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

THEOREM: arrays-have-non-zerop-lengths
(array-identifierp (*x*, *alist*) ∧ mg-alistp (*alist*))
→ (array-length (cadr (assoc (*x*, *alist*))) ≠ 0)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                                                                              ;;
;;                        EXACT-TIME LEMMA MG-INDEX-ARRAY                        ;;
;;                                                                              ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

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

THEOREM: lessp-plus-transitive
(((*x* + (*w* − 1)) < *z*) ∧ (*y* < *w*)) → (((*x* + *y*) < *z*) = **t**)

EVENT: Disable lessp-plus-transitive.

THEOREM: idifference-lessp2
((*x* ∈ **N**) ∧ (*x* < *y*))
→ (((idifference (*y*, *x*) = 0) = **f**) ∧ (idifference (*y*, *x*) ∈ **N**))

THEOREM: zerop-integerp-trichotomy
(integerp (*x*) ∧ (*x* ∉ **N**)) → negativep (*x*)

EVENT: Disable zerop-integerp-trichotomy.

THEOREM: p-object-type-int
(p-objectp-type ('int, tag ('int, *x*), *state*)
= small-integerp (*x*, p-word-size (*state*)))
∧ (p-objectp-type ('int, list ('int, *x*), *state*)
= small-integerp (*x*, p-word-size (*state*)))

THEOREM: simple-identifierp-options
(int-identifierp (*x*, *alist*)
∨ boolean-identifierp (*x*, *alist*)
∨ character-identifierp (*x*, *alist*))
→ simple-identifierp (*x*, *alist*)

THEOREM: small-integerp-difference
(small-integerp (*x*, *n*)

$\wedge$ small-integerp (y, n)
$\wedge$ ($x \not\simeq$ 0)
$\wedge$ ($\neg$ negativep (y)))
$\rightarrow$ small-integerp (idifference (x, y), n)

THEOREM: limits-for-small-integerp
((*x* < MAXINT) $\wedge$ ($x \not\simeq$ 0)) $\rightarrow$ small-integerp (x, 32)

THEOREM: simple-typed-identifier-simple-identifierp
simple-typed-identifierp (x, *type*, *alist*) $\rightarrow$ simple-identifierp (x, *alist*)

THEOREM: mg-var-ok-array-index-ok3
(mg-vars-list-ok-in-p-state (*mg-vars*, *bindings*, *temp-stk*)
$\wedge$ mg-alistp (*mg-vars*)
$\wedge$ array-identifierp (x, *mg-vars*))
$\rightarrow$ (((untag (value (x, *bindings*))
+ (array-length (cadr (assoc (x, *mg-vars*))) − 1))
< length (*temp-stk*))
= **t**)

THEOREM: idifference-lessp
(($\neg$ negativep (y)) $\wedge$ (idifference (x, y) $\not\simeq$ 0)) $\rightarrow$ (($y < x$) = **t**)

THEOREM: nat-p-objectp-reduction
p-objectp-type ('nat, tag ('nat, x), *state*)
= small-naturalp (x, p-word-size (*state*))

THEOREM: array-index-small-naturalp
((*temp-stk-size* < MG-MAX-TEMP-STK-SIZE)
$\wedge$ ((a + (*array-size* − 1)) < *temp-stk-size*)
$\wedge$ (*index* < *array-size*))
$\rightarrow$ small-naturalp (a + *index*, 32)

THEOREM: mg-index-array-args-have-simple-mg-type-refps
((car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-index-array)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ (simple-typed-identifierp (car (call-actuals (*stmt*)),
array-elemtype (cadr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))),
mg-alist (*mg-state*))
$\wedge$ array-identifierp (cadr (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ int-identifierp (caddr (call-actuals (*stmt*)), mg-alist (*mg-state*))

$\wedge$ (cadddr (call-actuals (*stmt*))
= array-length (cadr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*))))))

THEOREM: mg-index-array-args-definedp
((car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-index-array)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ (definedp (car (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ definedp (cadr (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ definedp (caddr (call-actuals (*stmt*)), mg-alist (*mg-state*)))

THEOREM: mg-index-array-arg4-small-integerp
((car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-index-array)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ small-integerp (cadddr (call-actuals (*stmt*)), 32)

THEOREM: not-zerop-mg-index-array-arg4
((car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-index-array)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ ((cadddr (call-actuals (*stmt*)) $\in$ **N**)
$\wedge$ (cadddr (call-actuals (*stmt*)) $\neq$ 0))

THEOREM: mg-index-array-steps-1-4
((*n* $\not\simeq$ 0)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-index-array)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))

$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*))
$\rightarrow$ (p-step (p-step (p-step (p-step (map-down (*mg-state*,
proc-list,
ctrl-stk,
temp-stk,
tag ('pc,
cons (*subr*,
length (code (*cinfo*)))),
t-cond-list)))))
= p-state (tag ('pc, cons (*subr*, length (code (*cinfo*)) + 4)),
ctrl-stk,
push (tag ('int, cadddr (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))

THEOREM: mg-index-array-step-5
((*n* $\not\simeq$ 0)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = 'predefined-proc-call-mg)

