg})\)
∧ (call-name (stmt) = 'mg-array-element-assignment)
∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
∧ ok-mg-statem (mg-state, r-cond-list)
∧ signatures-match (mg-alist (mg-state), name-alist))
→ (array-identifierp (car (call-actuals (stmt))), mg-alist (mg-state))
  ∧ int-identiferp (cadr (call-actuals (stmt))), mg-alist (mg-state))
  ∧ simple-typed-identifierp (caddr (call-actuals (stmt))),
      array-elemtype (cadr (assoc (car (call-actuals (stmt))),
      mg-alist (mg-state))))
  ∧ (cadddr (call-actuals (stmt)) = array-length (cadr (assoc (car (call-actuals (stmt))),
      mg-alist (mg-state))))

THEOREM: mg-array-element-assignment-arg3-simple
 ((car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-array-element-assignment)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-statem (mg-state, r-cond-list)
 ∧ signatures-match (mg-alist (mg-state), name-alist))
→ simple-identifierp (caddr (call-actuals (stmt))), mg-alist (mg-state))

THEOREM: mg-array-element-assignment-args-definedp
 ((car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-array-element-assignment)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-statem (mg-state, r-cond-list)
 ∧ signatures-match (mg-alist (mg-state), name-alist))
→ (definedp (car (call-actuals (stmt))), mg-alist (mg-state))
  ∧ definedp (cadr (call-actuals (stmt))), mg-alist (mg-state))
  ∧ definedp (caddr (call-actuals (stmt))), mg-alist (mg-state)))

THEOREM: mg-array-element-assignment-arg4-small-integerp
 ((car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-array-element-assignment)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-statem (mg-state, r-cond-list)
 ∧ signatures-match (mg-alist (mg-state), name-alist))
→ small-integerp (cadddr (call-actuals (stmt)), 32)

THEOREM: not-zerop-mg-array-element-assignment-arg4
 ((car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-array-element-assignment)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-statem (mg-state, r-cond-list)
\[ \begin{align*}
&\land \text{signatures-match (mg-alist (mg-state), name-alist))} \\
&\rightarrow ((\text{caddr (call-actuals (stmt))} \in \mathbb{N}) \\
&\land (\text{caddr (call-actuals (stmt))} \neq 0)) \\
\end{align*} \]

**Theorem:** mg-array-element-assignment-steps-1-4
\[ ((n \neq 0) \\
\land (\neg \text{resources-inadequatep (stmt,} \\
\text{proc-list,} \\
\text{list (length (temp-stk),} \\
\text{p-ctrl-stk-size (ctrl-stk)))) \\
\land (\text{car (stmt) = 'predefined-proc-call-mg}) \\
\land (\text{call-name (stmt) = 'mg-array-element-assignment}) \\
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list) } \\
\land \text{ok-mg-def-plistp (proc-list) } \\
\land \text{ok-mg-statep (mg-state, r-cond-list) } \\
\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))} \\
\quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),} \\
\text{code2)}) \\
\land \text{user-defined-procp (subr, proc-list) } \\
\land \text{listp (ctrl-stk)} \\
\land \text{all-cars-unique (mg-alist (mg-state)) } \\
\land \text{signatures-match (mg-alist (mg-state), name-alist) } \\
\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),} \\
\text{bindings (top (ctrl-stk)),} \\
\text{temp-stk)} \\
\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state)) } \\
\land \text{normal (mg-state)) } \\
\rightarrow (\text{p-step (p-step (p-step (p-step (map-down (mg-state,} \\
\text{proc-list,} \\
\text{ctrl-stk,} \\
\text{temp-stk,} \\
\text{tag ('pc,} \\
\text{cons (subr,} \\
\text{length (code (cinfo))},} \\
\text{t-cond-list))))))} \\
\quad = \text{p-state (tag ('pc,} \\
\text{cons (subr, length (code (cinfo)) + 4)),} \\
\text{ctrl-stk,} \\
\text{push (tag ('int, caddr (call-actuals (stmt))}, \\
\text{push (value (caddr (call-actuals (stmt))}, \\
\text{bindings (top (ctrl-stk))}, \\
\text{push (value (cadr (call-actuals (stmt))}, \\
\text{bindings (top (ctrl-stk))}, \\
\text{push (value (car (call-actuals (stmt))}, \\
\text{bindings (top (ctrl-stk))},} \\
\text{46})} \]
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-array-element-assignment-step-5
((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-array-element-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)
∧ 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, cons (subr, length (code (cinfo)) + 4)),
   ctrl-stk),
   push (tag (’int, caddr (call-actuals (stmt))),
   push (value (caddr (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
   push (value (cadr (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
   push (value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk))),
   map-down-values (mg-alist (mg-state)),
   47
Theorem: mg-array-element-assignment-steps-6-8

\[(n \neq 0) \land (\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), bindings (top (ctrl-stk)), temp-stk))})\]
\[
\begin{align*}
\text{p-ctrl-stk-size (ctrl-stk))} & \\
\land (\text{car (stmt) = 'predefined-proc-call-mg}) & \\
\land (\text{call-name (stmt) = 'mg-array-element-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)} \\
\quad = \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-step (p-step (p-state (tag ('pc,} & \\
\quad '\text{mg-array-element-assignment . 0)),} & \\
\quad \text{push (p-frame (cons (cons ('a,} & \\
\quad \quad \text{value (car (call-actuals (stmt)),} & \\
\quad \quad \text{bindings (top (ctrl-stk))},} & \\
\quad \quad \text{cons (cons ('i,} & \\
\quad \quad \quad \text{value (cadr (call-actuals (stmt)),} & \\
\quad \quad \quad \text{bindings (top (ctrl-stk))},} & \\
\quad \quad \quad \text{cons (cons ('value,} & \\
\quad \quad \quad \quad \text{value (caddr (call-actuals (stmt)),} & \\
\quad \quad \quad \quad \text{bindings (top (ctrl-stk))},} & \\
\quad \quad \quad \quad \text{cons (cons ('array-size,} & \\
\quad \quad \quad \quad \quad \text{tag ('int,} & \\
\quad \quad \quad \quad \quad \quad \text{caddr (call-actuals (stmt)} & \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad '((temp-i} & \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \text{nat} & \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad 0))))))))}}, & \\
\quad \quad \text{tag ('pc,} & \\
\quad \quad \text{cons (subr,} & \\
\quad \quad \text{length (code (cinfo))} + 5)))}, & \\
\quad \text{ctrl-stk),} & \\
\text{map-down-values (mg-alist (mg-state),} & \\
\text{bindings (top (ctrl-stk),} & \\
\text{temp-stk),} & \\
\end{align*}
\]
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-array-element-assignment . 3)),
push (p-frame (list (cons ('a,
            value (car (call-actuals (stmt))),
            bindings (top (ctrl-stk))))),
    cons ('i,
            value (cadr (call-actuals (stmt))),
            bindings (top (ctrl-stk))))),
    cons ('value,
            value (caddr (call-actuals (stmt))),
            bindings (top (ctrl-stk))))),
    cons ('array-size,
            tag ('int,
            caddr (call-actuals (stmt)))),
    cons ('temp-i,
            mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
            mg-alist (mg-state))))),
    tag ('pc,
            cons (subr, length (code (cinfo))
                  + 5)),
    ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
            mg-alist (mg-state)))))
    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-array-element-assignment-steps-9-12-neg-index
((n \neq 0)
\(\neg \) resources-inadequatep \((stmt, proc-list, list \text{ length} (temp-stk), \text{p-ctrl-stk-size} (ctrl-stk)))

\(\text{car} \ (stmt) = \ '\text{predefined-proc-call-mg}'\)

\(\text{call-name} \ (stmt) = \ '\text{mg-array-element-assignment}'\)

ok-mg-statement \((stmt, r-cond-list, name-alist, proc-list)\)

ok-mg-def-plistp \((proc-list)\)

ok-mg-statep \((mg-state, r-cond-list)\)

