

EVENT: Start with the library "c-predefined2".

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;                                EXACT-TIME LEMMA MG-INTEGGER-ADD
;;
;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

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

```

((car (stmt) = 'predefined-proc-call-mg)
  ^ (call-name (stmt) = 'mg-integer-add)
  ^ 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))
→ (int-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-add-args-definedp

```

((car (stmt) = 'predefined-proc-call-mg)
  ^ (call-name (stmt) = 'mg-integer-add)
  ^ 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-add-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-add)
  ^ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
  ^ ok-mg-def-plistp (proc-list)
  ^ ok-mg-statep (mg-state, r-cond-list)
  ^ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
                code2))
  ^ user-defined-procp (subr, proc-list)

```

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

THEOREM: mg-integer-add-step-4

$((n \neq 0)$
 \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-add)
 \wedge ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
 \wedge ok-mg-def-plistp (*proc-list*)

```

^ ok-mg-statep (mg-state, r-cond-list)
^ (code (translate-def-body (assoc (subr, proc-list), proc-list))
  = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
            code2))
^ user-defined-procp (subr, proc-list)
^ listp (ctrl-stk)
^ all-cars-unique (mg-alist (mg-state))
^ signatures-match (mg-alist (mg-state), name-alist)
^ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                bindings (top (ctrl-stk)),
                                temp-stk)
^ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
^ normal (mg-state)
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 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-add . 0)),
            push (p-frame (cons (cons ('ans,
                                    value (car (call-actuals (stmt)),
                                    bindings (top (ctrl-stk))),
                                    cons (cons ('y,
                                                value (cadr (call-actuals (stmt)),
                                                bindings (top (ctrl-stk))),
                                                cons (cons ('z,
                                                            value (caddr (call-actuals (stmt)),
                                                            bindings (top (ctrl-stk))),
                                                            '(t1 int 0))))),
                                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-add-steps-5-9

```

((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-add)
 ∧ 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-add
                                                         . 0)),
                                                         push (p-frame (cons (cons ('ans,
                                                                 value (car (call-actuals (stmt))),
                                                                 bindings (top (ctrl-stk)))),
                                                                 cons (cons ('y,
                                                                 value (cadr (call-actuals (st

```

```

bindings (top (ctrl-stk))
cons (cons ('z,
value (caddr (call-act
bindings (top (
'((t1
int
0))))),
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-add . 5)),
push (p-frame (cons (cons ('ans,
value (car (call-actuals (stmt))),
bindings (top (ctrl-stk)))))
cons (cons ('y,
value (cadr (call-actuals (stmt))),
bindings (top (ctrl-stk)))))
cons (cons ('z,
value (caddr (call-actuals (stmt))),
bindings (top (ctrl-stk)))))
'((t1 int 0))))),
tag ('pc,
cons (subr, length (code (cinfo))
+ 4))),
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))))
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state)))))
push ('(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))

```

THEOREM: mg-integer-add-step-10-nonerror

```

((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-add)
 ∧ 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)
 ∧ small-integerp (iplus (0,
                          iplus (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))),
                                  untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))))),
                          MG-WORD-SIZE))
 → (p-step (p-state (tag ('pc, '(mg-integer-add . 5)),
                    push (p-frame (cons (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                              bindings (top (ctrl-stk))))),
                                      cons (cons ('y,

```

```

value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons (cons ('z,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
'((t1 int 0)))),
tag ('pc,
cons (subr, length (code (cinfo)
+ 4))),
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state)))),
push ('(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-add . 6)),
push (p-frame (cons (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons (cons ('y,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons (cons ('z,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
'((t1 int 0)))),
tag ('pc,
cons (subr, length (code (cinfo)
+ 4))),
ctrl-stk),
push (tag ('int,
fix-small-integer (iplus (0,
iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (ca
mg-alist

```

```

untag (mg-to-p-simple-literal (caddr (assoc (caddr (c
mg-alist
MG-WORD-SIZE))),
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))