$\wedge$ (call-name (*stmt*) = `'mg-index-array`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*))
$\rightarrow$ (p-step (p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + 4)),
ctrl-stk,
push (tag (`'int`, cadddr (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*)),

bindings (top (*ctrl-stk*)))),
cons (cons (`'array-size`,
tag (`'int`,
cadddr (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`))

THEOREM: mg-index-array-steps-6-8
((*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*))
→ (p-step (p-step (p-step (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)),
                                                                              bindings (top (ctrl-stk)))),
                                                                  cons (cons ('array-size,
                                                                              tag ('int,
                                                                                   cadddr (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,
                                             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 (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))

THEOREM: mg-index-array-steps-9-12-neg-index
((*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)))))))
→   (p-step (p-step (p-step (p-step (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,
                                                                                cadddr (call-actuals (stmt)))),
                                                                    cons ('temp-i,
                                                                           mg-to-p-simple-literal (caddr (assoc (cadd
                                                                                                                mg-a
                                                                                  tag ('pc,
                                                                                       cons (subr,
                                                                                             length (code (cinfo))
                                                                                             +   5))),
                                                                    ctrl-stk),
                                                push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stm
                                                                                                  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,
'run))

THEOREM: mg-index-array-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-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*))))))))
→ (p-step (p-step (p-step (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,
cadddr (call-actuals (*stmt*)))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-a

```
                                                                          mg-alist (mg
                                          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-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,
                                            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 (mg-to-p-simple-literal (caddr (assoc (caddr (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*),
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`))

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

THEOREM: mg-index-array-step-12-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*))))))))
→ (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*)),

```
                                                    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),
                        push (mg-to-p-simple-literal (caddr (assoc (caddr (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),
                        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,
                                           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))

THEOREM: mg-index-array-step-13-index-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 (cadddr (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 . 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,
                                                    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 (stm
                                                                                    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 . 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,
                                             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))
```

THEOREM: mg-index-array-steps-14-16-index-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)
```

$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)
$\wedge$ ($\neg$ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))))
$\wedge$ (idifference (cadddr (call-actuals (*stmt*)),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))) $\simeq$ 0))
$\rightarrow$ (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,
cadddr (call-actuals (*stmt*)))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-a
mg-alist (*mg*
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`))

THEOREM: mg-index-array-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-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 (cadddr (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 . 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,
                                               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 (stm
                                                                                 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))),
```

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-14-15-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 (cadddr (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 . 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(
                                                                                        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))))),
```

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-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 (cadddr (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 . 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)))))),
                                   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))
    =   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,
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 ('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)

$$
\begin{array}{ll}
\wedge & (\text{call-name}\,(\textit{stmt}) = \texttt{'mg-index-array}) \\
\wedge & \text{ok-mg-statement}\,(\textit{stmt},\ \textit{r-cond-list},\ \textit{name-alist},\ \textit{proc-list}) \\
\wedge & \text{ok-mg-def-plistp}\,(\textit{proc-list}) \\
\wedge & \text{ok-mg-statep}\,(\textit{mg-state},\ \textit{r-cond-list}) \\
\wedge & (\text{code}\,(\text{translate-def-body}\,(\text{assoc}\,(\textit{subr},\ \textit{proc-list}),\ \textit{proc-list})) \\
 & \quad = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})), \\
 & \qquad\qquad \textit{code2})) \\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list}) \\
\wedge & \text{listp}\,(\textit{ctrl-stk}) \\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state})) \\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist}) \\
\wedge & \text{mg-vars-list-ok-in-p-state}\,(\text{mg-alist}\,(\textit{mg-state}), \\
 & \qquad\qquad\qquad\qquad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})), \\
 & \qquad\qquad\qquad\qquad \textit{temp-stk}) \\
\wedge & \text{no-p-aliasing}\,(\text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\ \text{mg-alist}\,(\textit{mg-state})) \\
\wedge & \text{normal}\,(\textit{mg-state}) \\
\wedge & (\neg\ \text{negativep}\,(\text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))))) \\
\wedge & (\text{idifference}\,(\text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad \text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))) \not\simeq \texttt{0})) \\
\rightarrow & (\text{p-step}\,(\text{p-state}\,(\text{tag}\,(\texttt{'pc},\ \texttt{'(mg-index-array . 11)}), \\
 & \qquad\qquad\qquad\qquad \text{push}\,(\text{p-frame}\,(\text{list}\,(\text{cons}\,(\texttt{'ans}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{value}\,(\text{car}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'a}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{value}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'i}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{value}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'array-size}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{tag}\,(\texttt{'int}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'temp-i}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{tag}\,(\texttt{'pc}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\textit{subr},\ \text{length}\,(\text{code}\,(\textit{cinfo})) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad + \quad \texttt{5}))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad \textit{ctrl-stk}), \\
 & \qquad\qquad\qquad\qquad\qquad \text{push}\,(\text{tag}\,(\texttt{'nat}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))),
\end{array}
$$

```
                    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,
                                                     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 ('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,
```