\((\text{code} \ (\text{translate-def-body} \ (\text{assoc} \ (\text{subr}, proc-list), proc-list))) = \ \text{append} \ (\text{code} \ (\text{translate} \ (\text{cinfo}, t-cond-list, stmt, proc-list)), code2))\)

user-defined-procp \((\text{subr}, proc-list)\)

listp \((ctrl-stk)\)

all-cars-unique \((\text{mg-alist} \ (mg-state))\)

signatures-match \((\text{mg-alist} \ (mg-state), name-alist)\)

mg-vars-list-ok-in-p-state \((\text{mg-alist} \ (mg-state), \text{bindings} \ (\text{top} \ (ctrl-stk)), temp-stk)\)

no-p-aliasing \((\text{bindings} \ (\text{top} \ (ctrl-stk)), \text{mg-alist} \ (mg-state))\)

normal \((mg-state)\)

negativep \((\text{untag} \ (\text{mg-to-p-simple-literal} \ (\text{caddr} \ (\text{assoc} \ \text{cadr} \ (\text{call-actuals} \ (stmt)), mg-alist \ (mg-state)))))))\)

\(\rightarrow\) \((\text{p-step} \ (\text{p-step} \ (\text{p-step} \ (\text{p-step} \ (\text{p-state} \ \text{tag} \ (\ '\text{pc}, (mg-array-element-assignment}, 3))\)),

\text{push} \ (\text{p-frame} \ (\text{list} \ \text{cons} \ (\ 'a, \text{value} \ (\text{car} \ (\text{call-actuals} \ (stmt)), \text{bindings} \ (\text{top} \ (ctrl-stk)))))\),

\text{cons} \ (\ 'i, \text{value} \ (\text{cadr} \ (\text{call-actuals} \ (stmt)), \text{bindings} \ (\text{top} \ (ctrl-stk)))))\),

\text{cons} \ (\ 'value, \text{value} \ (\text{caddr} \ (\text{call-actuals} \ (stmt)), \text{bindings} \ (\text{top} \ (ctrl-stk)))))\),

\text{cons} \ (\ 'array-size, \text{tag} \ (\ '\text{int}, \text{cadddr} \ (\text{call-actuals} \ (stmt))))))\),

\text{cons} \ (\ 'temp-i, \text{mg-to-p-simple-literal} \ (\text{caddr} \ (\text{assoc} \ \text{cadr} \ mg-alist, mg-alist))))\)

\text{tag} \ (\ 'pc, \text{cons} \ (\ 'subr, \text{length} \ (\text{code} \ (\text{cinfo}))))\)
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt) mg-alist (mg-state))))),
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, length (code (cinfo)) + 5)),
ctrl-stk,
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list ('c-c, 'nat 1)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,

'Theorem: mg-array-element-assignment-steps-9-11-no-error

((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-array-element-assignment)
  ∧ 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)
∧ (~ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state)))))))
→ (p-step (p-step (p-step (p-state (tag ('pc,
  (mg-array-element-assignment
   . 3)),
    push (p-frame (list (cons ('a,
      value (car (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('i,
      value (cadr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('value,
      value (caddr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('array-size,
      tag ('int,
        caddr (call-actuals (stmt))))),
    cons ('temp-i,
      mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
        mg-alist (mg-state))))
      tag ('pc,
        cons (subr,
          length (code (cinfo))
          + 5))))),
  ctrl-stk),
  push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state))))),
  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))))
53
Theorem: mg-array-element-assignment-step-12-no-error

\[ ((n \neq 0) \wedge (\neg \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))}) \wedge \text{car (stmt)} = \text{predefined-proc-call-mg}) \wedge \text{call-name (stmt) = mg-array-element-assignment}) \wedge \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \]
∧ 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                            mg-alist (mg-state)))))))
→ (p-step (p-state (tag ('pc, 'mg-array-element-assignment . 6)),
              push (p-frame (list (cons ('a,
                                      value (car (call-actuals (stmt)),
                                 bindings (top (ctrl-stk))))),
                  cons ('i,
                     value (cadr (call-actuals (stmt)),
                             bindings (top (ctrl-stk))))),
              cons ('value,
                 value (caddr (call-actuals (stmt)),
                        bindings (top (ctrl-stk))))),
              cons ('array-size,
                 tag ('int,
                      cadddr (call-actuals (stmt))))),
              cons ('temp-i,
                 mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                      mg-alist (mg-state))))),
              tag ('pc,
                 cons (subr, length (code (cinfo))
                        + 5)),
               ctrl-stk),
              push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                       mg-alist (mg-state))))),
              push (tag ('int, cadddr (call-actuals (stmt)) ),
                     map-down-values (mg-alist (mg-state),
                                      bindings (top (ctrl-stk)),
                                      temp-stk)),
              translate-proc-list (proc-list),
              55)
Theorem: mg-array-element-assignment-step-13-index-error
((n ≠ 0)
\[\neg \text{resources-inadequate} (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk})))\]

\[\text{car} (\text{stmt}) = \text{'predefined-proc-call-mg}\]

\[\text{call-name} (\text{stmt}) = \text{'mg-array-element-assignment}\]

\[\text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})\]

\[\text{ok-mg-def-plistp} (\text{proc-list})\]

\[\text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list})\]

\[\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list})) = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})\]

\[\text{user-defined-procp} (\text{subr}, \text{proc-list})\]

\[\text{listp} (\text{ctrl-stk})\]

\[\text{all-cars-unique} (\text{mg-alist} (\text{mg-state}))\]

\[\text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist})\]

\[\text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \text{bindings} (\text{top} (\text{ctrl-stk})), \text{temp-stk})\]

\[\text{no-p-aliasing} (\text{bindings} (\text{top} (\text{ctrl-stk})), \text{mg-alist} (\text{mg-state}))\]

\[\text{normal} (\text{mg-state})\]

\[\neg \neg \text{negativep} (\text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} \text{cadr} (\text{call-actuals} (\text{stmt})), \text{mg-alist} (\text{mg-state})))))\]

\[\neg \text{idifference} (\text{cadddr} (\text{call-actuals} (\text{stmt})), \text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} \text{cadr} (\text{call-actuals} (\text{stmt})), \text{mg-alist} (\text{mg-state}))))) ≃ 0)\]

\[\rightarrow (\text{p-step} (\text{p-state} (\text{tag} (\text{'pc, 'mg-array-element-assignment} \ . \ 7)), \text{push} (\text{p-frame} (\text{list} (\text{cons} (\text{'a, value} (\text{car} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk}))))), \text{cons} (\text{'i, value} (\text{cadr} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk})))))}, \text{cons} (\text{'value, value} (\text{caddr} (\text{call-actuals} (\text{stmt})), \text{bindings} (\text{top} (\text{ctrl-stk})))}, \text{cons} (\text{'array-size, value} (\text{caddadr} (\text{call-actuals} (\text{stmt}))), \text{tag} (\text{'int, caddadr} (\text{call-actuals} (\text{stmt})))), \text{cons} (\text{'temp-i, mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{cadr} (\text{call-actuals} (\text{stmt})), \text{mg-alist} (\text{mg-state})))))}, \text{tag} (\text{'pc, 57}))\]
\[
\text{cons}(\text{subr}, \text{length} (\text{code} (\text{cinfo})) + 5)),
\]
\[
\text{ctrl-stk},
\]
\[
\text{push} (\text{tag}(\text{'int},
\)
\]
\[
\text{idifference} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\]
\[
\text{untag} (\text{mg-to-p-simple-literal} (\text{caddr} (\text{assoc} (\text{cadr} (\text{call-actuals} (\text{stmt})
\text{mg-alist} (\text{mg-state})))))),
\]
\[
\text{map-down-values} (\text{mg-alist} (\text{mg-state}),
\]
\[
\text{bindings} (\text{top} (\text{ctrl-stk}),
\]
\[
\text{temp-stk}),
\]
\[
\text{translate-proc-list} (\text{proc-list}),
\]
\[
\text{list} (\text{list}(\text{'c-c},
\]
\[
\text{mg-cond-to-p-nat} (\text{cc} (\text{mg-state}, \text{t-cond-list})),
\]
\[
\text{MG-MAX-CTRL-STK-SIZE},
\]
\[
\text{MG-MAX-TEMP-STK-SIZE},
\]
\[
\text{MG-WORD-SIZE},
\]
\[
\text{'run}))
\]
\[
\text{= p-state (tag('pc,}
\]
\[
\text{'(mg-array-element-assignment . 16)),}
\]
\[
\text{push (p-frame (list (cons('a,}
\]
\[
\text{value} (\text{car} (\text{call-actuals} (\text{stmt})),
\]
\[
\text{bindings} (\text{top} (\text{ctrl-stk})))),
\]
\[
\text{cons('i,}
\]
\[
\text{value} (\text{cadr} (\text{call-actuals} (\text{stmt})),
\]
\[
\text{bindings} (\text{top} (\text{ctrl-stk})))),
\]
\[
\text{cons('value,}
\]
\[
\text{value} (\text{caddr} (\text{call-actuals} (\text{stmt})),
\]
\[
\text{bindings} (\text{top} (\text{ctrl-stk})))),
\]
\[
\text{cons('array-size,}
\]
\[
\text{tag('int,}
\]
\[
\text{caddr (call-actuals (stmt))}),
\]
\[
\text{cons('temp-i,}
\]
\[
\text{mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))
\text{mg-alist} (\text{mg-state})))))),
\]
\[
\text{tag('pc,}
\]
\[
\text{cons(subr, length (code (cinfo)) + 5))},
\]
\[
\text{ctrl-stk),
\]
\[
\text{map-down-values} (\text{mg-alist} (\text{mg-state}),
\]
\[
\text{bindings} (\text{top} (\text{ctrl-stk}),
\]
\[
\text{temp-stk}),
\]
\[
\text{translate-proc-list} (\text{proc-list}),
\]
\[
\text{list} (\text{list}(\text{'c-c},
\]
\[
\text{mg-cond-to-p-nat} (\text{cc} (\text{mg-state}, \text{t-cond-list})))),
\]
\[
58
\]