```

THEOREM: mg-integer-add-steps-11-17-nonerror

```

((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-add)
^ 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-step (p-step (p-state (tag ('pc,
'mg-integer-add
. 6)),
push (p-frame (cons (cons ('ans,
value (car (call-ac
bindings (t

```



```

cons (cons ('y,
           value (cadr (
                 bindi
                 cons (cons ('z,
                           value
                           '( (t1
                              int
                              0))))),
           tag ('pc,
              cons (subr,
                   length (code (cinfo)
                               + 4))),
              ctrl-stk),
      push (sum,
           push (tag ('bool,
                    'f),
               map-down-values (mg-alist (mg-
                                         bindings (top
                                         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)) + 4),
                        ctrl-stk,
                        rput (sum,
                            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-add-push-cc

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

THEOREM: mg-integer-add-sub1-cc

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep}(stmt, \\
& \quad \text{proc-list}, \\
& \quad \text{list}(\text{length}(temp-stk), \\
& \quad \quad \text{p-ctrl-stk-size}(ctrl-stk))))
\end{aligned}$$

$$\begin{aligned}
& \wedge (\text{car}(stmt) = \text{'predefined-proc-call-mg}) \\
& \wedge (\text{call-name}(stmt) = \text{'mg-integer-add}) \\
& \wedge \text{ok-mg-statement}(stmt, r\text{-cond-list}, name\text{-alist}, proc\text{-list}) \\
& \wedge \text{ok-mg-def-plistp}(proc\text{-list}) \\
& \wedge \text{ok-mg-statep}(mg\text{-state}, r\text{-cond-list}) \\
& \wedge (\text{code}(\text{translate-def-body}(\text{assoc}(subr, proc\text{-list}), proc\text{-list})) \\
& \quad = \text{append}(\text{code}(\text{translate}(cinfo, t\text{-cond-list}, stmt, proc\text{-list})), \\
& \quad \quad \quad code2)) \\
& \wedge \text{user-defined-procp}(subr, proc\text{-list}) \\
& \wedge \text{listp}(ctrl\text{-stk}) \\
& \wedge \text{all-cars-unique}(mg\text{-alist}(mg\text{-state})) \\
& \wedge \text{signatures-match}(mg\text{-alist}(mg\text{-state}), name\text{-alist}) \\
& \wedge \text{normal}(mg\text{-state}) \\
& \wedge (cc\text{-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)) + 5)), \\
& \quad \quad \quad ctrl\text{-stk}, \\
& \quad \quad \quad \text{push}(cc\text{-value}, temp\text{-stk}), \\
& \quad \quad \quad \text{translate-proc-list}(proc\text{-list}), \\
& \quad \quad \quad \text{list}(\text{list}(\text{'c-c}, cc\text{-value})), \\
& \quad \quad \quad \text{MG-MAX-CTRL-STK-SIZE}, \\
& \quad \quad \quad \text{MG-MAX-TEMP-STK-SIZE}, \\
& \quad \quad \quad \text{MG-WORD-SIZE}, \\
& \quad \quad \quad \text{'run})) \\
& = \text{p-state}(\text{tag}(\text{'pc}, \text{cons}(subr, \text{length}(\text{code}(cinfo)) + 6)), \\
& \quad \quad \quad ctrl\text{-stk}, \\
& \quad \quad \quad \text{push}(\text{tag}(\text{'nat}, \text{untag}(cc\text{-value}) - 1), temp\text{-stk}), \\
& \quad \quad \quad \text{translate-proc-list}(proc\text{-list}), \\
& \quad \quad \quad \text{list}(\text{list}(\text{'c-c}, cc\text{-value})), \\
& \quad \quad \quad \text{MG-MAX-CTRL-STK-SIZE}, \\
& \quad \quad \quad \text{MG-MAX-TEMP-STK-SIZE}, \\
& \quad \quad \quad \text{MG-WORD-SIZE}, \\
& \quad \quad \quad \text{'run}))
\end{aligned}$$

THEOREM: mg-integer-add-step-20-nonerror

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