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

THEOREM: mg-index-array-index-lessp-temp-stk-length
((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-statep (*mg-state*, *r-cond-list*)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
                               bindings (top (*ctrl-stk*)),
                               temp-stk)
 ∧ signatures-match (mg-alist (*mg-state*), *name-alist*)
 ∧ (¬ negativep (untag (caddr (assoc (caddr (call-actuals (*stmt*)),
                                      mg-alist (*mg-state*))))))
 ∧ (idifference (cadddr (call-actuals (*stmt*)),
                 untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
                                                             mg-alist (*mg-state*))))))
    ≠ 0))
→ (((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*))))))
    < length (*temp-stk*))
   = **t**)

THEOREM: mg-index-array-step-18-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(cadddr(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 . 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,
                                                  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('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,
                       MG-MAX-TEMP-STK-SIZE,
                       MG-WORD-SIZE,
                       'run))
  =   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,
                                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 (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 ≠ 0)
 ∧  (¬ resources-inadequatep (stmt,
                              proc-list,
                              list (length (temp-stk),
                                    p-ctrl-stk-size (ctrl-stk))))
 ∧  (car (stmt) = 'predefined-proc-call-mg)
```

$$
\begin{array}{ll}
\wedge & (\text{call-name}\,(\textit{stmt}) = \texttt{'mg-index-array}) \\
\wedge & \text{ok-mg-statement}\,(\textit{stmt},\ \textit{r-cond-list},\ \textit{name-alist},\ \textit{proc-list}) \\
\wedge & \text{ok-mg-def-plistp}\,(\textit{proc-list}) \\
\wedge & \text{ok-mg-statep}\,(\textit{mg-state},\ \textit{r-cond-list}) \\
\wedge & (\text{code}\,(\text{translate-def-body}\,(\text{assoc}\,(\textit{subr},\ \textit{proc-list}),\ \textit{proc-list})) \\
 & \quad = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})), \\
 & \qquad\qquad \textit{code2})) \\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list}) \\
\wedge & \text{listp}\,(\textit{ctrl-stk}) \\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state})) \\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist}) \\
\wedge & \text{mg-vars-list-ok-in-p-state}\,(\text{mg-alist}\,(\textit{mg-state}), \\
 & \qquad\qquad\qquad\qquad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})), \\
 & \qquad\qquad\qquad\qquad \textit{temp-stk}) \\
\wedge & \text{no-p-aliasing}\,(\text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\ \text{mg-alist}\,(\textit{mg-state})) \\
\wedge & \text{normal}\,(\textit{mg-state}) \\
\wedge & (\neg\ \text{negativep}\,(\text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state}))))))) \\
\wedge & (\text{idifference}\,(\text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad \text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))) \not\simeq \texttt{0})) \\
\rightarrow & (\text{p-step}\,(\text{p-state}\,(\text{tag}\,(\texttt{'pc},\ \texttt{'(mg-index-array . 13)}), \\
 & \qquad\qquad\qquad\qquad \text{push}\,(\text{p-frame}\,(\text{list}\,(\text{cons}\,(\texttt{'ans}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{value}\,(\text{car}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\texttt{'a}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{value}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\texttt{'i}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{value}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\texttt{'array-size}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{tag}\,(\texttt{'int}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\texttt{'temp-i}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{tag}\,(\texttt{'pc}, \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\textit{subr},\ \text{length}\,(\text{code}\,(\textit{cinfo})) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + \quad \texttt{5}))), \\
 & \qquad\qquad\qquad\qquad\quad \textit{ctrl-stk}), \\
 & \qquad\qquad\qquad\qquad \text{push}\,(\text{rget}\,(\text{untag}\,(\text{value}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})), \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad + \quad \text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})),
\end{array}
$$

```
                                                                    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 . 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,
                                            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 (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))