THEOREM: mg-array-element-assignment-steps-14-16-index-error

\((n \not\equiv 0)\)
\∧ (\neg \text{resources-inadequatep}(\text{stmt}, \text{proc-list},
\text{list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))))
\∧ (\text{car (stmt)} = \text{'predefined-proc-call-mg})
\∧ (\text{call-name (stmt)} = \text{'mg-array-element-assignment})
\∧ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list})
\∧ \text{ok-mg-def-plistp (proc-list)}
\∧ \text{ok-mg-statep (mg-state, r-cond-list)}
\∧ \text{(code (translate-def-body (assoc (subr, proc-list), proc-list))}
\quad = \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}
\∧ \text{user-defined-procp (subr, proc-list)}
\∧ \text{listp (ctrl-stk)}
\∧ \text{all-cars-unique (mg-alist (mg-state))}
\∧ \text{signatures-match (mg-alist (mg-state), name-alist)}
\∧ \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),
\text{bindings (top (ctrl-stk)), temp-stk})}
\∧ \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}
\∧ \text{normal (mg-state)}
\∧ (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))}))
\∧ (\text{idifference (caddr (call-actuals (stmt)),
\text{untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))} \equiv 0))
\rightarrow \text{(p-step (p-step (p-step (p-state (tag ('pc, \text{'mg-array-element-assignment
\ . 16)),
\text{push (p-frame (list (cons ('a, value (car (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('i, value (cadr (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('value, value (caddr (call-actuals (stmt))))},
\text{normal (mg-state))}))})))}
\quad \not\equiv 0)\)
\∧ \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))
\∧ (\text{car (stmt)} = \text{'predefined-proc-call-mg})
\∧ (\text{call-name (stmt)} = \text{'mg-array-element-assignment})
\∧ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list})
\∧ \text{ok-mg-def-plistp (proc-list)}
\∧ \text{ok-mg-statep (mg-state, r-cond-list)}
\∧ \text{(code (translate-def-body (assoc (subr, proc-list), proc-list))}
\quad = \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}
\∧ \text{user-defined-procp (subr, proc-list)}
\∧ \text{listp (ctrl-stk)}
\∧ \text{all-cars-unique (mg-alist (mg-state))}
\∧ \text{signatures-match (mg-alist (mg-state), name-alist)}
\∧ \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),
\text{bindings (top (ctrl-stk)), temp-stk})}
\∧ \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}
\∧ \text{normal (mg-state)}
\∧ (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))}))
\∧ (\text{idifference (caddr (call-actuals (stmt)),
\text{untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))} \equiv 0))
\rightarrow \text{(p-step (p-step (p-step (p-state (tag ('pc, 
\text{'mg-array-element-assignment
\ . 16)),
\text{push (p-frame (list (cons ('a, value (car (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('i, value (cadr (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('value, value (caddr (call-actuals (stmt))))),
\text{normal (mg-state))}))}))})
\quad \not\equiv 0)\)
\∧ \text{resources-inadequatep (stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))
\∧ (\text{car (stmt)} = \text{'predefined-proc-call-mg})
\∧ (\text{call-name (stmt)} = \text{'mg-array-element-assignment})
\∧ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list})
\∧ \text{ok-mg-def-plistp (proc-list)}
\∧ \text{ok-mg-statep (mg-state, r-cond-list)}
\∧ \text{(code (translate-def-body (assoc (subr, proc-list), proc-list))}
\quad = \quad \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2))}
\∧ \text{user-defined-procp (subr, proc-list)}
\∧ \text{listp (ctrl-stk)}
\∧ \text{all-cars-unique (mg-alist (mg-state))}
\∧ \text{signatures-match (mg-alist (mg-state), name-alist)}
\∧ \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),
\text{bindings (top (ctrl-stk)), temp-stk})}
\∧ \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}
\∧ \text{normal (mg-state)}
\∧ (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))}))
\∧ (\text{idifference (caddr (call-actuals (stmt)),
\text{untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
\text{mg-alist (mg-state))))))} \equiv 0))
\rightarrow \text{(p-step (p-step (p-step (p-state (tag ('pc, 
\text{'mg-array-element-assignment
\ . 16)),
\text{push (p-frame (list (cons ('a, value (car (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('i, value (cadr (call-actuals (stmt)),
\text{bindings (top (ctrl-stk))))),
\text{cons ('value, value (caddr (call-actuals (stmt))))),
\text{normal (mg-state))}))}))})})
\quad \not\equiv 0)\)
bindings (top (ctrl-stk))),
cons (array-size,
tag (int,
caddr (call-actuals (stmt)))),
cons (temp-i,
mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state)))))
tag (pc,
cons (subr,
length (code (cinfo)) + 5)),
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, length (code (cinfo)) + 5)),
ctrl-stk),
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk),
translate-proc-list (proc-list),
list (list (c-c, (nat 1))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

;; Need 13-15 and 16-17 no-error lemmas

THEOREM: mg-array-element-assignment-step-13-no-error
((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-array-element-assignment)
∧ 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
               mg-alist (mg-state)))))))
∧ (idifference (cadddr (call-actuals (stmt)),
   untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
               mg-alist (mg-state)))))) ≠ 0))
→ (p-step (p-state (tag ('pc,
   'mg-array-element-assignment . 7)),
   push (p-frame (list (cons ('a,
                          value (car (call-actuals (stmt)),
                          bindings (top (ctrl-stk))))),
   cons ('i,
   value (cadr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('value,
   value (caddr (call-actuals (stmt)),
   bindings (top (ctrl-stk))))),
   cons ('array-size,
   tag ('int,
   cadddr (call-actuals (stmt))))),
   cons ('temp-i,
   mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
               mg-alist (mg-state))))),
   tag ('pc,
    cons (subr, length (code (cinfo))
   + 5)) ),
   ctrl-stk),
   push (tag ('int,
idifference (caddr (call-actuals (stmt)),
untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))
mg-alist (mg-state))))),
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-array-element-assignment . 8)),
push (p-frame (list (cons ('a,
      value (car (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons ('i,
      value (cadr (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons ('value,
      value (caddr (call-actuals (stmt)),
         bindings (top (ctrl-stk))))),
   cons ('array-size,
      tag ('int,
         caddr (call-actuals (stmt))))),
   cons ('temp-i,
      mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state))))),
      tag (pc,
         cons (subr, length (code (cinfo))
+ 5))),
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-array-element-assignment-step-14-no-error

