

EVENT: Start with the library "c-predefined1".

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                                                                 ;;
;;              EXACT-TIME LEMMA MG-INTEGER-LE                    ;;
;;                                                                 ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

THEOREM: mg-integer-le-args-have-simple-mg-type-refps

```

((car (stmt) = 'predefined-proc-call-mg)
  ∧ (call-name (stmt) = 'mg-integer-le)
  ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
  ∧ ok-mg-statep (mg-state, r-cond-list)
  ∧ signatures-match (mg-alist (mg-state), name-alist))
→ (boolean-identifierp (car (call-actuals (stmt))), mg-alist (mg-state))
    ∧ int-identifierp (cadr (call-actuals (stmt))), mg-alist (mg-state))
    ∧ int-identifierp (caddr (call-actuals (stmt))), mg-alist (mg-state)))

```

THEOREM: mg-integer-le-args-definedp

```

((car (stmt) = 'predefined-proc-call-mg)
  ∧ (call-name (stmt) = 'mg-integer-le)
  ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
  ∧ ok-mg-statep (mg-state, r-cond-list)
  ∧ signatures-match (mg-alist (mg-state), name-alist))
→ (definedp (car (call-actuals (stmt))), mg-alist (mg-state))
    ∧ definedp (cadr (call-actuals (stmt))), mg-alist (mg-state))
    ∧ definedp (caddr (call-actuals (stmt))), mg-alist (mg-state)))

```

THEOREM: mg-integer-le-steps-1-3

```

((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-integer-le)
  ∧ 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))

```

$$\begin{aligned}
& \wedge \text{listp}(\text{ctrl-stk}) \\
& \wedge \text{all-cars-unique}(\text{mg-alist}(\text{mg-state})) \\
& \wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \\
& \wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \\
& \quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \quad \text{temp-stk}) \\
& \wedge \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state})) \\
& \wedge \text{normal}(\text{mg-state}) \\
\rightarrow & (\text{p-step}(\text{p-step}(\text{p-step}(\text{map-down}(\text{mg-state}, \\
& \quad \text{proc-list}, \\
& \quad \text{ctrl-stk}, \\
& \quad \text{temp-stk}, \\
& \quad \text{tag}('pc, \\
& \quad \quad \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo})))), \\
& \quad \text{t-cond-list})))) \\
= & \text{p-state}(\text{tag}('pc, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo}) + 3)), \\
& \quad \text{ctrl-stk}, \\
& \quad \text{push}(\text{value}(\text{caddr}(\text{call-actuals}(\text{stmt})), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \quad \text{push}(\text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \quad \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \quad \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \quad \quad \text{temp-stk}))), \\
& \quad \text{translate-proc-list}(\text{proc-list}), \\
& \quad \text{list}(\text{list}('c-c, \\
& \quad \quad \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}), \text{t-cond-list}))), \\
& \quad \text{MG-MAX-CTRL-STK-SIZE}, \\
& \quad \text{MG-MAX-TEMP-STK-SIZE}, \\
& \quad \text{MG-WORD-SIZE}, \\
& \quad \text{'run}))
\end{aligned}$$

THEOREM: mg-integer-le-step-4

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep}(\text{stmt}, \\
& \quad \text{proc-list}, \\
& \quad \text{list}(\text{length}(\text{temp-stk}), \\
& \quad \quad \text{p-ctrl-stk-size}(\text{ctrl-stk})))) \\
& \wedge (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg}) \\
& \wedge (\text{call-name}(\text{stmt}) = \text{'mg-integer-le}) \\
& \wedge \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
& \wedge \text{ok-mg-def-plistp}(\text{proc-list})
\end{aligned}$$

```

^ 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)) + 3)),
                  ctrl-stk,
                  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-integer-le . 0)),
              push (p-frame (list (cons ('ans,
                                          value (car (call-actuals (stmt)),
                                          bindings (top (ctrl-stk))),
                                          cons ('x,
                                              value (cadr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk))),
                                          cons ('y,
                                              value (caddr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                          tag ('pc,
                                              cons (subr, length (code (cinfo))
                                                  + 4))),

```

```

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

```

THEOREM: mg-integer-le-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-integer-le)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
                code2))
 ∧ user-defined-procp (subr, proc-list)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                               bindings (top (ctrl-stk)),
                               temp-stk)
 ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, '(mg-integer-le . 0)),
                    push (p-frame (list (cons ('ans,
                                              value (car (call-actuals (stmt))),
                                              bindings (top (ctrl-stk)))),
                                      cons ('x,
                                              value (cadr (call-actuals (stmt))),
                                              bindings (top (ctrl-stk)))),
                                      cons ('y,
                                              value (caddr (call-actuals (stmt))),

```

```

bindings (top (ctrl-stk))))),
tag ('pc,
  cons (subr, length (code (cinfo))
        + 4))),
ctrl-stk),
map-down-values (mg-alist (mg-state),
  bindings (top (ctrl-stk)),
  temp-stk),
translate-proc-list (proc-list),
list (list ('c-c,
  mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, ' (mg-integer-le . 1)),
  push (p-frame (list (cons ('ans,
    value (car (call-actuals (stmt)),
    bindings (top (ctrl-stk))))),
    cons ('x,
      value (cadr (call-actuals (stmt)),
      bindings (top (ctrl-stk))))),
    cons ('y,
      value (caddr (call-actuals (stmt)),
      bindings (top (ctrl-stk))))),
    tag ('pc,
      cons (subr, length (code (cinfo))
            + 4))),
    ctrl-stk),
  push (value (caddr (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-integer-le-step-6
 $((n \neq 0)$