THEOREM: mg-index-array-step-20-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 (cadddr (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 . 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,
                                                     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 (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 (stm
                                                                                      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,
                                             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),
                    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)
```

$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)
$\wedge$ ($\neg$ negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))))
$\wedge$ (idifference (cadddr (call-actuals (*stmt*)),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))) $\not\simeq$ 0))
$\rightarrow$ (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,
cadddr (call-actuals (*stmt*)))),
cons ('temp-i,
mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (
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 (*str*
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)))
= p-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-push-cc
((*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*))

$$
\begin{array}{ll}
 & = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})),\\
 & \qquad\qquad \textit{code2}))\\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list})\\
\wedge & \text{listp}\,(\textit{ctrl-stk})\\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state}))\\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist})\\
\wedge & \text{normal}\,(\textit{mg-state}))\\
\rightarrow & (\text{p-step}\,(\text{p-state}\,(\text{tag}\,(\texttt{'pc},\ \text{cons}\,(\textit{subr},\ \text{length}\,(\text{code}\,(\textit{cinfo}))\ +\ \texttt{5})),\\
 & \qquad\qquad\qquad \textit{ctrl-stk},\\
 & \qquad\qquad\qquad \textit{temp-stk},\\
 & \qquad\qquad\qquad \text{translate-proc-list}\,(\textit{proc-list}),\\
 & \qquad\qquad\qquad \text{list}\,(\text{list}\,(\texttt{'c-c},\ \textit{cc-value})),\\
 & \qquad\qquad\qquad \text{MG-MAX-CTRL-STK-SIZE},\\
 & \qquad\qquad\qquad \text{MG-MAX-TEMP-STK-SIZE},\\
 & \qquad\qquad\qquad \text{MG-WORD-SIZE},\\
 & \qquad\qquad\qquad \texttt{'run}))\\
 & = \quad \text{p-state}\,(\text{tag}\,(\texttt{'pc},\ \text{cons}\,(\textit{subr},\ \text{length}\,(\text{code}\,(\textit{cinfo}))\ +\ \texttt{6})),\\
 & \qquad\qquad \textit{ctrl-stk},\\
 & \qquad\qquad \text{push}\,(\textit{cc-value},\ \textit{temp-stk}),\\
 & \qquad\qquad \text{translate-proc-list}\,(\textit{proc-list}),\\
 & \qquad\qquad \text{list}\,(\text{list}\,(\texttt{'c-c},\ \textit{cc-value})),\\
 & \qquad\qquad \text{MG-MAX-CTRL-STK-SIZE},\\
 & \qquad\qquad \text{MG-MAX-TEMP-STK-SIZE},\\
 & \qquad\qquad \text{MG-WORD-SIZE},\\
 & \qquad\qquad \texttt{'run}))
\end{array}
$$

THEOREM: mg-index-array-sub1-cc

$$
\begin{array}{ll}
 & ((n \not\simeq \texttt{0})\\
\wedge & (\neg\ \text{resources-inadequatep}\,(\textit{stmt},\\
 & \qquad\qquad\qquad\qquad\qquad \textit{proc-list},\\
 & \qquad\qquad\qquad\qquad\qquad \text{list}\,(\text{length}\,(\textit{temp-stk}),\\
 & \qquad\qquad\qquad\qquad\qquad\quad \text{p-ctrl-stk-size}\,(\textit{ctrl-stk}))))\\
\wedge & (\text{car}\,(\textit{stmt}) = \texttt{'predefined-proc-call-mg})\\
\wedge & (\text{call-name}\,(\textit{stmt}) = \texttt{'mg-index-array})\\
\wedge & \text{ok-mg-statement}\,(\textit{stmt},\ \textit{r-cond-list},\ \textit{name-alist},\ \textit{proc-list})\\
\wedge & \text{ok-mg-def-plistp}\,(\textit{proc-list})\\
\wedge & \text{ok-mg-statep}\,(\textit{mg-state},\ \textit{r-cond-list})\\
\wedge & (\text{code}\,(\text{translate-def-body}\,(\text{assoc}\,(\textit{subr},\ \textit{proc-list}),\ \textit{proc-list}))\\
 & \quad = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})),\\
 & \qquad\qquad \textit{code2}))\\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list})\\
\wedge & \text{listp}\,(\textit{ctrl-stk})\\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state}))\\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist})
\end{array}
$$

$\wedge$ normal (*mg-state*)
$\wedge$ (*cc-value* $\in$ list (`'(nat 1)`, `'(nat 2)`)))
$\rightarrow$ (p-step (p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + `6`)),
ctrl-stk,
push (*cc-value*, *temp-stk*),
translate-proc-list (*proc-list*),
list (list (`'c-c`, *cc-value*)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))
= p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + `7`)),
ctrl-stk,
push (tag (`'nat`, untag (*cc-value*) $-$ 1), *temp-stk*),
translate-proc-list (*proc-list*),
list (list (`'c-c`, *cc-value*)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))

THEOREM: mg-index-array-last-step-error-case
((*n* $\not\simeq$ `0`)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-index-array`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)

```
∧   (negativep (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                              mg-alist (mg-state))))))
      ∨   (idifference (cadddr (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, 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))
      =   p-state (tag ('pc,
                        cons (subr,
                              if normal (mg-meaning-r (stmt,
                                                       proc-list,
                                                       mg-state,
                                                       n,
                                                       list (length (temp-stk),
                                                             p-ctrl-stk-size (ctrl-stk))))
                              then length (code (translate (cinfo,
                                                            t-cond-list,
                                                            stmt,
                                                            proc-list)))
                              else find-label (fetch-label (cc (mg-meaning-r (stmt,
                                                                              proc-list,
                                                                              mg-state,
                                                                              n,
                                                                              list (length (temp-stk),
                                                                                    p-ctrl-stk-size (ctrl-stk)))),
                                                           label-alist (translate (cinfo,
                                                                                   t-cond-list,
                                                                                   stmt,
                                                                                   proc-list))),
                                               append (code (translate (cinfo,
                                                                        t-cond-list,
                                                                        stmt,
                                                                        proc-list)),
                                                       code2)) endif)),
```

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

THEOREM: simple-typed-identifier-type-equivalence
simple-typed-identifierp (*b*, *type*, *alist*)
→ ((cadr (assoc (*b*, *alist*)) = *type*) = t)

THEOREM: mg-index-array-step-25-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 (cadddr (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, 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 (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))
   =  p-state (tag ('pc,
                    cons (subr,
                          if normal (mg-meaning-r (stmt,
                                                   proc-list,
                                                   mg-state,
                                                   n,
                                                   list (length (temp-stk),
                                                         p-ctrl-stk-size (ctrl-stk))))
                          then length (code (translate (cinfo,
```

t-cond-list,
stmt,
proc-list)))
else find-label (fetch-label (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
label-alist (translate (*cinfo*,
t-cond-list,
stmt,
proc-list))),
append (code (translate (*cinfo*,
t-cond-list,
stmt,
proc-list)),
code2)) **endif**)),
ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
bindings (top (*ctrl-stk*)),
temp-stk),
translate-proc-list (*proc-list*),
list (list (`'c-c`,
mg-cond-to-p-nat (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))