\((n \not\equiv 0)\)
\(\land (\neg \text{resources-inadequatep (stmt, \)}
  \text{proc-list,}
  \text{list (length (temp-stk),}
  \text{p-ctrl-stk-size (ctrl-stk))})\)
\(\land (\text{car (stmt) = 'predefined-proc-call-mg})\)
\(\land (\text{call-name (stmt) = 'mg-array-element-assignment})\)
\(\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}\)
\(\land \text{ok-mg-def-plistp (proc-list)}\)
\(\land \text{ok-mg-statep (mg-state, r-cond-list)}\)
\(\land (\text{code (translate-def-body (assoc (subr, proc-list), proc-list)}) = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)), code2)})\)
\(\land \text{user-defined-procp (subr, proc-list)}\)
\(\land \text{listp (ctrl-stk)}\)
\(\land \text{all-cars-unique (mg-alist (mg-state))}\)
\(\land \text{signatures-match (mg-alist (mg-state), name-alist)}\)
\(\land \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),}
  \text{bindings (top (ctrl-stk)), temp-stk)}\)
\(\land \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))}\)
\(\land \text{normal (mg-state)}\)
\(\land (\neg \text{negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state))))))})\)
\(\land (\text{idifference (cadddr (call-actuals (stmt))),}
  \text{untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state))))))} \not\equiv 0)\)
\(\to \ (\text{p-step (p-state (tag ('pc,}
  \text{'mg-array-element-assignment . 8)),}
  \text{push (p-frame (list (cons ('a,}
    \text{value (car (call-actuals (stmt)),}
    \text{bindings (top (ctrl-stk))))),}
  \text{cons ('i,}
    \text{value (cadr (call-actuals (stmt)),}
    \text{bindings (top (ctrl-stk))))),}
  \text{cons ('value,}
    \text{value (caddr (call-actuals (stmt)),}
    \text{bindings (top (ctrl-stk))))),}
  \text{cons ('array-size,}
    \text{tag ('int,}
    \text{cadddr (call-actuals (stmt))))),}
  \text{cons ('temp-i,}
    \text{mg-to-p-simple-literal (cadddr (assoc (cadr (call-actuals (stmt))))},
  \text{code2))}))\)
tag ('pc,
   cons (subr, length (code (cinfo))
        + 5)),

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-array-element-assignment . 9)),
push (p-frame (list (cons ('a,
   value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk)))))
   cons ('i,
   value (cadr (call-actuals (stmt))),
   bindings (top (ctrl-stk)))))
   cons ('value,
   value (caddr (call-actuals (stmt))),
   bindings (top (ctrl-stk)))))
   cons ('array-size,
   tag ('int,
   cadddr (call-actuals (stmt))))))
   cons ('temp-i,
   mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))),
   mg-alist (mg-state))))),
   tag ('pc,
   cons (subr, length (code (cinfo))
        + 5)),

ctrl-stk
push (value ('value,
   list (cons ('a,
   value (car (call-actuals (stmt))),
   bindings (top (ctrl-stk)))))
   cons ('i,
   value (cadr (call-actuals (stmt))),
   bindings (top (ctrl-stk)))))
   cons ('value,
Theorem: mg-array-element-assignment-step-15-no-error

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

→ (p-step (p-state (tag ('pc,
  (mg-array-element-assignment . 9)),
  push (p-frame (list (cons ('a,
    value (car (call-actuals (stmt)),
      bindings (top (ctrl-stk))))),
    cons ('i,
      value (cadr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('value,
      value (caddr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('array-size,
      tag ('int,
        caddr (call-actuals (stmt)))),
    cons ('temp-i,
      mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
        mg-alist (mg-state))))))),
  tag ('pc,
    cons (subr, length (code (cinfo))
      + 5)),
  ctrl-stk),
  push (value ('value,
    list (cons ('a,
      value (car (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('i,
      value (cadr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('value,
      value (caddr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('array-size,
      tag ('int,
        caddr (call-actuals (stmt)))),
    cons ('temp-i,
      mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
        mg-alist (mg-state))))),
    map-down-values (mg-alist (mg-state),
      bindings (top (ctrl-stk)),
      temp-stk)),
  translate-proc-list (proc-list),
  66)
Theorem: mg-array-element-assignment-steps-16-17-no-error
((n \not\equiv 0)
\land \neg \text{resources-inadequatep} (stmt, proc-list, run))
∧ (car (stmt) = 'predefined-proc-call-mg)
∧ (call-name (stmt) = 'mg-array-element-assignment)
∧ 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))))
∧ (idifference (caddr (call-actuals (stmt)),
  untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))) ≠ 0))
→ (p-step (p-step (p-state (tag ('pc,
  'mg-array-element-assignment
  . 10)),
  push (p-frame (list (cons ('a,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  cons ('i,
    value (cadr (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  cons ('value,
    value (caddr (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
  cons ('array-size,
    tag ('int,
      cadddr (call-actuals (stmt))))),
  cons ('temp-i,
    mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
      mg-alist (mg-state))))),
  tag ('pc,
  cons (subr,
length (code (cinfo))
+ 5)),

\(ctrl-stk\),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))), mg-alist (mg-state))))),
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-array-element-assignment . 12)),
push (p-frame (list (cons (’a,
 value (car (call-actuals (stmt))), bindings (top (ctrl-stk)))))),
cons (’i,
 value (cadr (call-actuals (stmt))), bindings (top (ctrl-stk)))),
cons (’value,
 value (caddr (call-actuals (stmt))), bindings (top (ctrl-stk)))),
cons (’array-size,
 tag (’int,
 caddr (call-actuals (stmt)))),
cons (’temp-i,
 mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))), mg-alist (mg-state))))),
tag (’pc,
 cons (subr, length (code (cinfo))
+ 5)),

ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))), mg-alist (mg-state))))),
push (value (car (call-actuals (stmt))), bindings (top (ctrl-stk))),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))), mg-alist (mg-state))))),
map-down-values (mg-alist (mg-state),
Theorem: mg-array-element-assignment-step-18-no-error
\[ ((n \neq 0) \land \neg \text{resources-inadequatep } (\text{stmt}, \text{proc-list}, \text{list } (\text{length } (\text{temp-stk}), \text{p-ctrl-stk-size } (\text{ctrl-stk})))) \]
\[ \land (\text{car } (\text{stmt}) = '\text{predefined-proc-call-mg}) \]
\[ \land (\text{call-name } (\text{stmt}) = '\text{mg-array-element-assignment}) \]
\[ \land \text{ok-mg-statement } (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \]
\[ \land \text{ok-mg-def-plistp } (\text{proc-list}) \]
\[ \land \text{ok-mg-statep } (\text{mg-state}, \text{r-cond-list}) \]
\[ \land (\text{code } (\text{translate-def-body } (\text{assoc } (\text{subr}, \text{proc-list}), \text{proc-list}))) = \text{append } (\text{code } (\text{translate } (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \]
\[ \land \text{user-defined-procp } (\text{subr}, \text{proc-list}) \]
\[ \land \text{listp } (\text{ctrl-stk}) \]
\[ \land \text{all-cars-unique } (\text{mg-alist } (\text{mg-state})) \]
\[ \land \text{signatures-match } (\text{mg-alist } (\text{mg-state}), \text{name-alist}) \]
\[ \land \text{mg-vars-list-ok-in-p-state } (\text{mg-alist } (\text{mg-state}), \text{bindings } (\text{top } (\text{ctrl-stk})), \text{temp-stk}) \]
\[ \land \text{no-p-aliasing } (\text{bindings } (\text{top } (\text{ctrl-stk})), \text{mg-alist } (\text{mg-state})) \]
\[ \land \text{normal } (\text{mg-state}) \]
\[ \land (\neg \text{negativep } (\text{untag } (\text{mg-to-p-simple-literal } (\text{caddr } (\text{assoc } (\text{cadr } (\text{call-actuals } (\text{stmt})), \text{mg-alist } (\text{mg-state}))))))) \]
\[ \land (\text{idifference } (\text{caddr } (\text{call-actuals } (\text{stmt}))), \text{untag } (\text{mg-to-p-simple-literal } (\text{caddr } (\text{assoc } (\text{cadr } (\text{call-actuals } (\text{stmt})), \text{mg-alist } (\text{mg-state})))))) \neq 0) \]
\[ \rightarrow (\text{p-step } (\text{p-state } (\text{tag } ('\text{pc}, '\text{mg-array-element-assignment . 12})), \text{push } (\text{p-frame } (\text{list } (\text{cons } ('\text{a}, \text{value } (\text{car } (\text{call-actuals } (\text{stmt}))), \text{bindings } (\text{top } (\text{ctrl-stk}))))), \text{cons } ('\text{i}, \text{cons } ('\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-W}