$$\begin{aligned}
& \wedge (\neg \text{resources-inadequatep} (stmt, \\
& \quad \text{proc-list}, \\
& \quad \text{list} (\text{length} (temp-stk), \\
& \quad \quad \text{p-ctrl-stk-size} (ctrl-stk)))) \\
& \wedge (\text{car} (stmt) = \text{'predefined-proc-call-mg}) \\
& \wedge (\text{call-name} (stmt) = \text{'mg-integer-le}) \\
& \wedge \text{ok-mg-statement} (stmt, r-cond-list, name-alist, proc-list) \\
& \wedge \text{ok-mg-def-plistp} (proc-list) \\
& \wedge \text{ok-mg-statep} (mg-state, r-cond-list) \\
& \wedge (\text{code} (\text{translate-def-body} (\text{assoc} (subr, proc-list), proc-list)) \\
& \quad = \text{append} (\text{code} (\text{translate} (cinfo, t-cond-list, stmt, proc-list)), \\
& \quad \quad \text{code2})) \\
& \wedge \text{user-defined-procp} (subr, proc-list) \\
& \wedge \text{listp} (ctrl-stk) \\
& \wedge \text{all-cars-unique} (\text{mg-alist} (mg-state)) \\
& \wedge \text{signatures-match} (\text{mg-alist} (mg-state), name-alist) \\
& \wedge \text{mg-vars-list-ok-in-p-state} (\text{mg-alist} (mg-state), \\
& \quad \text{bindings} (\text{top} (ctrl-stk)), \\
& \quad \text{temp-stk}) \\
& \wedge \text{no-p-aliasing} (\text{bindings} (\text{top} (ctrl-stk)), \text{mg-alist} (mg-state)) \\
& \wedge \text{normal} (mg-state)) \\
\rightarrow & (\text{p-step} (\text{p-state} (\text{tag} (\text{'pc}, \text{'(mg-integer-le . 1)}), \\
& \quad \text{push} (\text{p-frame} (\text{list} (\text{cons} (\text{'ans}, \\
& \quad \quad \text{value} (\text{car} (\text{call-actuals} (stmt))), \\
& \quad \quad \text{bindings} (\text{top} (ctrl-stk)))), \\
& \quad \text{cons} (\text{'x}, \\
& \quad \quad \text{value} (\text{cadr} (\text{call-actuals} (stmt))), \\
& \quad \quad \text{bindings} (\text{top} (ctrl-stk)))), \\
& \quad \text{cons} (\text{'y}, \\
& \quad \quad \text{value} (\text{caddr} (\text{call-actuals} (stmt))), \\
& \quad \quad \text{bindings} (\text{top} (ctrl-stk))))), \\
& \quad \text{tag} (\text{'pc}, \\
& \quad \quad \text{cons} (subr, \text{length} (\text{code} (cinfo)) \\
& \quad \quad \quad + 4))), \\
& \quad \text{ctrl-stk}), \\
& \text{push} (\text{value} (\text{caddr} (\text{call-actuals} (stmt)), \\
& \quad \text{bindings} (\text{top} (ctrl-stk))), \\
& \quad \text{map-down-values} (\text{mg-alist} (mg-state), \\
& \quad \quad \text{bindings} (\text{top} (ctrl-stk)), \\
& \quad \quad \text{temp-stk})), \\
& \text{translate-proc-list} (proc-list), \\
& \text{list} (\text{list} (\text{'c-c}, \\
& \quad \text{mg-cond-to-p-nat} (\text{cc} (mg-state), t-cond-list))), \\
& \text{MG-MAX-CTRL-STK-SIZE},
\end{aligned}$$

```

MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, '(mg-integer-le . 2)),
push (p-frame (list (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
cons ('x,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
cons ('y,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
tag ('pc,
cons (subr, length (code (cinfo))
+ 4))),
ctrl-stk),
push (rget (untag (value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk)),
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-le-step-7

```

((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-integer-le)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statementp (mg-state, r-cond-list)

```

```

^ (code (translate-def-body (assoc (subr, proc-list), proc-list))
   = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
              code2))
^ user-defined-procp (subr, proc-list)
^ listp (ctrl-stk)
^ all-cars-unique (mg-alist (mg-state))
^ signatures-match (mg-alist (mg-state), name-alist)
^ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                               bindings (top (ctrl-stk)),
                               temp-stk)
^ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
^ normal (mg-state))
→ (p-step (p-state (tag ('pc, '(mg-integer-le . 2)),
                      push (p-frame (list (cons ('ans,
                                                  value (car (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk)))),
                                                  cons ('x,
                                                  value (cadr (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk)))),
                                                  cons ('y,
                                                  value (caddr (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk))))),
                      tag ('pc,
                          cons (subr, length (code (cinfo)
                                                    + 4))),
                      ctrl-stk),
      push (rget (untag (value (caddr (call-actuals (stmt)),
                                      bindings (top (ctrl-stk)))),
                  map-down-values (mg-alist (mg-state),
                                      bindings (top (ctrl-stk)),
                                      temp-stk)),
          map-down-values (mg-alist (mg-state),
                          bindings (top (ctrl-stk)),
                          temp-stk)),
      translate-proc-list (proc-list),
      list (list ('c-c,
                  mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
      MG-MAX-CTRL-STK-SIZE,
      MG-MAX-TEMP-STK-SIZE,
      MG-WORD-SIZE,
      'run))
= p-state (tag ('pc, '(mg-integer-le . 3)),
           push (p-frame (list (cons ('ans,
                                       value (car (call-actuals (stmt)),

```



```

bindings (top (ctrl-stk))),
cons ('x,
      value (cadr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
cons ('y,
      value (caddr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
tag ('pc,
     cons (subr, length (code (cinfo)
                             + 4))),
ctrl-stk),
push (value (cadr (call-actuals (stmt)),
             bindings (top (ctrl-stk))),
      push (rget (untag (value (caddr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))),
                             map-down-values (mg-alist (mg-state),
                                                         bindings (top (ctrl-stk))),
                                             temp-stk))),
          map-down-values (mg-alist (mg-state),
                              bindings (top (ctrl-stk))),
                          temp-stk))),
translate-proc-list (proc-list),
list (list ('c-c,
            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-le-step-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-integer-le)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
                code2))
 ∧ user-defined-procp (subr, proc-list)

```