THEOREM: mg-index-array-exact-time-lemma
((*n* $\not\simeq$ `0`)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,

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

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

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                                                                              ;;
;;              EXACT-TIME-LEMMA MG-ARRAY-ELEMENT-ASSIGNMENT                    ;;
;;                                                                              ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; stmt looks like '(mg-predefined-proc-call mg-array-element-assignment (A i x size))
```

THEOREM: mg-array-element-assignment-args-have-simple-mg-type-refps
((car (*stmt*) = 'predefined-proc-call-mg)

$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ (array-identifierp (car (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ int-identifierp (cadr (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ simple-typed-identifierp (caddr (call-actuals (*stmt*)),
array-elemtype (cadr (assoc (car (call-actuals (*stmt*)),
mg-alist (*mg-state*)))),
mg-alist (*mg-state*))
$\wedge$ (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`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ simple-identifierp (caddr (call-actuals (*stmt*)), mg-alist (*mg-state*))

THEOREM: mg-array-element-assignment-args-definedp
((car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ (definedp (car (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ definedp (cadr (call-actuals (*stmt*)), mg-alist (*mg-state*))
$\wedge$ definedp (caddr (call-actuals (*stmt*)), mg-alist (*mg-state*)))

THEOREM: mg-array-element-assignment-arg4-small-integerp
((car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ small-integerp (cadddr (call-actuals (*stmt*)), `32`)

THEOREM: not-zerop-mg-array-element-assignment-arg4
((car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)

$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*))
$\rightarrow$ ((cadddr (call-actuals (*stmt*)) $\in$ **N**)
$\wedge$ (cadddr (call-actuals (*stmt*)) $\neq$ 0))

THEOREM: mg-array-element-assignment-steps-1-4
((*n* $\not\simeq$ 0)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = 'predefined-proc-call-mg)
$\wedge$ (call-name (*stmt*) = 'mg-array-element-assignment)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*))
$\rightarrow$ (p-step (p-step (p-step (p-step (map-down (*mg-state*,
proc-list,
ctrl-stk,
temp-stk,
tag ('pc,
cons (*subr*,
length (code (*cinfo*)))),
t-cond-list)))))
= p-state (tag ('pc, cons (*subr*, length (code (*cinfo*)) + 4)),
ctrl-stk,
push (tag ('int, cadddr (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`))

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-statep (*mg-state*, *r-cond-list*)
∧ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
∧ user-defined-procp (*subr*, *proc-list*)
∧ listp (*ctrl-stk*)
∧ all-cars-unique (mg-alist (*mg-state*))
∧ signatures-match (mg-alist (*mg-state*), *name-alist*)
∧ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
∧ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
∧ normal (*mg-state*))
→ (p-step (p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + `4`)),
ctrl-stk,
push (tag (`'int`, cadddr (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-array-element-assignment . 0)),
             push (p-frame (cons (cons ('a,
                                        value (car (call-actuals (stmt)),
                                               bindings (top (ctrl-stk)))),
                                 cons (cons ('i,
                                             value (cadr (call-actuals (stmt)),
                                                    bindings (top (ctrl-stk)))),
                                       cons (cons ('value,
                                                   value (caddr (call-actuals (stmt)),
                                                          bindings (top (ctrl-stk)))),
                                             cons (cons ('array-size,
                                                         tag ('int,
                                                              cadddr (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))
```

THEOREM: mg-array-element-assignment-steps-6-8

```
((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))
→   (p-step (p-step (p-step (p-state (tag ('pc,
                                           '(mg-array-element-assignment
                                             . 0)),
                                     push (p-frame (cons (cons ('a,
                                                                value (car (call-actuals (stmt)),
                                                                       bindings (top (ctrl-stk))))),
                                                         cons (cons ('i,
                                                                     value (cadr (call-actuals (stmt)),
                                                                            bindings (top (ctrl-stk))))),
                                                               cons (cons ('value,
                                                                           value (caddr (call-actuals (stmt)),
                                                                                  bindings (top (ctrl-stk))))),
                                                                     cons (cons ('array-size,
                                                                                 tag ('int,
                                                                                      cadddr (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-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,
                                                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)))),
                       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
$((n \not\simeq 0)$

```
∧  (¬ 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-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,
                                                                                cadddr (call-actuals (stmt)))),
                                                                     cons ('temp-i,
                                                                           mg-to-p-simple-literal (caddr (assoc (cadr
                                                                                                                 mg-a
                                                 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)))))
    =   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))
```

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,
                                                                         cadddr (call-actuals (stmt)))),
                                                              cons ('temp-i,
                                                                    mg-to-p-simple-literal (caddr (assoc (cadr (call-ac
                                                                                                         mg-alist (mg
                                                       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))))
```

```
=  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),
           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-12-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-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),
```

```
                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 . 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 (cadddr (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))
```