```

^ (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)
^ small-integerp (iplus (0,
                        iplus (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                           mg-alist (mg-state)))))),
                        untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                           mg-alist (mg-state))))))),
                    MG-WORD-SIZE))
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)),
                    ctrl-stk,
                    push (tag ('nat,
                              untag (mg-cond-to-p-nat (cc (mg-state),
                                                         t-cond-list)) - 1),
                              rput (tag ('int,
                                        fix-small-integer (iplus (0,
                                                                iplus (untag (mg-to-p-simple-literal (caddr (assoc
                                                                                                                                           untag (mg-to-p-simple-literal (caddr (assoc
                                                                                                                                           MG-WORD-SIZE))),
                                                                                                                                           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-add-step-10-error

(($n \neq 0$)
 \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-add)
 \wedge ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
 \wedge ok-mg-def-plistp (*proc-list*)
 \wedge ok-mg-statep (*mg-state*, *r-cond-list*)
 \wedge (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*)
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
 \wedge user-defined-procp (*subr*, *proc-list*)
 \wedge listp (*ctrl-stk*)
 \wedge all-cars-unique (mg-alist (*mg-state*))
 \wedge signatures-match (mg-alist (*mg-state*), *name-alist*)
 \wedge mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
 \wedge no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
 \wedge normal (*mg-state*)
 \wedge (\neg small-integerp (iplus (0,
iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (*stmt*)),
mg-alist (*mg-state*))))),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (*stmt*)),
mg-alist (*mg-state*))))))),
MG-WORD-SIZE))))
 \rightarrow (p-step (p-state (tag ('pc, '(mg-integer-add . 5)),
push (p-frame (cons (cons ('ans,
value (car (call-actuals (*stmt*)),
bindings (top (*ctrl-stk*))),
cons (cons ('y,
value (cadr (call-actuals (*stmt*)),
bindings (top (*ctrl-stk*))),
cons (cons ('z,
value (caddr (call-actuals (*stmt*)),
bindings (top (*ctrl-stk*))),
'((t1 int 0))))),

```

tag ('pc,
    cons (subr, length (code (cinfo))
          + 4)),
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                          mg-alist (mg-state))))),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                          mg-alist (mg-state))))),
push ('(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-add . 6)),
push (p-frame (cons (cons ('ans,
                          value (car (call-actuals (stmt)),
                                      bindings (top (ctrl-stk))))),
                  cons (cons ('y,
                              value (cadr (call-actuals (stmt)),
                                      bindings (top (ctrl-stk))))),
                  cons (cons ('z,
                              value (caddr (call-actuals (stmt)),
                                      bindings (top (ctrl-stk))))),
                          '(t1 int 0))))),
tag ('pc,
    cons (subr, length (code (cinfo))
          + 4)),
ctrl-stk),
push (tag ('int,
          fix-small-integer (iplus (0,
                                   iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                          mg-alist (mg-state))))),
                                         untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                          mg-alist (mg-state))))),
                                         MG-WORD-SIZE))),
          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))

```

THEOREM: mg-integer-add-steps-11-15-error

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                             proc-list,
                             list (length (temp-stk),
                                     p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-add)
 ∧ 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-add
                                                         . 6)),
                                                         push (p-frame (cons (cons ('ans,
                                                                 value (car (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk)))),
                                                                 cons (cons ('y,
                                                                 value (cadr (call-actuals (st
                                                                 bindings (top (ctrl-st
                                                                 cons (cons ('z,
                                                                 value (caddr (call-act
                                                                 bindings (top (

```