\[
\begin{align*}
\text{value} & \text{(cadr (call-actuals (stmt)))}, \\
\text{bindings} & \text{(top (ctrl-stk)))}, \\
\text{cons} & \text{('value,} \\
\text{value} & \text{(caddr (call-actuals (stmt)))}, \\
\text{bindings} & \text{(top (ctrl-stk)))}, \\
\text{cons} & \text{('array-size,} \\
\text{tag} & \text{('int,} \\
\text{caddr} & \text{(call-actuals (stmt)))}, \\
\text{cons} & \text{('temp-i,} \\
\text{mg-to-p-simple-literal} & \text{(caddr (assoc (cadr (call-actuals (stmt))),} \\
\text{mg-alist (mg-state))))}, \\
\text{tag} & \text{('pc,} \\
\text{cons} & \text{(subr, length (code (cinfo))} \\
\text{} & \text{+ 5))),} \\
\text{ctrl-stk),} \\
\text{push} & \text{(mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),} \\
\text{mg-alist (mg-state))))),} \\
\text{push} & \text{(value (car (call-actuals (stmt)),} \\
\text{bindings (top (ctrl-stk)))}, \\
\text{push} & \text{(mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),} \\
\text{mg-alist (mg-state))))),} \\
\text{map-down-values} & \text{(mg-alist (mg-state),} \\
\text{bindings (top (ctrl-stk)),} \\
\text{temp-stk)))}, \\
\text{translate-proc-list} & \text{(proc-list),} \\
\text{list} & \text{(list ('c-c,} \\
\text{mg-cond-to-p-nat (cc (mg-state), t-cond-list)))}, \\
\text{MG-MAX-CTRL-STK-SIZE,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{MG-WORD-SIZE,} \\
\text{'run))} \\
\text{=} & \text{p-state (tag ('pc,} \\
\text{'(mg-array-element-assignment . 13)),} \\
\text{push (p-frame (list (cons ('a,} \\
\text{value (car (call-actuals (stmt)),} \\
\text{bindings (top (ctrl-stk))))),} \\
\text{cons ('i,} \\
\text{value (cadr (call-actuals (stmt)),} \\
\text{bindings (top (ctrl-stk))))}, \\
\text{cons ('value,} \\
\text{value (caddr (call-actuals (stmt)),} \\
\text{bindings (top (ctrl-stk))))}, \\
\text{cons ('array-size,} \\
\text{tag ('int,}
\end{align*}
\]
\begin{align*}
caddr \text{ (call-actuals } \text{ (stmt)})
\end{align*}
\begin{align*}
\text{cons } \text{ ('}\text{temp-i}', \\
\text{mg-to-p-simple-literal } \text{ (caddr } \text{ (assoc } \text{ (cdr (call-actuals } \text{ (stmt)}) \\
\text{ mg-alist } (\text{mg-state})))))
\end{align*}
\begin{align*}
tag (\text{'}\text{pc}', \\
\text{cons } \text{ (subr, length } \text{ (code } \text{ (cinfo)}) \\
+ \text{ 5)})),
\end{align*}
\begin{align*}
\text{ctrl-stk),} \\
\text{push} \text{ (tag } (\text{'}\text{nat}, \\
\text{untag } \text{ (mg-to-p-simple-literal } \text{ (caddr } \text{ (assoc } \text{ (cdr (call-actuals } \text{ (stmt)}) \\
\text{ mg-alist } \text{ (mg-state)))))))
\end{align*}
\begin{align*}
\text{push (value (car (call-actuals } \text{ (stmt)}) \\
\text{ bindings (top } \text{ (ctrl-stk)))}, \\
\text{push (mg-to-p-simple-literal } \text{ (caddr } \text{ (assoc } \text{ (cdr (call-actuals } \text{ (stmt)}) \\
\text{ mg-alist } \text{ (mg-state)))))))
\end{align*}
\begin{align*}
\text{map-down-values (mg-alist } \text{ (mg-state)} \\
\text{ bindings (top } \text{ (ctrl-stk)),} \\
\text{temp-stk])))
\end{align*}
\begin{align*}
\text{translate-proc-list (proc-list),} \\
\text{list (list } \text{ ('}\text{c-c}, \\
\text{ mg-cond-to-p-nat (cc } \text{ (mg-state), } t\text{-cond-list})),} \\
\text{MG-MAX-CTRL-STK-SIZE,} \\
\text{MG-MAX-TEMP-STK-SIZE,} \\
\text{MG-WORD-SIZE,} \\
\text{'}\text{run}))
\end{align*}

**THEOREM:** mg-array-element-assignment-step-19-no-error

\((n \not\equiv 0) \land (\neg \text{resources-inadequatp } (\text{stmt,}) \\
\text{proc-list,} \\
\text{list (length } \text{ (temp-stk),} \\
\text{p-ctrl-stk-size (ctrl-stk))))
\land (\text{car (stmt) } = \text{'}\text{predefined-proc-call-mg})
\land (\text{call-name (stmt) } = \text{'}\text{mg-array-element-assignment})
\land \text{ok-mg-statement (stmt, } r\text{-cond-list, name-alist, proc-list)}
\land \text{ok-mg-def-plistp (proc-list)}
\land \text{ok-mg-statep (mg-state, } r\text{-cond-list)}
\land (\text{code (translate-def-body } (\text{assoc } \text{ (subr, proc-list), proc-list}) \\
= \text{ append (code (translate (cinfo, } t\text{-cond-list, stmt, proc-list)),} \\
\text{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)}
\( \land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state})), \)
\( \land \text{bindings}(\text{top}(\text{ctrl-stk})), \)
\( \text{temp-stk} \)
\( \land \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state})) \)
\( \land \text{normal}(\text{mg-state}) \)
\( \land (\neg \text{negativep}(\text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))))) \)
\( \land (\text{idifference}(\text{cadddr}(\text{call-actuals}(\text{stmt})), \text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})))))) \not\equiv 0) \)
\( \rightarrow (\text{p-step}(\text{p-state}(\text{tag}('pc, '\text{mg-array-element-assignment} . 13)), \)
\( \text{push}(\text{p-frame}(\text{list}(\text{cons}('a, \)
\( \text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk}))))), \)
\( \text{cons}('i, \text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk}))))), \)
\( \text{cons}('\text{value}, \text{value}(\text{cadddr}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk}))))), \)
\( \text{cons}('\text{array-size}, \text{tag}('\text{int}, \text{cadddr}(\text{call-actuals}(\text{stmt}))))), \)
\( \text{cons}('\text{temp-i}, \text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))), \)
\( \text{tag}('pc, \text{cons}('subr, \text{length}(\text{code}(\text{cinfo})), \text{+} 5))), \)
\( \text{ctrl-stk})), \)
\( \text{push}(\text{tag}('\text{nat}, \text{untag}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})))))), \)
\( \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \text{bindings}(\text{top}(\text{ctrl-stk})))), \text{push}(\text{mg-to-p-simple-literal}(\text{caddr}(\text{assoc}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))), \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk})))), \)
\( \text{translate-proc-list}(\text{proc-list}), \text{list}(\text{list}('c-c, \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}), \text{t-cond-list})))) \)

73
\[ \text{Theorem: } \text{mg-array-element-assignment-index-lessp-temp-stk-length} \]
\[
\begin{align*}
((\text{car}(\text{stmt}) &= \text{'predefined-proc-call-mg}) \\
\land (\text{call-name}(\text{stmt}) &= \text{'mg-array-element-assignment}) \\
\land \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
\land \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list}) \\
\land \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \\
\quad \text{bindings}(\text{top}(\text{ctrl-stk})), \text{temp-stk}) \\
\land \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \\
\land (\neg \text{negativep}(\text{untag}(\text{caddr}(\text{assoc}(\text{cdr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))) \\
\land (\text{idifference}(\text{cadddr}(\text{call-actuals}(\text{stmt})), \\
\quad \text{untag}(\text{mg-to-p-simple.literal}(\text{caddr}(\text{assoc}(\text{cdr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state}))))) \neq 0) \\
\rightarrow ((\text{untag}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt}))), \text{bindings}(\text{top}(\text{ctrl-stk})))) \\
\quad + \text{untag}(\text{mg-to-p-simple.literal}(\text{caddr}(\text{assoc}(\text{cdr}(\text{call-actuals}(\text{stmt})), \\
\quad \text{mg-alist}(\text{mg-state}))))) \\
\quad < \text{length}(\text{temp-stk}) \\
\quad = \text{t})
\end{align*}
\]