THEOREM: mg-array-element-assignment-step-13-index-error
$((n \not\simeq 0)$

```
∧  (¬ 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 (cadddr (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 . 16)),
             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),
             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-14-16-index-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-step (p-step (p-state (tag ('pc,
'(mg-array-element-assignment
. 16)),
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-ac
mg-alist (*mg*
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*))))

$\wedge$ (car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)
$\wedge$ ($\neg$ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))))
$\wedge$ (idifference (cadddr (call-actuals (*stmt*)),
untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))) $\not\simeq$ `0`))
$\rightarrow$ (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 (cadddr (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,
                                            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),
             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 ≠ 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 . 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,
                                                   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),
                            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,
```

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*)))))))),
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-15-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 . 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,
                                              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))))))),
                           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 . 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))
```

THEOREM: mg-array-element-assignment-steps-16-17-no-error

```
((n ≠ 0)
 ∧   (¬ resources-inadequatep (stmt,
                               proc-list,
```

$$
\begin{array}{ll}
 & \quad \text{list}\,(\text{length}\,(\textit{temp-stk}),\\
 & \qquad \text{p-ctrl-stk-size}\,(\textit{ctrl-stk}))))\\
\wedge & (\text{car}\,(\textit{stmt}) = \texttt{'predefined-proc-call-mg})\\
\wedge & (\text{call-name}\,(\textit{stmt}) = \texttt{'mg-array-element-assignment})\\
\wedge & \text{ok-mg-statement}\,(\textit{stmt},\ \textit{r-cond-list},\ \textit{name-alist},\ \textit{proc-list})\\
\wedge & \text{ok-mg-def-plistp}\,(\textit{proc-list})\\
\wedge & \text{ok-mg-statep}\,(\textit{mg-state},\ \textit{r-cond-list})\\
\wedge & (\text{code}\,(\text{translate-def-body}\,(\text{assoc}\,(\textit{subr},\ \textit{proc-list}),\ \textit{proc-list}))\\
 & \quad = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})),\\
 & \qquad\quad \textit{code2}))\\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list})\\
\wedge & \text{listp}\,(\textit{ctrl-stk})\\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state}))\\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist})\\
\wedge & \text{mg-vars-list-ok-in-p-state}\,(\text{mg-alist}\,(\textit{mg-state}),\\
 & \qquad\qquad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\\
 & \qquad\qquad \textit{temp-stk})\\
\wedge & \text{no-p-aliasing}\,(\text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\ \text{mg-alist}\,(\textit{mg-state}))\\
\wedge & \text{normal}\,(\textit{mg-state})\\
\wedge & (\neg\ \text{negativep}\,(\text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state}))))))\\
\wedge & (\text{idifference}\,(\text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\quad \text{untag}\,(\text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state})))))) \not\simeq \texttt{0}))\\
\rightarrow & (\text{p-step}\,(\text{p-step}\,(\text{p-state}\,(\text{tag}\,(\texttt{'pc},\\
 & \qquad\qquad\qquad\qquad\qquad \texttt{'(mg-array-element-assignment}\\
 & \qquad\qquad\qquad\qquad\qquad\quad \texttt{. 10)}),\\
 & \qquad\qquad\qquad\qquad \text{push}\,(\text{p-frame}\,(\text{list}\,(\text{cons}\,(\texttt{'a},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{value}\,(\text{car}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'i},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{value}\,(\text{cadr}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'value},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{value}\,(\text{caddr}\,(\text{call-actuals}\,(\textit{stmt})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})))),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'array-size},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{tag}\,(\texttt{'int},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{cadddr}\,(\text{call-actuals}\,(\textit{stmt})))),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\texttt{'temp-i},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{mg-to-p-simple-literal}\,(\text{caddr}\,(\text{assoc}\,(\text{cadr}\,(\text{call-actuals}\,(s\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{mg-alist}\,(\textit{mg-state}))\\
 & \qquad\qquad\qquad\qquad\quad \text{tag}\,(\texttt{'pc},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{cons}\,(\textit{subr},
\end{array}
$$

```
                                                  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,
                                          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 (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),
```

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-18-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 . 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,
                                         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 (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),
                                               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 . 13)),
            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 (`'nat`,
untag (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*),
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-19-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 . 13)),
                    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 ('nat,
                               untag (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),
                                                       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 . 14)),
             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 ('nat,
                        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)))))),
                   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))
```

THEOREM: mg-array-element-assignment-index-lessp-temp-stk-length

((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-statep (*mg-state*, *r-cond-list*)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
 bindings (top (*ctrl-stk*)),
 temp-stk)
 ∧ signatures-match (mg-alist (*mg-state*), *name-alist*)
 ∧ (¬ negativep (untag (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))
→ (((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*))))))
 < length (*temp-stk*))
 = **t**)

THEOREM: mg-array-element-assignment-step-20-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 . 14)),
                     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 ('nat,
                                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)))))),
                           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 . 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)))),
                      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))
```

THEOREM: mg-array-element-assignment-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-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-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 (s
                                                                                               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)))),
```

```
                                    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, length (code (cinfo)) + 5)),
               ctrl-stk,
               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))
```

THEOREM: mg-array-element-assignment-push-cc

```
((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)
```

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

THEOREM: mg-array-element-assignment-sub1-cc
((*n* $\not\simeq$ `0`)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)