```

                                                                    '((t1
                                                                      int
                                                                      0))))),
tag('pc,
  cons(subr,
    length(code(cinfo))
    + 4))),
ctrl-stk),
push(sum,
  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)) + 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, '(nat 1))),
  MG-MAX-CTRL-STK-SIZE,
  MG-MAX-TEMP-STK-SIZE,
  MG-WORD-SIZE,
  'run))

```

THEOREM: mg-integer-add-step-18-error

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep(stmt,
  proc-list,
  list(length(temp-stk),
    p-ctrl-stk-size(ctrl-stk))))
 ∧ (car(stmt) = 'predefined-proc-call-mg)
 ∧ (call-name(stmt) = 'mg-integer-add)
 ∧ 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)
^ (¬ small-integerp (iplus (0,
                           iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))),
                                   untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))))),
                               MG-WORD-SIZE)))
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)),
                    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-add-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-add)

```

```

^ 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-SUBTRACT                               ;;
;;                                                                                               ;;
;;                                                                                               ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

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

```

((car (stmt) = 'predefined-proc-call-mg)
  ∧ (call-name (stmt) = 'mg-integer-subtract)
  ∧ 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))
→ (int-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)))

```



```

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

```

THEOREM: mg-integer-subtract-step-4

```

((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-subtract)
  ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
  ∧ ok-mg-def-plistp (proc-list)
  ∧ ok-mg-statep (mg-state, r-cond-list)
  ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
            code2))
  ∧ user-defined-procp (subr, proc-list)
  ∧ listp (ctrl-stk)
  ∧ all-cars-unique (mg-alist (mg-state))
  ∧ signatures-match (mg-alist (mg-state), name-alist)
  ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
      bindings (top (ctrl-stk)),
      temp-stk)
  ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
  ∧ normal (mg-state)
  → (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 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-subtract . 0),
push (p-frame (cons (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons (cons ('y,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons (cons ('z,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
'((t1 int 0))))),
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-subtract-steps-5-9

```

((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-subtract)
^ 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-subtract
                                                    . 0)),
                                                    push (p-frame (cons (cons ('ans,
                                                                    value (car (call-actuals (stmt))),
                                                                    bindings (top (ctrl-stk))),
                                                                    cons (cons ('y,
                                                                    value (cadr (call-actuals (stmt))),
                                                                    bindings (top (ctrl-stk))),
                                                                    cons (cons ('z,
                                                                    value (caddr (call-actuals (stmt))),
                                                                    bindings (top (ctrl-stk))),
                                                                    '((t1
                                                                    int
                                                                    0))))),
                                                    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-subtract . 5)),
           push (p-frame (cons (cons ('ans,
                                     value (car (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                               cons (cons ('y,
                                     value (cadr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                               cons (cons ('z,
                                     value (caddr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                                     '((t1 int 0))))),
           tag ('pc,
               cons (subr, length (code (cinfo)
                                     + 4))),
           ctrl-stk),
  push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                             mg-alist (mg-state))))),
  push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                             mg-alist (mg-state))))),
  push ('(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))

```

THEOREM: mg-integer-subtract-step-10-nonerror

```

((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-subtract)
 ∧ 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)
^ small-integerp (idifference (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                           mg-alist (mg-state))))),
                              iplus (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                           mg-alist (mg-state))))),
                                      0)),
                  MG-WORD-SIZE))
→ (p-step (p-state (tag ('pc, '(mg-integer-subtract . 5)),
                    push (p-frame (cons (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk)))),
                                          cons (cons ('y,
                                              value (cadr (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk)))),
                                          cons (cons ('z,
                                              value (caddr (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk)))),
                                              '((t1 int 0))))),
                    tag ('pc,
                        cons (subr, length (code (cinfo))
                              + 4))),
                    ctrl-stk),
    push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                mg-alist (mg-state))))),
    push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                mg-alist (mg-state))))),
    push ('(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-subtract . 6),
           push (p-frame (cons (cons ('ans,
                                     value (car (call-actuals (stmt))),
                                     bindings (top (ctrl-stk)))))
                    cons (cons ('y,
                                value (cadr (call-actuals (stmt))),
                                bindings (top (ctrl-stk)))))
                    cons (cons ('z,
                                value (caddr (call-actuals (stmt))),
                                bindings (top (ctrl-stk)))))
                    '(t1 int 0))))),
           tag ('pc,
               cons (subr, length (code (cinfo))
                    + 4))),
           ctrl-stk),
push (tag ('int,
         fix-small-integer (idifference (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))),
                                          mg-alist))))
                                iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))),
                                          mg-alist))))
                                0))),
      MG-WORD-SIZE)),
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))