```

^ listp (ctrl-stk)
^ all-cars-unique (mg-alist (mg-state))
^ signatures-match (mg-alist (mg-state), name-alist)
^ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                     bindings (top (ctrl-stk)),
                                     temp-stk)
^ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
^ normal (mg-state))
→ (p-step (p-state (tag ('pc, '(mg-integer-le . 3)),
                    push (p-frame (list (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                              cons ('x,
                                              value (cadr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                              cons ('y,
                                              value (caddr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk))))),
                    tag ('pc,
                        cons (subr, length (code (cinfo))
                            + 4))),
                    ctrl-stk),
    push (value (cadr (call-actuals (stmt)),
                bindings (top (ctrl-stk))),
    push (rget (untag (value (caddr (call-actuals (stmt)),
                bindings (top (ctrl-stk)))),
    map-down-values (mg-alist (mg-state),
                bindings (top (ctrl-stk)),
                temp-stk),
    map-down-values (mg-alist (mg-state),
                bindings (top (ctrl-stk)),
                temp-stk))),
    translate-proc-list (proc-list),
    list (list ('c-c,
                mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
    MG-MAX-CTRL-STK-SIZE,
    MG-MAX-TEMP-STK-SIZE,
    MG-WORD-SIZE,
    'run))
= p-state (tag ('pc, '(mg-integer-le . 4)),
    push (p-frame (list (cons ('ans,
                              value (car (call-actuals (stmt)),
                              bindings (top (ctrl-stk)))),
                              cons ('x,
```

```

value (cadr (call-actuals (stmt)),
      bindings (top (ctrl-stk)))),
cons ('y,
      value (caddr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
tag ('pc,
     cons (subr, length (code (cinfo))
           + 4))),
ctrl-stk),
push (rget (untag (value (cadr (call-actuals (stmt)),
                             bindings (top (ctrl-stk)))),
          map-down-values (mg-alist (mg-state),
                                bindings (top (ctrl-stk)),
                                temp-stk)),
      push (rget (untag (value (caddr (call-actuals (stmt)),
                             bindings (top (ctrl-stk)))),
          map-down-values (mg-alist (mg-state),
                                bindings (top (ctrl-stk)),
                                temp-stk)),
          map-down-values (mg-alist (mg-state),
                                bindings (top (ctrl-stk)),
                                temp-stk))),
translate-proc-list (proc-list),
list (list ('c-c,
           mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-le-step-9

```

((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-integer-le)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statement (mg-state, r-cond-list)
 ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
    = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
              code2))

```

```

^ user-defined-procp (subr, proc-list)
^ listp (ctrl-stk)
^ all-cars-unique (mg-alist (mg-state))
^ signatures-match (mg-alist (mg-state), name-alist)
^ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                     bindings (top (ctrl-stk)),
                                     temp-stk)
^ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
^ normal (mg-state)
→ (p-step (p-state (tag ('pc, '(mg-integer-le . 4)),
                    push (p-frame (list (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                              cons ('x,
                                              value (cadr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                              cons ('y,
                                              value (caddr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk))))),
                    tag ('pc,
                        cons (subr, length (code (cinfo))
                            + 4))),
                    ctrl-stk),
    push (rget (untag (value (cadr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
    map-down-values (mg-alist (mg-state),
                        bindings (top (ctrl-stk)),
                        temp-stk),
    push (rget (untag (value (caddr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
    map-down-values (mg-alist (mg-state),
                        bindings (top (ctrl-stk)),
                        temp-stk),
    map-down-values (mg-alist (mg-state),
                        bindings (top (ctrl-stk)),
                        temp-stk)),
    translate-proc-list (proc-list),
    list (list ('c-c,
                mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
    MG-MAX-CTRL-STK-SIZE,
    MG-MAX-TEMP-STK-SIZE,
    MG-WORD-SIZE,
    'run))
= p-state (tag ('pc, '(mg-integer-le . 5)),

```

```

push (p-frame (list (cons ('ans,
                           value (car (call-actuals (stmt)),
                           bindings (top (ctrl-stk)))))
                 cons ('x,
                       value (cadr (call-actuals (stmt)),
                       bindings (top (ctrl-stk)))))
                 cons ('y,
                       value (caddr (call-actuals (stmt)),
                       bindings (top (ctrl-stk)))))
                 tag ('pc,
                     cons (subr, length (code (cinfo)
                     + 4))),
      ctrl-stk),
push (tag ('bool,
          if ilessp (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                    untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))))

          then 't
          else 'f endif),
      map-down-values (mg-alist (mg-state),
                        bindings (top (ctrl-stk)),
                        temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
           mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-le-step-10

```

((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-integer-le)
 ∧ 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)
^  (lt-value ∈ ' (t f))
→  (p-step (p-state (tag ('pc, '(mg-integer-le . 5)),
                        push (p-frame (list (cons ('ans,
                                                    value (car (call-actuals (stmt)),
                                                    bindings (top (ctrl-stk)))),
                                                    cons ('x,
                                                    value (cadr (call-actuals (stmt)),
                                                    bindings (top (ctrl-stk)))),
                                                    cons ('y,
                                                    value (caddr (call-actuals (stmt)),
                                                    bindings (top (ctrl-stk))))),
                        tag ('pc,
                            cons (subr, length (code (cinfo)
                                                    + 4))),
                        ctrl-stk),
                        push (tag ('bool, lt-value),
                            map-down-values (mg-alist (mg-state),
                                                bindings (top (ctrl-stk)),
                                                temp-stk)),
                        translate-proc-list (proc-list),
                        list (list ('c-c,
                                    mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
                        MG-MAX-CTRL-STK-SIZE,
                        MG-MAX-TEMP-STK-SIZE,
                        MG-WORD-SIZE,
                        'run))
=  p-state (tag ('pc, '(mg-integer-le . 6)),
            push (p-frame (list (cons ('ans,
                                        value (car (call-actuals (stmt)),
                                        bindings (top (ctrl-stk)))),
                                        cons ('x,
                                        value (cadr (call-actuals (stmt)),
                                        bindings (top (ctrl-stk)))),
                                        cons ('y,

```

```

value (caddr (call-actuals (stmt)),
       bindings (top (ctrl-stk)))))
tag ('pc,
     cons (subr, length (code (cinfo))
           + 4))),
ctrl-stk),
push (tag ('bool, not-bool (lt-value)),
      map-down-values (mg-alist (mg-state),
                          bindings (top (ctrl-stk)),
                          temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-le-step-11

```

((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-integer-le)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
     = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
               code2))
 ∧ user-defined-procp (subr, proc-list)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                               bindings (top (ctrl-stk)),
                               temp-stk)
 ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ∧ normal (mg-state))
→ (p-step (p-state (tag ('pc, ' (mg-integer-le . 6)),
                       push (p-frame (list (cons ('ans,
                                                  value (car (call-actuals (stmt))),