$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ normal (*mg-state*)
$\wedge$ (*cc-value* $\in$ list (`'(nat 1)`, `'(nat 2)`)))
$\rightarrow$ (p-step (p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + 6)),
ctrl-stk,
push (*cc-value*, *temp-stk*),
translate-proc-list (*proc-list*),
list (list (`'c-c`, *cc-value*)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))
= p-state (tag (`'pc`, cons (*subr*, length (code (*cinfo*)) + 7)),
ctrl-stk,
push (tag (`'nat`, untag (*cc-value*) $-$ 1), *temp-stk*),
translate-proc-list (*proc-list*),
list (list (`'c-c`, *cc-value*)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))

THEOREM: mg-array-element-assignment-last-step-error-case
((*n* $\not\simeq$ `0`)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
$\wedge$ (car (*stmt*) = `'predefined-proc-call-mg`)
$\wedge$ (call-name (*stmt*) = `'mg-array-element-assignment`)
$\wedge$ ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
$\wedge$ ok-mg-def-plistp (*proc-list*)
$\wedge$ ok-mg-statep (*mg-state*, *r-cond-list*)
$\wedge$ (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)

$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)
$\wedge$ (negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*))))))
$\vee$ (idifference (cadddr (call-actuals (*stmt*)),
untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*))))))) $\simeq$ 0)))
$\rightarrow$ (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))
= p-state (tag ('pc,
cons (*subr*,
if normal (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*))))
then length (code (translate (*cinfo*,
t-cond-list,
stmt,
proc-list)))
else find-label (fetch-label (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
label-alist (translate (*cinfo*,
t-cond-list,
stmt,
proc-list))),
append (code (translate (*cinfo*,
t-cond-list,
stmt,

proc-list)),
code2)) **endif**)),
ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
bindings (top (*ctrl-stk*)),
temp-stk),
translate-proc-list (*proc-list*),
list (list ('`c-c`,
mg-cond-to-p-nat (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'`run`))

THEOREM: put-preserves-ok-mg-array-value
(ok-mg-array-value (*z*, *type*)
∧ simple-typed-literalp (*x*, array-elemtype (*type*))
∧ (*i* < length (*z*)))
→ ok-mg-array-value (put (*x*, *i*, *z*), *type*)

THEOREM: arrays-have-ok-values
(mg-alistp (*mg-alist*) ∧ array-identifierp (*a*, *mg-alist*))
→ ok-mg-array-value (caddr (assoc (*a*, *mg-alist*)), cadr (assoc (*a*, *mg-alist*)))

DEFINITION:
put-deposit-array-value-induction-hint (*lst*, *nat*, *temp-stk*, *index*)
= **if** *index* ≃ `0` **then t**
else put-deposit-array-value-induction-hint (cdr (*lst*),
add1-nat (*nat*),
deposit-temp (mg-to-p-simple-literal (car (*lst*)),
nat,
temp-stk),
index − 1) **endif**

```
;; I think the splits over whether (nlistp (cdr lst)) are not necessary
```

THEOREM: put-deposit-array-value-rewrite
((*index* < length (*lst*))
 ∧ ((untag (*nat*) + (length (*lst*) − 1)) < length (*temp-stk*))
 ∧ (untag (*nat*) ∈ **N**))
→ (deposit-array-value (put (*value*, *index*, *lst*), *nat*, *temp-stk*)
 = rput (mg-to-p-simple-literal (*value*),
 untag (*nat*) + *index*,
 deposit-array-value (*lst*, *nat*, *temp-stk*)))

EVENT: Disable put-deposit-array-value-rewrite.

THEOREM: mg-array-element-assignment-deposit-array-value-rewrite
(all-cars-unique (*mg-vars*)
 ∧ mg-alistp (*mg-vars*)
 ∧ mg-vars-list-ok-in-p-state (*mg-vars*, *bindings*, *temp-stk*)
 ∧ no-p-aliasing (*bindings*, *mg-vars*)
 ∧ (*index* < array-length (cadr (assoc (*a*, *mg-vars*))))
 ∧ array-identifierp (*a*, *mg-vars*))
→ (deposit-array-value (put (*value*, *index*, caddr (assoc (*a*, *mg-vars*))),
 cdr (assoc (*a*, *bindings*)),
 map-down-values (*mg-vars*, *bindings*, *temp-stk*))
 = rput (mg-to-p-simple-literal (*value*),
 untag (cdr (assoc (*a*, *bindings*))) + *index*,
 map-down-values (*mg-vars*, *bindings*, *temp-stk*)))

EVENT: Disable mg-array-element-assignment-deposit-array-value-rewrite.