```

THEOREM: mg-integer-subtract-steps-11-17-nonerror

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

```

                                list (length (temp-stk),
                                        p-ctrl-stk-size (ctrl-stk)))
^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-integer-subtract)
^ 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-step (p-state (tag ('pc,
                                                                ' (mg-integer-subtract
                                                                    . 6)),
                                                                push (p-frame (cons (cons ('ans,
                                                                    value (car (call-ac
                                                                    bindings (t
                                                                cons (cons ('y,
                                                                    value (cadr (
                                                                    bindi
                                                                cons (cons ('z,
                                                                    value (
                                                                    '( (t1
                                                                    int
                                                                    0))))),
                                                                tag ('pc,
                                                                    cons (subr,
                                                                    length (code (cinfo
                                                                    + 4))),
                                                                ctrl-stk),
                                                                push (diff,
                                                                    push (tag ('bool,
                                                                    'f),
                                                                    map-down-values (mg-alist (mg-
                                                                    bindings (top

```

```

                                                                    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)) + 4)),
           ctrl-stk,
           rput (diff,
                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-subtract-push-cc

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                             proc-list,
                             list (length (temp-stk),
                                     p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-subtract)
 ∧ 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)
 ∧ normal (mg-state))

```

$$\begin{aligned}
&\rightarrow (\text{p-step } (\text{p-state } (\text{tag } ('pc, \text{cons } (subr, \text{length } (\text{code } (cinfo)) + 4)), \\
&\quad ctrl-stk, \\
&\quad temp-stk, \\
&\quad \text{translate-proc-list } (proc-list), \\
&\quad \text{list } (\text{list } ('c-c, cc-value)), \\
&\quad \text{MG-MAX-CTRL-STK-SIZE}, \\
&\quad \text{MG-MAX-TEMP-STK-SIZE}, \\
&\quad \text{MG-WORD-SIZE}, \\
&\quad 'run)) \\
&= \text{p-state } (\text{tag } ('pc, \text{cons } (subr, \text{length } (\text{code } (cinfo)) + 5)), \\
&\quad ctrl-stk, \\
&\quad \text{push } (cc-value, temp-stk), \\
&\quad \text{translate-proc-list } (proc-list), \\
&\quad \text{list } (\text{list } ('c-c, cc-value)), \\
&\quad \text{MG-MAX-CTRL-STK-SIZE}, \\
&\quad \text{MG-MAX-TEMP-STK-SIZE}, \\
&\quad \text{MG-WORD-SIZE}, \\
&\quad 'run))
\end{aligned}$$

THEOREM: mg-integer-subtract-sub1-cc

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

```

MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))
= p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)),
ctrl-stk,
push (tag ('nat, untag (cc-value) - 1), temp-stk),
translate-proc-list (proc-list),
list (list ('c-c, cc-value)),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-integer-subtract-step-20-nonerror

```

((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-subtract)
 ∧ 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)
 ∧ small-integerp (idifference (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))),
ipius (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))),
0)),
MG-WORD-SIZE))
→ (p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)),

```



```

ctrl-stk,
push (tag ('nat,
          untag (mg-cond-to-p-nat (cc (mg-state),
                                     t-cond-list)) - 1),
      rput (tag ('int,
                 fix-small-integer (idifference (untag (mg-to-p-simple-literal (caddr (assoc (
                                                                                               iplus (untag (mg-to-p-simple-literal (caddr (
                                                                                               0))),
                                                                                               MG-WORD-SIZE))),
          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))),
                                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-subtract-step-10-error