**Theorem:** mg-array-element-assignment-step-20-no-error
\[
((n \neq 0) \\
\land (\neg \text{resources-inadequatep}(\text{stmt}, \\
\quad \text{proc-list}, \\
\quad \text{list}(\text{length}(\text{temp-stk}), \\
\quad \text{p-ctrl-stk-size}(\text{ctrl-stk})))) \\
\land (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg}) \\
\land (\text{call-name}(\text{stmt}) = \text{'mg-array-element-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 \quad \quad \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}), \\
\quad \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}) \\
\land (\neg \text{negativep}(\text{untag}(\text{mg-to-p-simple.literal}(\text{caddr}(\text{assoc}(\text{cdr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})))))))}
\( \text{difference } (\text{cadddr } (\text{call-actuals } (\text{stmt}))) \)
untag (\text{mg-to-p-simple-literal } (\text{cadddr } (\text{assoc } (\text{caddr } (\text{call-actuals } (\text{stmt}))) \text{mg-alist } (\text{mg-state})))))) \neq 0) \\
\rightarrow (\text{p-step } (\text{p-state } (\text{tag } (\text{pc}, \quad (\text{mg-array-element-assignment . 14})), \\
push (\text{p-frame } (\text{list } (\text{cons } (\text{a,} \\
\text{value } (\text{car } (\text{call-actuals } (\text{stmt}))), \\
\text{bindings } (\text{top } (\text{ctrl-stk}))))), \\
\text{cons } (\text{i,} \\
\text{value } (\text{cadddr } (\text{call-actuals } (\text{stmt}))), \\
\text{bindings } (\text{top } (\text{ctrl-stk}))))), \\
\text{cons } (\text{value,} \\
\text{value } (\text{cadddr } (\text{call-actuals } (\text{stmt}))), \\
\text{bindings } (\text{top } (\text{ctrl-stk}))))), \\
\text{cons } (\text{array-size,} \\
\text{tag } (\text{int,} \\
\text{cadddr } (\text{call-actuals } (\text{stmt})))), \\
\text{cons } (\text{temp-i,} \\
\text{mg-to-p-simple-literal } (\text{cadddr } (\text{assoc } (\text{caddr } (\text{call-actuals } (\text{stmt}))), \\
\text{mg-alist } (\text{mg-state}))))), \\
\text{tag } (\text{pc,} \\
\text{cons } (\text{subr,} \text{ length } (\text{code } (\text{cinfo})) \ + \ 5)), \\
\text{ctrl-stk}), \\
push (\text{tag } (\text{nat,} \\
\text{untag } (\text{value } (\text{car } (\text{call-actuals } (\text{stmt}))), \\
\text{bindings } (\text{top } (\text{ctrl-stk})))) + \ \text{untag } (\text{mg-to-p-simple-literal } (\text{cadddr } (\text{assoc } (\text{caddr } (\text{call-actuals } (\text{stmt}))), \\
\text{mg-alist } (\text{mg-state}))))), \\
push (\text{mg-to-p-simple-literal } (\text{cadddr } (\text{assoc } (\text{caddr } (\text{call-actuals } (\text{stmt}))), \\
\text{mg-alist } (\text{mg-state})))), \\
\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{mg-array-element-assignment . 15}), \\
push (\text{p-frame } (\text{list } (\text{cons } (\text{a,} \\
\text{76}))))
Theorem: mg-array-element-assignment-steps-21-22-no-error

\((n \not= 0)\)
\(\land\) (\(\neg\) resources-inadequatep \((stmt, proc-list, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))\)
\(\land\) (\(\text{car} (stmt) = \text{'predefined-proc-call-mg}\))
\(\land\) (\(\text{call-name} (stmt) = \text{'mg-array-element-assignment}\))
\(\land\) 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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))))
∧ (idifference (cadddr (call-actuals (stmt))),
  untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))) \neq 0))
→ (p-step (p-step (p-state (tag ('pc,
  'mg-array-element-assignment
   , 15)),
    push (p-frame (list (cons ('a,
        value (car (call-actuals (stmt)),
          bindings (top (ctrl-stk))))),
            cons ('i,
              value (cadr (call-actuals (stmt))),
                bindings (top (ctrl-stk))))),
            cons ('value,
              value (caddr (call-actuals (stmt))),
                bindings (top (ctrl-stk))))),
              cons (array-size,
                tag ('int,
                  cadddr (call-actuals (stmt))))),
              cons (temp-i,
                mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                  mg-alist (mg-state))))))
    tag ('pc,
      cons (subr,
        length (code (cinfo))
          + 5)),
    ctrl-stk),
  rput (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
        mg-alist (mg-state))))),
  78)
Theorem: mg-array-element-assignment-push-cc
((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)} = \text{predefined-proc-call-mg})
\land (\text{call-name (stmt)} = \text{mg-array-element-assignment})
\land \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)}
\land \text{ok-mg-def-plistp (proc-list)}

79
\[\begin{align*}
&\land \text{ok-mg-statep} (\text{mg-state}, \text{r-cond-list}) \\
&\land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list}))) = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \\
&\land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \\
&\land \text{listp} (\text{ctrl-stk}) \\
&\land \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \\
&\land \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{name-alist}) \\
&\land \text{normal} (\text{mg-state}) \\
\rightarrow (\text{p-step} (\text{p-state} (\text{tag} ('\text{pc}, \text{cons} (\text{subr}, \text{length} (\text{code} (\text{cinfo}))) + 5)), \text{ctrl-stk}, \text{temp-stk}, \text{translate-proc-list} (\text{proc-list}), \text{list} (\text{list} ('\text{c-c}, \text{cc-value})), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}, '\text{run})) \\
&\land (\text{p-state} (\text{tag} ('\text{pc}, \text{cons} (\text{subr}, \text{length} (\text{code} (\text{cinfo}))) + 6)), \text{ctrl-stk}, \text{push} (\text{cc-value}, \text{temp-stk}), \text{translate-proc-list} (\text{proc-list}), \text{list} (\text{list} ('\text{c-c}, \text{cc-value})), \text{MG-MAX-CTRL-STK-SIZE}, \text{MG-MAX-TEMP-STK-SIZE}, \text{MG-WORD-SIZE}, '\text{run}))
\end{align*}\]

**Theorem:** mg-array-element-assignment-sub1-cc

\[\left( (n \not\equiv 0) \land \neg \text{resources-inadequatep} (\text{stmt}, \text{proc-list}, \text{list} (\text{length} (\text{temp-stk}), \text{p-ctrl-stk-size} (\text{ctrl-stk}))) \right) \land \left( \text{car} (\text{stmt}) = '\text{predefined-proc-call-mg} \right) \land \left( \text{call-name} (\text{stmt}) = '\text{mg-array-element-assignment} \right) \land \text{ok-mg-statement} (\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \land \text{ok-mg-def-plistp} (\text{proc-list}) \land \text{ok-mg-statemtp} (\text{mg-state}, \text{r-cond-list}) \land (\text{code} (\text{translate-def-body} (\text{assoc} (\text{subr}, \text{proc-list}), \text{proc-list}))) = \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \text{code2})) \land \text{user-defined-procp} (\text{subr}, \text{proc-list}) \land \text{listp} (\text{ctrl-stk})\]
∧ all-cars-unique (mg-alist (mg-state))
∧ signatures-match (mg-alist (mg-state), name-alist)
∧ normal (mg-state)
∧ (cc-value ∈ list (\(\langle\text{nat} 1\rangle, \langle\text{nat} 2\rangle\)))
→ (p-step (p-state (tag (\('\text{pc}', cons (subr, length (code (cinfo)) + 6))),
  ctrl-stk,
  push (cc-value, temp-stk),
  translate-proc-list (proc-list),
  list (list (\('\text{c-c}', cc-value)),
    MG-MAX-CTRL-STK-SIZE,
    MG-MAX-TEMP-STK-SIZE,
    MG-WORD-SIZE,
    \('\text{run}'))
  = p-state (tag (\('\text{pc}', cons (subr, length (code (cinfo)) + 7))),
    ctrl-stk,
    push (tag (\('\text{nat}', untag (cc-value) - 1), temp-stk),
    translate-proc-list (proc-list),
    list (list (\('\text{c-c}', cc-value)),
      MG-MAX-CTRL-STK-SIZE,
      MG-MAX-TEMP-STK-SIZE,
      MG-WORD-SIZE,
      \('\text{run}'))
  