THEOREM: mg-array-element-assignment-step-25-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))

$\wedge$ user-defined-procp (*subr*, *proc-list*)
$\wedge$ listp (*ctrl-stk*)
$\wedge$ all-cars-unique (mg-alist (*mg-state*))
$\wedge$ signatures-match (mg-alist (*mg-state*), *name-alist*)
$\wedge$ mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
$\wedge$ no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
$\wedge$ normal (*mg-state*)
$\wedge$ ($\neg$ negativep (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))))
$\wedge$ (idifference (cadddr (call-actuals (*stmt*)),
untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*)))))) $\not\simeq$ 0))
$\rightarrow$ (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*,

t-cond-list,
stmt,
proc-list)))
else find-label (fetch-label (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
label-alist (translate (*cinfo*,
t-cond-list,
stmt,
proc-list))),
append (code (translate (*cinfo*,
t-cond-list,
stmt,
proc-list)),
code2)) **endif**)),
ctrl-stk,
map-down-values (mg-alist (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
bindings (top (*ctrl-stk*)),
temp-stk),
translate-proc-list (*proc-list*),
list (list (`'c-c`,
mg-cond-to-p-nat (cc (mg-meaning-r (*stmt*,
proc-list,
mg-state,
n,
list (length (*temp-stk*),
p-ctrl-stk-size (*ctrl-stk*)))),
t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
`'run`))

THEOREM: mg-array-element-assignment-exact-time-lemma
((*n* $\not\simeq$ `0`)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,

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

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

;;;
;; ;;
;; EXACT-TIME-LEMMA for PREDEFINED's ;;
;; ;;
;;;

THEOREM: predefined-proc-call-exact-time-lemma
((*n* $\not\simeq$ 0)
$\wedge$ ($\neg$ resources-inadequatep (*stmt*,
proc-list,

$$
\begin{array}{ll}
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{list}\,(\text{length}\,(\textit{temp-stk}),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\quad \text{p-ctrl-stk-size}\,(\textit{ctrl-stk}))))\\
\wedge & (\text{car}\,(\textit{stmt}) = \texttt{'predefined-proc-call-mg})\\
\wedge & \text{ok-mg-statement}\,(\textit{stmt},\ \textit{r-cond-list},\ \textit{name-alist},\ \textit{proc-list})\\
\wedge & \text{ok-mg-def-plistp}\,(\textit{proc-list})\\
\wedge & \text{ok-translation-parameters}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list},\ \textit{code2})\\
\wedge & \text{ok-mg-statep}\,(\textit{mg-state},\ \textit{r-cond-list})\\
\wedge & \text{cond-subsetp}\,(\textit{r-cond-list},\ \textit{t-cond-list})\\
\wedge & (\text{code}\,(\text{translate-def-body}\,(\text{assoc}\,(\textit{subr},\ \textit{proc-list}),\ \textit{proc-list}))\\
 & \quad = \quad \text{append}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\ \textit{t-cond-list},\ \textit{stmt},\ \textit{proc-list})),\\
 & \qquad\qquad \textit{code2}))\\
\wedge & \text{user-defined-procp}\,(\textit{subr},\ \textit{proc-list})\\
\wedge & \text{plistp}\,(\textit{temp-stk})\\
\wedge & \text{listp}\,(\textit{ctrl-stk})\\
\wedge & \text{mg-vars-list-ok-in-p-state}\,(\text{mg-alist}\,(\textit{mg-state}),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad \textit{temp-stk})\\
\wedge & \text{no-p-aliasing}\,(\text{bindings}\,(\text{top}\,(\textit{ctrl-stk})),\ \text{mg-alist}\,(\textit{mg-state}))\\
\wedge & \text{signatures-match}\,(\text{mg-alist}\,(\textit{mg-state}),\ \textit{name-alist})\\
\wedge & \text{normal}\,(\textit{mg-state})\\
\wedge & \text{all-cars-unique}\,(\text{mg-alist}\,(\textit{mg-state}))\\
\wedge & (\neg\ \text{resource-errorp}\,(\text{mg-meaning-r}\,(\textit{stmt},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{proc-list},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{mg-state},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad n,\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{list}\,(\text{length}\,(\textit{temp-stk}),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{p-ctrl-stk-size}\,(\textit{ctrl-stk})))))\\
\rightarrow & (\text{p}\,(\text{map-down}\,(\textit{mg-state},\\
 & \qquad\qquad\qquad\qquad\qquad\quad \textit{proc-list},\\
 & \qquad\qquad\qquad\qquad\qquad\quad \textit{ctrl-stk},\\
 & \qquad\qquad\qquad\qquad\qquad\quad \textit{temp-stk},\\
 & \qquad\qquad\qquad\qquad\qquad\quad \text{tag}\,(\texttt{'pc},\ \text{cons}\,(\textit{subr},\ \text{length}\,(\text{code}\,(\textit{cinfo})))),\\
 & \qquad\qquad\qquad\qquad\qquad\quad \textit{t-cond-list}),\\
 & \qquad \text{clock}\,(\textit{stmt},\ \textit{proc-list},\ \textit{mg-state},\ n))\\
 & \quad = \quad \text{p-state}\,(\text{tag}\,(\texttt{'pc},\\
 & \qquad\qquad\qquad\qquad\qquad\quad \text{cons}\,(\textit{subr},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\quad \textbf{if}\ \text{normal}\,(\text{mg-meaning-r}\,(\textit{stmt},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \textit{proc-list},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \textit{mg-state},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad n,\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{list}\,(\text{length}\,(\textit{temp-stk}),\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{p-ctrl-stk-size}\,(\textit{ctrl-stk}))))\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\quad \textbf{then}\ \text{length}\,(\text{code}\,(\text{translate}\,(\textit{cinfo},\\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \textit{t-cond-list},
\end{array}
$$

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

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

Index