```

```

bindings (top (ctrl-stk))),
cons ('x,
      value (cadr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
cons ('y,
      value (caddr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
tag ('pc,
     cons (subr, length (code (cinfo))
           + 4))),
ctrl-stk),
push (tag ('bool, not-bool (lt-value)),
      map-down-values (mg-alist (mg-state),
                          bindings (top (ctrl-stk)),
                          temp-stk))),
translate-proc-list (proc-list),
list (list ('c-c,
            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, ' (mg-integer-le . 7)),
           push (p-frame (list (cons ('ans,
                                      value (car (call-actuals (stmt)),
                                                bindings (top (ctrl-stk)))),
                                cons ('x,
                                      value (cadr (call-actuals (stmt)),
                                                bindings (top (ctrl-stk)))),
                                cons ('y,
                                      value (caddr (call-actuals (stmt)),
                                                bindings (top (ctrl-stk))))),
              tag ('pc,
                   cons (subr, length (code (cinfo))
                         + 4))),
              ctrl-stk),
           push (value (car (call-actuals (stmt)),
                        bindings (top (ctrl-stk))),
               push (tag ('bool, not-bool (lt-value)),
                     map-down-values (mg-alist (mg-state),
                                             bindings (top (ctrl-stk)),
                                             temp-stk))),
               translate-proc-list (proc-list),
               list (list ('c-c,

```


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

THEOREM: mg-integer-le-step-12

$((n \neq 0)$
 $\wedge \quad (\neg \text{resources-inadequatep}(\text{stmt},$
 $\quad \quad \quad \text{proc-list},$
 $\quad \quad \quad \text{list}(\text{length}(\text{temp-stk}),$
 $\quad \quad \quad \text{p-ctrl-stk-size}(\text{ctrl-stk}))))$
 $\wedge \quad (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg})$
 $\wedge \quad (\text{call-name}(\text{stmt}) = \text{'mg-integer-le})$
 $\wedge \quad \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list})$
 $\wedge \quad \text{ok-mg-def-plistp}(\text{proc-list})$
 $\wedge \quad \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list})$
 $\wedge \quad (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))$
 $\quad = \quad \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})),$
 $\quad \quad \quad \text{code2}))$
 $\wedge \quad \text{user-defined-procp}(\text{subr}, \text{proc-list})$
 $\wedge \quad \text{listp}(\text{ctrl-stk})$
 $\wedge \quad \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))$
 $\wedge \quad \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})$
 $\wedge \quad \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}),$
 $\quad \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})),$
 $\quad \quad \quad \text{temp-stk})$
 $\wedge \quad \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))$
 $\wedge \quad \text{normal}(\text{mg-state}))$
 $\rightarrow \quad (\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, \text{'(mg-integer-le . 7)}),$
 $\quad \quad \quad \text{push}(\text{p-frame}(\text{list}(\text{cons}(\text{'ans},$
 $\quad \quad \quad \quad \quad \text{value}(\text{car}(\text{call-actuals}(\text{stmt})),$
 $\quad \quad \quad \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})))),$
 $\quad \quad \quad \text{cons}(\text{'x},$
 $\quad \quad \quad \quad \quad \text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})),$
 $\quad \quad \quad \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})))),$
 $\quad \quad \quad \text{cons}(\text{'y},$
 $\quad \quad \quad \quad \quad \text{value}(\text{caddr}(\text{call-actuals}(\text{stmt})),$
 $\quad \quad \quad \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))))),$
 $\quad \quad \quad \text{tag}(\text{'pc},$
 $\quad \quad \quad \quad \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo}))$
 $\quad \quad \quad \quad \quad + \quad 4))),$
 $\quad \quad \quad \text{ctrl-stk}),$
 $\quad \quad \quad \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt})),$

```

bindings (top (ctrl-stk))),
push (tag ('bool, not-bool (lt-value)),
      map-down-values (mg-alist (mg-state),
                           bindings (top (ctrl-stk)),
                           temp-stk))),
translate-proc-list (proc-list),
list (list ('c-c,
            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, ' (mg-integer-le . 8)),
           push (p-frame (list (cons ('ans,
                                     value (car (call-actuals (stmt)),
                                     bindings (top (ctrl-stk)))),
                                cons ('x,
                                     value (cadr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk)))),
                                cons ('y,
                                     value (caddr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
               tag ('pc,
                   cons (subr, length (code (cinfo))
                       + 4))),
               ctrl-stk),
           rput (tag ('bool, not-bool (lt-value)),
                untag (value (car (call-actuals (stmt)),
                              bindings (top (ctrl-stk)))),
                map-down-values (mg-alist (mg-state),
                              bindings (top (ctrl-stk)),
                              temp-stk)),
                translate-proc-list (proc-list),
                list (list ('c-c,
                            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
                MG-MAX-CTRL-STK-SIZE,
                MG-MAX-TEMP-STK-SIZE,
                MG-WORD-SIZE,
                'run))

```

THEOREM: mg-integer-le-step-13-true-case

$((n \neq 0)$
 $\wedge \quad (\neg \text{resources-inadequatep} (stmt,$
 $\quad \quad \quad proc-list,$

```

                                list (length (temp-stk),
                                      p-ctrl-stk-size (ctrl-stk))))
^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-integer-le)
^ 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)
^ illessp (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                         mg-alist (mg-state))))),
           untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                         mg-alist (mg-state)))))))
→ (p-step (p-state (tag ('pc, '(mg-integer-le . 8)),
                    push (p-frame (list (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                          cons ('x,
                                              value (cadr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                          cons ('y,
                                              value (caddr (call-actuals (stmt)),
                                              bindings (top (ctrl-stk))))),
                                          tag ('pc,
                                              cons (subr, length (code (cinfo))
                                                  + 4))),
                                              ctrl-stk),
                    rput (tag ('bool, not-bool ('t)),
                        untag (value (car (call-actuals (stmt)),
                                      bindings (top (ctrl-stk))))),
                        map-down-values (mg-alist (mg-state),
                                      bindings (top (ctrl-stk)),
                                      temp-stk)),
                    translate-proc-list (proc-list),