  \[\text{Theorem: mg-array-element-assignment-last-step-error-case}~\]
  ((n \not\equiv 0)
  ∧ (\neg \text{resources-inadequatep (stmt,}
      proc-list,
      list (length (temp-stk),
      p-ctl-stk-size (ctrl-stk)))))
∧ (car (stmt) = \('\text{predefined-proc-call-mg}~\)
∧ (call-name (stmt) = \('\text{mg-array-element-assignment}~\)
∧ 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-proc (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))
81
\[\begin{align*}
\land & \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
\land & \text{normal (mg-state)} \\
\land & \text{(negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state)))))))} \\
\lor & \text{(idifference (caddr (call-actuals (stmt))), untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)), mg-alist (mg-state)))))) \equiv 0)} \\
\rightarrow & \text{(p-step (p-state (tag (pc, cons (subr, length (code (cinfo)) + 7))), ctrl-stk, push (tag (nat, untag (nat 1)) - 1), map-down-values (mg-alist (mg-state), bindings (top (ctrl-stk)), temp-stk)), translate-proc-list (proc-list), '((c-c (nat 1)))}, \\
& \text{MG-MAX-CTRL-STK-SIZE,} \\
& \text{MG-MAX-TEMP-STK-SIZE,} \\
& \text{MG-WORD-SIZE,} \\
& \text{'run)} \\
= & \text{p-state (tag (pc, cons (subr, if normal (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk)))) then length (code (translate (cinfo, t-cond-list, stmt, proc-list))) else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n), list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list))), append (code (translate (cinfo, t-cond-list, stmt))}, 
\end{align*}\]
Theorem: put-preserves-ok-mg-array-value
\((ok-mg-array-value (z \; type)) \land \text{simple-typed-literalp} (x, \text{array-elemtype} (type)) \land (i < \text{length} (z))\)  
\(\rightarrow\) ok-mg-array-value (put \((x, i, z), type\))

Theorem: arrays-have-ok-values
\((\text{mg-alistp} (\text{mg-alist}) \land \text{array-identifierp} (a, \text{mg-alist}))\)  
\(\rightarrow\) ok-mg-array-value (caddr (assoc \((a, \text{mg-alist})\)), cadr (assoc \((a, \text{mg-alist})\)))

Definition:
put-deposit-array-value-induction-hint \((lst, nat, temp-stk, index)\)  
= if \(index \simeq 0\) then \(t\) 
else put-deposit-array-value-induction-hint \((\text{cdr} \; lst), \text{add1-nat} \; (nat), \text{deposit-temp} \; (\text{mg-to-p-simple-literal} \; (\text{car} \; lst)), \text{nat}, \text{temp-stk}, \text{index} - 1\) endif
;;; I think the splits over whether \(\text{nlistp (cdr lst)}\) are not necessary

**Theorem: put-deposit-array-value-rewrite**

\[
\begin{align*}
& (\text{index} < \text{length (lst)}) \\
\land & \ ((\text{untag (nat)} + \text{length (lst)} - 1) < \text{length (temp-stk)}) \\
\land & \ (\text{untag (nat)} \in \mathbb{N}) \\
\rightarrow & \ (\text{deposit-array-value (put (value, index, lst), nat, temp-stk)}) \\
& = \ rput (\text{mg-to-p-simple-literal (value)}, \\
& \text{untag (nat)} + \text{index}, \\
& \text{deposit-array-value (lst, nat, temp-stk)})
\end{align*}
\]

**Event:** Disable put-deposit-array-value-rewrite.

**Theorem: mg-array-element-assignment-deposit-array-value-rewrite**

\[
\begin{align*}
& (\text{all-cars-unique (mg-vars)} \\
\land & \ \text{mg-alistp (mg-vars)} \\
\land & \ \text{mg-vars-list-ok-in-p-state (mg-vars, bindings, temp-stk)} \\
\land & \ \text{no-p-aliasing (bindings, mg-vars)} \\
\land & \ (\text{index} < \text{array-length (cadr (assoc (a, mg-vars))})) \\
\land & \ \text{array-identifierp (a, mg-vars)} \\
\rightarrow & \ (\text{deposit-array-value (put (value, index, caddr (assoc (a, mg-vars))),} \\
& \text{cdr (assoc (a, bindings)),} \\
& \text{map-down-values (mg-vars, bindings, temp-stk)}) \\
& = \ rput (\text{mg-to-p-simple-literal (value)}, \\
& \text{untag (cdr (assoc (a, bindings))) + index,} \\
& \text{map-down-values (mg-vars, bindings, temp-stk)})
\end{align*}
\]

**Event:** Disable mg-array-element-assignment-deposit-array-value-rewrite.

**Theorem: mg-array-element-assignment-step-25-no-error**

\[
\begin{align*}
& (n \not\equiv 0) \\
\land & \ (\neg \text{resources-inadequatep (stmt,} \\
& \text{proc-list,} \\
& \text{list (length (temp-stk),} \\
& \text{p-ctrl-stk-size (ctrl-stk)))) \\
\land & \ (\text{car (stmt) = 'predefined-proc-call-mg}) \\
\land & \ (\text{call-name (stmt) = 'mg-array-element-assignment}) \\
\land & \ \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
\land & \ \text{ok-mg-def-plistp (proc-list)} \\
\land & \ \text{ok-mg-statep (mg-state, r-cond-list)} \\
\land & \ (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))} \\
& = \ \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),} \\
& \text{code2)})
\end{align*}
\]

84
∧ user-defined-proc (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)
∧ (¬ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))))
∧ (idifference (cadddr (call-actuals (stmt)),
  untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
  mg-alist (mg-state)))))) ≠ 0)
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 7),
  ctrl-stk,
  push (tag ('nat,
    untag (mg-cond-to-p-nat (cc (mg-state),
      t-cond-list)) − 1),
    rput (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
      mg-alist (mg-state))))),
    untag (value (car (call-actuals (stmt)),
      bindings (top (ctrl-stk)))))
  + untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
      mg-alist (mg-state))))),
  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),
Theorem: mg-array-element-assignment-exact-time-lemma

\((\forall n \neq 0)\)
\(\land (\neg \text{resources-inadequatep } stmt, \)