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                              proc-list,
                              list (length (temp-stk),
                                      p-ctrl-stk-size (ctrl-stk)))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-integer-subtract)
 ∧ 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)
^ (¬ small-integerp (idifference (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))),
                                iplus (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                              mg-alist (mg-state))))),
                                0)),
    MG-WORD-SIZE)))
→ (p-step (p-state (tag ('pc, '(mg-integer-subtract . 5)),
                    push (p-frame (cons (cons ('ans,
                                              value (car (call-actuals (stmt)),
                                              bindings (top (ctrl-stk)))),
                                        cons (cons ('y,
                                                  value (cadr (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk)))),
                                        cons (cons ('z,
                                                  value (caddr (call-actuals (stmt)),
                                                  bindings (top (ctrl-stk))),
                                                  '((t1 int 0))))),
                                tag ('pc,
                                    cons (subr, length (code (cinfo)
                                                                + 4))),
                                ctrl-stk),
                    push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                    push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                    push ('(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))

```



```

^ 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-subtract
                                                           . 6)),
                                                         push (p-frame (cons (cons ('ans,
                                                                 value (car (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk))),
                                                                 cons (cons ('y,
                                                                 value (cadr (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk))),
                                                                 cons (cons ('z,
                                                                 value (caddr (call-actuals (stmt)),
                                                                 bindings (top (ctrl-stk))),
                                                                 '((t1
                                                                 int
                                                                 0))))),
                                                         tag ('pc,
                                                         cons (subr,
                                                         length (code (cinfo))
                                                         + 4))),
                                                         ctrl-stk),
                                                         push (diff,
                                                         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,

```

$$\begin{aligned}
& \text{MG-WORD-SIZE,} \\
& \text{'run}}))))) \\
= & \text{p-state (tag ('pc, cons (subr, length (code (cinfo)) + 4),} \\
& \text{ctrl-stk,} \\
& \text{map-down-values (mg-alist (mg-state),} \\
& \text{bindings (top (ctrl-stk)),} \\
& \text{temp-stk),} \\
& \text{translate-proc-list (proc-list),} \\
& \text{list (list ('c-c, 'nat 1)),} \\
& \text{MG-MAX-CTRL-STK-SIZE,} \\
& \text{MG-MAX-TEMP-STK-SIZE,} \\
& \text{MG-WORD-SIZE,} \\
& \text{'run))}
\end{aligned}$$

THEOREM: mg-integer-subtract-step-18-error

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep (stmt,} \\
& \quad \text{proc-list,} \\
& \quad \text{list (length (temp-stk),} \\
& \quad \text{p-ctrl-stk-size (ctrl-stk))})) \\
& \wedge (\text{car (stmt) = 'predefined-proc-call-mg}) \\
& \wedge (\text{call-name (stmt) = 'mg-integer-subtract}) \\
& \wedge \text{ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)} \\
& \wedge \text{ok-mg-def-plistp (proc-list)} \\
& \wedge \text{ok-mg-statep (mg-state, r-cond-list)} \\
& \wedge (\text{code (translate-def-body (assoc (subr, proc-list), proc-list))} \\
& \quad = \text{append (code (translate (cinfo, t-cond-list, stmt, proc-list)),} \\
& \quad \text{code2})) \\
& \wedge \text{user-defined-procp (subr, proc-list)} \\
& \wedge \text{listp (ctrl-stk)} \\
& \wedge \text{all-cars-unique (mg-alist (mg-state))} \\
& \wedge \text{signatures-match (mg-alist (mg-state), name-alist)} \\
& \wedge \text{mg-vars-list-ok-in-p-state (mg-alist (mg-state),} \\
& \quad \text{bindings (top (ctrl-stk)),} \\
& \quad \text{temp-stk)} \\
& \wedge \text{no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))} \\
& \wedge \text{normal (mg-state)} \\
& \wedge (\neg \text{small-integerp (idifference (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),} \\
& \quad \text{mg-alist (mg-state)))))),} \\
& \quad \text{iplus (untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)} \\
& \quad \text{mg-alist (mg-state)))))),} \\
& \quad \text{0)),} \\
& \quad \text{MG-WORD-SIZE))}) \\
\rightarrow & (\text{p-step (p-state (tag ('pc, cons (subr, length (code (cinfo)) + 6)),}
\end{aligned}$$

```

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),

```

```

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-subtract-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-subtract)
^ 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-BOOLEAN-OR
;;
;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

EVENT: Disable or-bool.