```

```

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

```

$\text{mg-cond-to-p-nat}(\text{cc}(\text{mg-meaning-r}(\text{stmt},$
 $\text{proc-list},$
 $\text{mg-state},$
 $n,$
 $\text{list}(\text{length}(\text{temp-stk}),$
 $\text{p-ctrl-stk-size}(\text{ctrl-stk}))))),$
 $t\text{-cond-list}))))),$
 $\text{MG-MAX-CTRL-STK-SIZE},$
 $\text{MG-MAX-TEMP-STK-SIZE},$
 $\text{MG-WORD-SIZE},$
 $\text{'run}))$

THEOREM: mg-integer-le-step-13-false-case

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

```

value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons ('y,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
tag ('pc,
cons (subr, length (code (cinfo)
+ 4))),
ctrl-stk),
rput (tag ('bool, not-bool ('f)),
untag (value (car (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc,
cons (subr,
if normal (mg-meaning-r (stmt,
proc-list,
mg-state,
n,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk))))
then length (code (translate (cinfo,
t-cond-list,
stmt,
proc-list)))
else find-label (fetch-label (cc (mg-meaning-r (stmt,
proc-list,
mg-state,
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-integer-le-exact-time-lemma

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                              proc-list,
                              list (length (temp-stk),
                                          p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-le)
 ∧ 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-INTEGER-UNARY-MINUS              ;;
;;                                                                                   ;;
;;                                                                                   ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

THEOREM: min-int-only-non-negatable-small-int
 $(\text{small-integerp}(x, n) \wedge (x \neq (- \exp(2, n - 1))))$
 $\rightarrow \text{small-integerp}(\text{inegate}(x), n)$

THEOREM: mg-integer-unary-minus-args-have-simple-mg-type-refps
 $((\text{car}(stmt) = \text{'predefined-proc-call-mg})$
 $\wedge (\text{call-name}(stmt) = \text{'mg-integer-unary-minus})$
 $\wedge \text{ok-mg-statement}(stmt, r\text{-cond-list}, name\text{-alist}, proc\text{-list})$
 $\wedge \text{ok-mg-statep}(mg\text{-state}, r\text{-cond-list})$
 $\wedge \text{signatures-match}(mg\text{-alist}(mg\text{-state}), name\text{-alist}))$
 $\rightarrow (\text{int-identifierp}(\text{car}(\text{call-actuals}(stmt)), mg\text{-alist}(mg\text{-state}))$
 $\wedge \text{int-identifierp}(\text{cadr}(\text{call-actuals}(stmt)), mg\text{-alist}(mg\text{-state})))$

THEOREM: mg-integer-unary-minus-args-definedp
 $((\text{car}(stmt) = \text{'predefined-proc-call-mg})$
 $\wedge (\text{call-name}(stmt) = \text{'mg-integer-unary-minus})$
 $\wedge \text{ok-mg-statement}(stmt, r\text{-cond-list}, name\text{-alist}, proc\text{-list})$
 $\wedge \text{ok-mg-statep}(mg\text{-state}, r\text{-cond-list})$

$$\begin{aligned}
& \wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \\
\rightarrow & (\text{definedp}(\text{car}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})) \\
& \wedge \text{definedp}(\text{cadr}(\text{call-actuals}(\text{stmt})), \text{mg-alist}(\text{mg-state})))
\end{aligned}$$

THEOREM: mg-integer-unary-minus-steps-1-2

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep}(\text{stmt}, \\
& \quad \text{proc-list}, \\
& \quad \text{list}(\text{length}(\text{temp-stk}), \\
& \quad \text{p-ctrl-stk-size}(\text{ctrl-stk})))) \\
& \wedge (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg}) \\
& \wedge (\text{call-name}(\text{stmt}) = \text{'mg-integer-unary-minus}) \\
& \wedge \text{ok-mg-statement}(\text{stmt}, \text{r-cond-list}, \text{name-alist}, \text{proc-list}) \\
& \wedge \text{ok-mg-def-plistp}(\text{proc-list}) \\
& \wedge \text{ok-mg-statep}(\text{mg-state}, \text{r-cond-list}) \\
& \wedge (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list})) \\
& \quad = \text{append}(\text{code}(\text{translate}(\text{cinfo}, \text{t-cond-list}, \text{stmt}, \text{proc-list})), \\
& \quad \text{code2})) \\
& \wedge \text{user-defined-procp}(\text{subr}, \text{proc-list}) \\
& \wedge \text{listp}(\text{ctrl-stk}) \\
& \wedge \text{all-cars-unique}(\text{mg-alist}(\text{mg-state})) \\
& \wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist}) \\
& \wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}), \\
& \quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \quad \text{temp-stk}) \\
& \wedge \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state})) \\
& \wedge \text{normal}(\text{mg-state})) \\
\rightarrow & (\text{p-step}(\text{p-step}(\text{map-down}(\text{mg-state}, \\
& \quad \text{proc-list}, \\
& \quad \text{ctrl-stk}, \\
& \quad \text{temp-stk}, \\
& \quad \text{tag}(\text{'pc}, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo})))), \\
& \quad \text{t-cond-list}))) \\
& = \text{p-state}(\text{tag}(\text{'pc}, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo})) + 2)), \\
& \quad \text{ctrl-stk}, \\
& \quad \text{push}(\text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \quad \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt})), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))), \\
& \quad \text{map-down-values}(\text{mg-alist}(\text{mg-state}), \\
& \quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})), \\
& \quad \quad \text{temp-stk}))), \\
& \quad \text{translate-proc-list}(\text{proc-list}), \\
& \quad \text{list}(\text{list}(\text{'c-c},
\end{aligned}$$

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

THEOREM: mg-integer-unary-minus-step-3

$((n \neq 0)$
 $\wedge (\neg \text{resources-inadequatep}(\text{stmt},$
 $\text{proc-list},$
 $\text{list}(\text{length}(\text{temp-stk}),$
 $\text{p-ctrl-stk-size}(\text{ctrl-stk}))))$
 $\wedge (\text{car}(\text{stmt}) = \text{'predefined-proc-call-mg})$
 $\wedge (\text{call-name}(\text{stmt}) = \text{'mg-integer-unary-minus})$
 $\wedge \text{ok-mg-statement}(\text{stmt}, r\text{-cond-list}, \text{name-alist}, \text{proc-list})$
 $\wedge \text{ok-mg-def-plistp}(\text{proc-list})$
 $\wedge \text{ok-mg-statep}(\text{mg-state}, r\text{-cond-list})$
 $\wedge (\text{code}(\text{translate-def-body}(\text{assoc}(\text{subr}, \text{proc-list}), \text{proc-list}))$
 $\quad = \text{append}(\text{code}(\text{translate}(\text{cinfo}, t\text{-cond-list}, \text{stmt}, \text{proc-list})),$
 $\quad \text{code2}))$
 $\wedge \text{user-defined-procp}(\text{subr}, \text{proc-list})$
 $\wedge \text{listp}(\text{ctrl-stk})$
 $\wedge \text{all-cars-unique}(\text{mg-alist}(\text{mg-state}))$
 $\wedge \text{signatures-match}(\text{mg-alist}(\text{mg-state}), \text{name-alist})$
 $\wedge \text{mg-vars-list-ok-in-p-state}(\text{mg-alist}(\text{mg-state}),$
 $\quad \text{bindings}(\text{top}(\text{ctrl-stk})),$
 $\quad \text{temp-stk})$
 $\wedge \text{no-p-aliasing}(\text{bindings}(\text{top}(\text{ctrl-stk})), \text{mg-alist}(\text{mg-state}))$
 $\wedge \text{normal}(\text{mg-state}))$
 $\rightarrow (\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, \text{cons}(\text{subr}, \text{length}(\text{code}(\text{cinfo})) + 2)),$
 $\quad \text{ctrl-stk},$
 $\quad \text{push}(\text{value}(\text{cadr}(\text{call-actuals}(\text{stmt})),$
 $\quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))),$
 $\quad \text{push}(\text{value}(\text{car}(\text{call-actuals}(\text{stmt})),$
 $\quad \quad \text{bindings}(\text{top}(\text{ctrl-stk}))),$
 $\quad \text{map-down-values}(\text{mg-alist}(\text{mg-state}),$
 $\quad \quad \text{bindings}(\text{top}(\text{ctrl-stk})),$
 $\quad \quad \text{temp-stk}))),$
 $\quad \text{translate-proc-list}(\text{proc-list}),$
 $\quad \text{list}(\text{list}(\text{'c-c},$
 $\quad \quad \text{mg-cond-to-p-nat}(\text{cc}(\text{mg-state}), t\text{-cond-list}))),$
 $\quad \text{MG-MAX-CTRL-STK-SIZE},$
 $\quad \text{MG-MAX-TEMP-STK-SIZE},$
 $\quad \text{MG-WORD-SIZE},$