\([\neg \text{resources-inadequatep } stmt, \)
\[
\text{proc-list,} \\
\text{list (length (temp-stk),} \\
\text{p-ctrl-stk-size (ctrl-stk)))} \\
\wedge (\text{car (stmt) = ’predefined-proc-call-mg}) \\
\wedge (\text{call-name (stmt) = ’mg-array-element-assignment}) \\
\wedge \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list}) \\
\wedge \text{ok-mg-def-plistp (proc-list)} \\
\wedge \text{ok-mg-statep (mg-state, r-cond-list)} \\
\wedge (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))} \\
\quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),} \\
\quad \text{code2)}) \\
\wedge \text{user-defined-procp (subr, proc-list)} \\
\wedge \text{listp (ctrl-stk)} \\
\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),} \\
\text{bindings (top (ctrl-stk)),} \\
\text{temp-stk)} \\
\wedge \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
\wedge \text{normal (mg-state))} \\
\rightarrow (p (\text{map-down (mg-state,} \\
\text{proc-list,} \\
\text{ctrl-stk,} \\
\text{temp-stk,} \\
\text{tag (“pc, cons (subr, length (code (cinfo))))),} \\
\text{t-cond-list),} \\
\text{clock (stmt, proc-list, mg-state, n)}) \\
\quad = \text{p-state (tag (“pc,} \\
\quad \text{cons (subr,} \\
\quad \text{if normal (mg-meaning-r (stmt,} \\
\quad \text{proc-list,} \\
\quad \text{mg-state,} \\
\quad \text{n,} \\
\quad \text{list (length (temp-stk),} \\
\quad \text{p-ctrl-stk-size (ctrl-stk))))} \\
\text{then length (code (translate (cinfo,} \\
\text{t-cond-list,} \\
\text{stmt,} \\
\text{proc-list)))} \\
\text{else find-label (fetch-label (cc (mg-meaning-r (stmt,} \\
\text{proc-list,} \\
\text{mg-state,} \\
\text{n,} \\
\text{list (length (temp-stk),} \\
\text{87})}
\text{Theorem: predefined-proc-call-exact-time-lemma}
\begin{align*}
& ((n \neq 0) \\
& \land \neg \text{resources-inadequatep} (stmt, proc-list,)
\end{align*}
\[\begin{align*}
\text{list} & (\text{length } (\text{temp-stk}), \\
& \quad \text{p-ctrl-stk-size } (\text{ctrl-stk}))) \\
\land & \ (\text{car } (\text{stmt}) = \text{ predefined-proc-call-mg }) \\
\land & \ \text{ok-mg-statement} (\text{stmt}, \text{ r-cond-list}, \text{ name-alist}, \text{ proc-list}) \\
\land & \ \text{ok-mg-def-plistp} (\text{proc-list}) \\
\land & \ \text{ok-translation-parameters} (\text{cinfo}, \text{ t-cond-list}, \text{ stmt}, \text{ proc-list}, \text{ code2}) \\
\land & \ \text{ok-mg-statep} (\text{mg-state}, \text{ r-cond-list}) \\
\land & \ \text{cond-subsetp} (\text{r-cond-list}, \text{ t-cond-list}) \\
\land & \ (\text{code} (\text{translate-def-body} \ (\text{assoc} (\text{subr}, \text{ proc-list}), \text{ proc-list}))) \\
& \quad = \ \text{append} (\text{code} (\text{translate} (\text{cinfo}, \text{ t-cond-list}, \text{ stmt}, \text{ proc-list})), \\
& \quad \quad \quad \quad \text{code2})) \\
\land & \ \text{user-defined-procsp} (\text{subr}, \text{ proc-list}) \\
\land & \ \text{plistp} (\text{temp-stk}) \\
\land & \ \text{lisp} (\text{ctrl-stk}) \\
\land & \ \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (\text{mg-state}), \\
& \quad \text{bindings} (\text{top } (\text{ctrl-stk})), \\
& \quad \text{temp-stk}) \\
\land & \ \text{no-p-aliasing} (\text{bindings} (\text{top } (\text{ctrl-stk})), \text{ mg-alist} (\text{mg-state})) \\
\land & \ \text{signatures-match} (\text{mg-alist} (\text{mg-state}), \text{ name-alist}) \\
\land & \ \text{normal} (\text{mg-state}) \\
\land & \ \text{all-cars-unique} (\text{mg-alist} (\text{mg-state})) \\
\land & \ (\neg \ \text{resource-errorp} (\text{mg-meaning-r} (\text{stmt}, \\
& \quad \text{proc-list}, \\
& \quad \text{mg-state}, \\
& \quad n, \\
& \quad \text{list} (\text{length } (\text{temp-stk}), \\
& \quad \quad \quad \text{p-ctrl-stk-size } (\text{ctrl-stk})))))) \\
\rightarrow & \ (\text{p} (\text{map-down} (\text{mg-state}, \\
& \quad \text{proc-list}, \\
& \quad \text{ctrl-stk}, \\
& \quad \text{temp-stk}, \\
& \quad \text{tag } (\text{pc}, \text{cons} (\text{subr}, \text{length } (\text{code } (\text{cinfo})))), \\
& \quad \text{t-cond-list}), \\
& \quad \text{clock } (\text{stmt}, \text{ proc-list}, \text{ mg-state}, n))) \\
= & \ \text{p-state} (\text{tag } (\text{pc}, \\
& \quad \text{cons} (\text{subr}, \\
& \quad \quad \text{if normal} (\text{mg-meaning-r} (\text{stmt}, \\
& \quad \quad \quad \text{proc-list}, \\
& \quad \quad \quad \text{mg-state}, \\
& \quad \quad \quad n, \\
& \quad \quad \quad \text{list} (\text{length } (\text{temp-stk}), \\
& \quad \quad \quad \quad \text{p-ctrl-stk-size } (\text{ctrl-stk})))))) \\
& \quad \quad \quad \text{then} \ \text{length } (\text{code} (\text{translate} (\text{cinfo}, \\
& \quad \quad \quad \text{t-cond-list}, \\
& \quad \quad \quad \text{code2})))) \\
& \quad \quad \quad \text{if} \ \text{normal} (\text{mg-meaning-r} (\text{stmt}, \\
& \quad \quad \quad \text{proc-list}, \\
& \quad \quad \quad \text{mg-state}, \\
& \quad \quad \quad n, \\
& \quad \quad \quad \text{list} (\text{length } (\text{temp-stk}), \\
& \quad \quad \quad \quad \text{p-ctrl-stk-size } (\text{ctrl-stk})))) \\
& \quad \quad \quad \text{then} \ \text{length } (\text{code} (\text{translate} (\text{cinfo}, \\
& \quad \quad \quad \text{t-cond-list}, \\
& \quad \quad \quad \text{code2}))))
\end{align*}\]
else find-label (fetch-label (cc (mg-meaning-r (stmt, proc-list, mg-state, n, list (length (temp-stk), p-ctrl-stk-size (ctrl-stk))))), label-alist (translate (cinfo, t-cond-list, stmt, proc-list))),
append (code (translate (cinfo, t-cond-list, stmt, proc-list))),

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

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

t-cond-list)),

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

EVENT: Make the library "c-predefined4".
Index

add1-nat, 83
all-cars-unique, 4–6, 8, 10, 12, 14,
16, 18, 20, 22, 25, 27, 30,
32, 35, 37, 38, 40, 43, 46,
47, 49, 51, 52, 55, 57, 59,
61, 63, 65, 68, 70, 72, 75,
78, 80, 81, 84, 85, 87, 89
array-elemtype, 2, 45, 83
array-identifierp, 1, 2, 45, 83, 84
array-index-small-naturalp, 2
array-length, 1–3, 45, 84
arrays-have-non-zerop-lengths, 1
arrays-have-ok-values, 83
bindings, 4–36, 38–44, 46–79, 81–
83, 85–90
boolean-identifierp, 1
call-actuals, 2–35, 39, 41, 45–79, 82,
85
call-name, 2, 3, 5, 6, 8, 10, 12, 14,
16, 18, 20, 22, 25, 27, 30,
32, 34, 36–38, 40, 43, 45–
47, 49, 51, 52, 54, 57, 59,
61, 63, 65, 68, 70, 72, 75,
77, 79–81, 84, 87
cc, 4–9, 11–17, 19–24, 26, 28, 29,
31–34, 36, 39–42, 44, 47,
48, 50, 52–54, 56, 58, 60,
62, 64, 65, 67, 69–74, 76,
77, 79, 82, 83, 85, 86, 88, 90
character-identifierp, 1
clock, 43, 87, 89
code, 3–44, 46–90
cond-subsetp, 89
definedp, 3, 45
deposit-array-value, 84
deposit-temp, 83
fetch-label, 39, 42, 44, 82, 86, 88, 90
find-label, 39, 42, 44, 83, 86, 88, 90
idifference, 1, 2, 14, 15, 17–20, 23–
25, 27, 28, 30, 32, 35, 39,
41, 56–59, 61–63, 66, 68,
70, 73, 75, 76, 78, 82, 85
idifference-lessp, 2
idifference-lessp2, 1
int-identifierp, 1, 2, 45
integerp, 1, 24
label-alist, 39, 42, 44, 82, 86, 88, 90
length, 2–44, 46–90
lessp-plus-transitive, 1
limits-for-small-integerp, 2
map-down, 4, 43, 46, 87, 89
map-down-values, 4–9, 11–17, 19–
24, 26, 28, 29, 31–36, 39–
42, 44, 47–50, 52–56, 58,
60, 62, 64–67, 69–74, 76,
77, 79, 82–86, 88, 90
maxint, 2
mg-alist, 2–83, 85–90
mg-alistp, 1, 2, 83, 84
mg-array-element-assignment-arg
3-simple, 45
4-small-integerp, 45
s-definedp, 45
s-have-simple-mg-type-refps, 44
mg-array-element-assignment-dep
osit-array-value-rewrite, 84
mg-array-element-assignment-exa
ct-time-lemma, 86
mg-array-element-assignment-inde
x-lessp-temp-stk-length, 75
mg-array-element-assignment-last
-step-error-case, 81
mg-array-element-assignment-pus
h-cc, 79
mg-array-element-assignment-step