EVENT: Disable p-objectp-type.

THEOREM: mg-boolean-or-args-have-simple-mg-type-refps

```

((car (stmt) = 'predefined-proc-call-mg)
 & (call-name (stmt) = 'mg-boolean-or)
 & 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))
    & boolean-identifierp (cadr (call-actuals (stmt)),
                           mg-alist (mg-state))
    & boolean-identifierp (caddr (call-actuals (stmt)),
                           mg-alist (mg-state)))

```

THEOREM: mg-boolean-or-args-definedp

```

((car (stmt) = 'predefined-proc-call-mg)
 & (call-name (stmt) = 'mg-boolean-or)
 & 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-boolean-or-steps-1-3

```

((n ≠ 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-boolean-or)
 $\wedge$  ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 $\wedge$  ok-mg-def-plistp (proc-list)
 $\wedge$  ok-mg-statep (mg-state, r-cond-list)
 $\wedge$  (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
                code2))
 $\wedge$  user-defined-procp (subr, proc-list)
 $\wedge$  listp (ctrl-stk)
 $\wedge$  all-cars-unique (mg-alist (mg-state))
 $\wedge$  signatures-match (mg-alist (mg-state), name-alist)
 $\wedge$  mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                                       bindings (top (ctrl-stk)),
                                       temp-stk)
 $\wedge$  no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 $\wedge$  normal (mg-state)
 $\rightarrow$  (p-step (p-step (p-step (map-down (mg-state,
                                         proc-list,
                                         ctrl-stk,
                                         temp-stk,
                                         tag ('pc,
                                             cons (subr, length (code (cinfo)))),
                                             t-cond-list))))
          = p-state (tag ('pc, cons (subr, length (code (cinfo) + 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))

THEOREM: mg-boolean-or-step-4

(($n \neq 0$)
 \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-boolean-or)
 \wedge ok-mg-statement (*stmt*, *r-cond-list*, *name-alist*, *proc-list*)
 \wedge ok-mg-def-plistp (*proc-list*)
 \wedge ok-mg-statement (*mg-state*, *r-cond-list*)
 \wedge (code (translate-def-body (assoc (*subr*, *proc-list*), *proc-list*))
= append (code (translate (*cinfo*, *t-cond-list*, *stmt*, *proc-list*)),
code2))
 \wedge user-defined-procp (*subr*, *proc-list*)
 \wedge listp (*ctrl-stk*)
 \wedge all-cars-unique (mg-alist (*mg-state*))
 \wedge signatures-match (mg-alist (*mg-state*), *name-alist*)
 \wedge mg-vars-list-ok-in-p-state (mg-alist (*mg-state*),
bindings (top (*ctrl-stk*)),
temp-stk)
 \wedge no-p-aliasing (bindings (top (*ctrl-stk*)), mg-alist (*mg-state*))
 \wedge normal (*mg-state*)
 \rightarrow (p-step (p-state (tag ('pc, cons (*subr*, length (code (*cinfo*)) + 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-boolean-or . 0)),
           push (p-frame (list (cons ('ans,
                                     value (car (call-actuals (stmt)),
                                     bindings (top (ctrl-stk))))),
                        cons ('b1,
                              value (cadr (call-actuals (stmt)),
                              bindings (top (ctrl-stk))))),
                        cons ('b2,
                              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-boolean-or-steps-5-8

```

((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-boolean-or)
 ∧ 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-state (tag ('pc,
                                         '(mg-boolean-or . 0)),
                                         push (p-frame (list (cons ('ans,
                                                                    value (car (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk)))),
                                                                    cons ('b1,
                                                                    value (cadr (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk)))),
                                                                    cons ('b2,
                                                                    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-boolean-or . 4)),
           push (p-frame (list (cons ('ans,
                                     value (car (call-actuals (stmt)),
                                     bindings (top (ctrl-stk)))),
                                     cons ('b1,
                                     value (cadr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk)))),
                                     cons ('b2,
                                     value (caddr (call-actuals (stmt)),
                                     bindings (top (ctrl-stk)))))),
           tag ('pc,
               cons (subr, length (code (cinfo))
                   + 4))),