```

      'run))
=  p-state (tag ('pc, '(mg-integer-unary-minus . 0)),
      push (p-frame (cons (cons ('ans,
                                value (car (call-actuals (stmt))),
                                bindings (top (ctrl-stk)))))
              cons (cons ('x,
                          value (cadr (call-actuals (stmt))),
                          bindings (top (ctrl-stk)))))
              '((min-int int -2147483648)
                (temp-x int 0))),
      tag ('pc,
            cons (subr, length (code (cinfo)
                                     + 3))),
      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-integer-unary-minus-steps-4-8

```

((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-integer-unary-minus)
 ∧ 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-step (p-step (p-state (tag ('pc,
' (mg-integer-unary-minus
. 0)),
push (p-frame (cons (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
cons (cons ('x,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
' ((min-int
int
-2147483648)
(temp-x
int
0))))),
tag ('pc,
cons (subr,
length (code (cinfo))
+ 3))),
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-integer-unary-minus . 5)),
push (p-frame (list (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
cons ('x,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk)))),
' (min-int int -2147483648),
cons ('temp-x,

```

```

                                mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))),
                                                                mg-alist (mg-state))))),
                                tag ('pc,
                                    cons (subr, length (code (cinfo))
                                          + 3)),
                                ctrl-stk),
push (tag ('bool,
          if untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))),
                                                                mg-alist (mg-state))))
              = untag ('(int -2147483648))
              then 't
              else 'f endif),
      map-down-values (mg-alist (mg-state),
                        bindings (top (ctrl-stk)),
                        temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
            mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

;; (test-bool-and-jump f 0)
;; (push-constant (nat 1))
;; (pop-global c-c)
;; (jump 1)
;; (dl 1 nil (ret))

```

THEOREM: mg-integer-unary-minus-steps-9-13-error-case

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                             proc-list,
                             list (length (temp-stk),
                                     p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-unary-minus)
 ∧ 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)
^ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))))
    = untag ('(int -2147483648))))
→ (p-step (p-step (p-step (p-step (p-step (p-state (tag ('pc,
                                                         '(mg-integer-unary-minus
                                                           . 5)),
                                                         push (p-frame (list (cons ('ans,
                                                                 value (car (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk))),
                                                                 cons ('x,
                                                                 value (cadr (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk))),
                                                                 '(min-int
                                                                    int
                                                                    -2147483648),
                                                                 cons ('temp-x,
                                                                 mg-to-p-simple-literal (caddr (assoc
                                                                 (cadr (call-actuals (stmt)),
                                                                 mg-alist (mg-state))))))
                                                                 tag ('pc,
                                                                 cons (subr,
                                                                 length (code (cinfo))
                                                                 + 3))),
                                                                 ctrl-stk),
                                                         push (tag ('bool, 't),
                                                         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)) + 3)),
      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-integer-unary-minus-steps-9-12-nonerror-case

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                              proc-list,
                              list (length (temp-stk),
                                      p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-unary-minus)
 ∧ 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)
 ∧ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))))
   ≠ untag ('(int -2147483648))))
→ (p-step (p-step (p-step (p-step (p-state (tag ('pc,
                                                  '(mg-integer-unary-minus
                                                    . 5)),
                                                  push (p-frame (list (cons ('ans,
                                                                           value (car (call-actuals (stmt)),
                                                                           bindings (top (ctrl-stk))))),

```



```

cons ('x,
      value (cadr (call-actuals (stmt)),
              bindings (top (ctrl-stk)))),
      ' (min-int
          int
          -2147483648),
cons ('temp-x,
      mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                              mg-alist (mg-state))))),
tag ('pc,
     cons (subr,
            length (code (cinfo))
            + 3))),
ctrl-stk),
push (tag ('bool, 'f),
      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-integer-unary-minus . 12)),
           push (p-frame (list (cons ('ans,
                                      value (car (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk)))),
                                      cons ('x,
                                            value (cadr (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk)))),
                                      ' (min-int int -2147483648),
                                      cons ('temp-x,
                                            mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                                      mg-alist (mg-state))))))),
tag ('pc,
     cons (subr, length (code (cinfo))
            + 3))),
ctrl-stk),
push (value (car (call-actuals (stmt)),
            bindings (top (ctrl-stk))),
      push (tag ('int,

```

```

                                inegate (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))

;; (deposit-temp-stk)
;; (dl 1 nil (ret))

```

THEOREM: mg-integer-unary-minus-steps-13-14-nonerror-case

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                               proc-list,
                               list (length (temp-stk),
                                           p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-unary-minus)
 ∧ 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)
 ∧ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))))
    ≠ untag ('(int -2147483648))))
→ (p-step (p-step (p-state (tag ('pc,

```

```

      '(mg-integer-unary-minus . 12)),
push (p-frame (list (cons ('ans,
                           value (car (call-actuals (stmt)),
                           bindings (top (ctrl-stk)))),
                     cons ('x,
                           value (cadr (call-actuals (stmt)),
                           bindings (top (ctrl-stk)))),
                     '(min-int int
                        -2147483648),
                     cons ('temp-x,
                           mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                    mg-alist (mg-state))))),
                     tag ('pc,
                           cons (subr,
                                  length (code (cinfo))
                                  + 3))),
      ctrl-stk),
push (value (car (call-actuals (stmt)),
            bindings (top (ctrl-stk))),
push (tag ('int,
          inegate (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)) + 3)),
          ctrl-stk,
          rput (tag ('int,
                    inegate (untag (caddr (assoc (cadr (call-actuals (stmt)),
                                                  mg-alist (mg-state))))),
                    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-integer-unary-minus-push-c-c-effect

```

((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-integer-unary-minus)
 ∧ 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)
 ∧ 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)) + 3)),
                    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)) + 4)),
                  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-integer-unary-minus-sub1-nat-effect

$((n \neq 0)$
 $\wedge (\neg \text{resources-inadequatep}(stmt,$
 $\quad proc-list,$
 $\quad \text{list}(\text{length}(temp-stk),$
 $\quad \text{p-ctrl-stk-size}(ctrl-stk))))$
 $\wedge (\text{car}(stmt) = \text{'predefined-proc-call-mg})$
 $\wedge (\text{call-name}(stmt) = \text{'mg-integer-unary-minus})$
 $\wedge \text{ok-mg-statement}(stmt, r-cond-list, name-alist, proc-list)$
 $\wedge \text{ok-mg-def-plistp}(proc-list)$
 $\wedge \text{ok-mg-statep}(mg-state, r-cond-list)$
 $\wedge (\text{code}(\text{translate-def-body}(\text{assoc}(subr, proc-list), proc-list))$
 $\quad = \text{append}(\text{code}(\text{translate}(cinfo, t-cond-list, stmt, proc-list)),$
 $\quad \text{code2}))$
 $\wedge \text{user-defined-procp}(subr, proc-list)$
 $\wedge \text{listp}(ctrl-stk)$
 $\wedge \text{all-cars-unique}(\text{mg-alist}(mg-state))$
 $\wedge \text{signatures-match}(\text{mg-alist}(mg-state), name-alist)$
 $\wedge \text{normal}(mg-state)$
 $\wedge (cc-value \in \text{list}(\text{'(nat 1)}, \text{'(nat 2)}))$
 $\rightarrow (\text{p-step}(\text{p-state}(\text{tag}(\text{'pc}, \text{cons}(subr, \text{length}(\text{code}(cinfo)) + 4)),$
 $\quad ctrl-stk,$
 $\quad \text{push}(cc-value, temp-stk),$
 $\quad \text{translate-proc-list}(proc-list),$
 $\quad \text{list}(\text{list}(\text{'c-c}, cc-value)),$
 $\quad \text{MG-MAX-CTRL-STK-SIZE},$
 $\quad \text{MG-MAX-TEMP-STK-SIZE},$
 $\quad \text{MG-WORD-SIZE},$
 $\quad \text{'run}))$
 $= \text{p-state}(\text{tag}(\text{'pc}, \text{cons}(subr, \text{length}(\text{code}(cinfo)) + 5)),$
 $\quad ctrl-stk,$
 $\quad \text{push}(\text{tag}(\text{'nat}, \text{untag}(cc-value) - 1), temp-stk),$
 $\quad \text{translate-proc-list}(proc-list),$
 $\quad \text{list}(\text{list}(\text{'c-c}, cc-value)),$
 $\quad \text{MG-MAX-CTRL-STK-SIZE},$
 $\quad \text{MG-MAX-TEMP-STK-SIZE},$
 $\quad \text{MG-WORD-SIZE},$
 $\quad \text{'run}))$

THEOREM: mg-integer-unary-minus-step-16-error

$((n \neq 0)$
 $\wedge (\neg \text{resources-inadequatep}(stmt,$

```

                                proc-list,
                                list (length (temp-stk),
                                        p-ctrl-stk-size (ctrl-stk))))
^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-integer-unary-minus)
^ 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)
^ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))))
  = untag ('(int -2147483648)))
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 5)),
                                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: mg-integer-unary-minus-step-17-nonerror
 $((n \neq 0)$
 $\wedge \quad (\neg \text{resources-inadequatep} (stmt,$

```

                                proc-list,
                                list (length (temp-stk),
                                        p-ctrl-stk-size (ctrl-stk))))
^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-integer-unary-minus)
^ 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)
^ (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))))
  ≠ untag ('(int -2147483648)))
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 5)),
                                ctrl-stk,
                                push (tag ('nat,
                                            untag (mg-cond-to-p-nat (cc (mg-state),
                                                                    t-cond-list)) - 1),
                                            rput (tag ('int,
                                                    inegate (untag (caddr (assoc (cadr (call-actuals (stmt)),
                                                                    mg-alist (mg-state)))))),
                                                    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-integer-unary-minus-exact-time-lemma

(($n \neq 0$)
 $\wedge$ ($\neg$ 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-integer-unary-minus)
 $\wedge$ 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 (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))

```

EVENT: Make the library "**c-predefined2**".

Index

- all-cars-unique, 2–4, 6, 8, 10, 12, 14, 15, 17, 19, 21, 24, 26–28, 31, 32, 34, 36–38, 40, 42
- bindings, 2–36, 38–43
- boolean-identifierp, 1
- call-actuals, 1–19, 21, 22, 25–35, 38, 40
- call-name, 1, 2, 4, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25–28, 30, 32, 34, 36–38, 40, 42
- cc, 2–18, 20–25, 27–31, 33–36, 39–41, 43
- clock, 24, 42
- code, 1–24, 26–43
- definedp, 1, 26
- exp, 25
- fetch-label, 20, 22, 24, 39, 41, 43
- find-label, 20, 23, 24, 39, 41, 43
- illessp, 13, 19, 21
- inegate, 25, 34, 35, 40
- int-identifierp, 1, 25
- label-alist, 20, 22, 24, 39, 41, 43
- length, 1–43
- map-down, 2, 24, 26, 42
- map-down-values, 2–16, 18–20, 22, 23, 25–35, 38–41, 43
- mg-alist, 1–43
- mg-cond-to-p-nat, 2–18, 20–23, 25, 27–31, 33–36, 39–41, 43
- mg-integer-le-args-definedp, 1
- mg-integer-le-args-have-simple-mg-type-refps, 1
- mg-integer-le-exact-time-lemma, 23
- mg-integer-le-step-10, 13
- mg-integer-le-step-11, 15
- mg-integer-le-step-12, 17
- mg-integer-le-step-13-false-case, 21
- mg-integer-le-step-13-true-case, 18
- mg-integer-le-step-4, 2
- mg-integer-le-step-5, 4
- mg-integer-le-step-6, 5
- mg-integer-le-step-7, 7
- mg-integer-le-step-8, 9
- mg-integer-le-step-9, 11
- mg-integer-le-steps-1-3, 1
- mg-integer-unary-minus-args-defi-nedp, 25
- mg-integer-unary-minus-args-have-simple-mg-type-refps, 25
- mg-integer-unary-minus-exact-time-lemma, 42
- mg-integer-unary-minus-push-c-c-effect, 36
- mg-integer-unary-minus-step-16-error, 37
- mg-integer-unary-minus-step-17-nonerror, 39
- mg-integer-unary-minus-step-3, 27
- mg-integer-unary-minus-steps-1-2, 26
- mg-integer-unary-minus-steps-13-14-nonerror-case, 34
- mg-integer-unary-minus-steps-4-8, 28
- mg-integer-unary-minus-steps-9-12-nonerror-case, 32
- mg-integer-unary-minus-steps-9-13-error-case, 30
- mg-integer-unary-minus-sub1-nat-effect, 37
- mg-max-ctrl-stk-size, 2–18, 20–23, 25, 27–40, 42, 43
- mg-max-temp-stk-size, 2–5, 7–18, 20–23, 25, 27–40, 42, 43
- mg-meaning-r, 20–25, 38, 39, 41–43

mg-to-p-simple-literal, 13, 19, 21, 30–35, 38, 40
 mg-vars-list-ok-in-p-state, 2–4, 6, 8, 10, 12, 14, 15, 17, 19, 21, 24, 26, 27, 29, 31, 32, 34, 38, 40, 42
 mg-word-size, 2–5, 7–18, 20–23, 25, 27–40, 42, 43
 min-int-only-non-negatable-small-int, 25
 no-p-aliasing, 2–4, 6, 8, 10, 12, 14, 15, 17, 19, 21, 24, 26, 27, 29, 31, 32, 34, 36, 38, 40, 42
 normal, 2–4, 6, 8, 10, 12, 14, 15, 17, 19–22, 24, 26, 27, 29, 31, 32, 34, 36–38, 40–42
 not-bool, 15, 16, 18, 19, 22
 ok-mg-def-plistp, 1, 2, 4, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 26–28, 30, 32, 34, 36–38, 40, 42
 ok-mg-statement, 1, 2, 4, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25–28, 30, 32, 34, 36–38, 40, 42
 ok-mg-statep, 1, 3, 4, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25–28, 30, 32, 34, 36–38, 40, 42
 p, 24, 42
 p-ctrl-stk-size, 1, 2, 4, 6, 7, 9, 11, 13, 15, 17, 19–28, 30, 32, 34, 36–43
 p-frame, 3, 5–19, 22, 28–31, 33, 35
 p-state, 2–5, 7–18, 20–23, 25, 27–40, 42, 43
 p-step, 2, 3, 5, 7, 8, 10, 12, 14, 16, 18, 20, 22, 26, 28, 29, 31, 33, 35–38, 40
 push, 2–19, 22, 26–31, 33–38, 40
 resources-inadequatep, 1, 2, 4, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 26–28, 30, 32, 34, 36–38, 40, 42
 rget, 7–12
 rput, 18, 19, 22, 35, 40
 signatures-match, 1–4, 6, 8, 10, 12, 14, 15, 17, 19, 21, 24–28, 31, 32, 34, 36–38, 40, 42
 small-integerp, 25
 tag, 2–24, 26–43
 top, 2–36, 38–43
 translate, 1, 3, 4, 6, 8, 9, 11, 13, 15, 17, 19–24, 26–28, 30, 32, 34, 36–43
 translate-def-body, 1, 3, 4, 6, 8, 9, 11, 13, 15, 17, 19, 21, 23, 26–28, 30, 32, 34, 36–38, 40, 42
 translate-proc-list, 2–16, 18–20, 22, 23, 25–41, 43
 untag, 7–13, 18, 19, 21, 22, 30–32, 34, 35, 37, 38, 40
 user-defined-procp, 1, 3, 4, 6, 8, 9, 12, 14, 15, 17, 19, 21, 23, 26–28, 31, 32, 34, 36–38, 40, 42
 value, 2–19, 21, 22, 26–29, 31–33, 35, 40