```

```

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

```

THEOREM: mg-boolean-or-steps-9-11

```

((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-boolean-or)
 ^ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ^ ok-mg-def-plistp (proc-list)
 ^ ok-mg-statep (mg-state, r-cond-list)
 ^ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
               code2))
 ^ user-defined-procp (subr, proc-list)
 ^ listp (ctrl-stk)
 ^ all-cars-unique (mg-alist (mg-state))
 ^ signatures-match (mg-alist (mg-state), name-alist)
 ^ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                               bindings (top (ctrl-stk)),
                               temp-stk)
 ^ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ^ normal (mg-state))
 → (p-step (p-step (p-step (p-state (tag ('pc, ' (mg-boolean-or . 4)),
                                       push (p-frame (list (cons ('ans,
                                                                value (car (call-actuals (stmt)),
                                                                bindings (top (ctrl-stk))))),
                                                                cons ('b1,

```

```

value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk))),
cons ('b2,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
tag ('pc,
cons (subr,
length (code (cinfo))
+ 4)),
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state))))),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state))))),
map-down-values (mg-alist (mg-state),
bindings (top (ctrl-stk)),
temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
mg-cond-to-p-nat (cc (mg-state),
t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run)))
= p-state (tag ('pc, '(mg-boolean-or . 7)),
push (p-frame (list (cons ('ans,
value (car (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
cons ('b1,
value (cadr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
cons ('b2,
value (caddr (call-actuals (stmt)),
bindings (top (ctrl-stk)))))
tag ('pc,
cons (subr, length (code (cinfo))
+ 4)),
ctrl-stk),
rput (tag ('bool,
or-bool (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
mg-alist (mg-state))))),
untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
mg-alist (mg-state)))))))))

```



```

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: boolean-literal-simple-typed-literalp
boolean-literalp (*x*) → simple-typed-literalp (*x*, 'boolean-mg)

THEOREM: mg-boolean-or-step-12

```

((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-boolean-or)
 ∧ 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-boolean-or . 7)),
                       push (p-frame (list (cons ('ans,
                                                value (car (call-actuals (stmt)),
                                                                    bindings (top (ctrl-stk))))),
                                                cons ('b1,

```

```

value (cadr (call-actuals (stmt)),
      bindings (top (ctrl-stk))),
cons ('b2,
      value (caddr (call-actuals (stmt)),
            bindings (top (ctrl-stk))),
tag ('pc,
     cons (subr, length (code (cinfo))
          + 4))),
ctrl-stk),
rput (tag ('bool,
          or-bool (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                             mg-alist (mg-state))))),
                  untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                             mg-alist (mg-state))))))),
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-boolean-or-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-boolean-or)
 ∧ 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 (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-BOOLEAN-AND           ;;
;;                                                                                               ;;
;;                                                                                               ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

```

EVENT: Disable and-bool.

EVENT: Disable p-objectp-type.

THEOREM: mg-boolean-and-args-have-simple-mg-type-refps

```

((car (stmt) = 'predefined-proc-call-mg)
 & (call-name (stmt) = 'mg-boolean-and)
 & 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))
    & boolean-identifierp (cadr (call-actuals (stmt))),

```

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

THEOREM: mg-boolean-and-args-definedp

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

THEOREM: mg-boolean-and-steps-1-3

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

```

= 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))

```

THEOREM: mg-boolean-and-step-4

```

((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-boolean-and)
 ∧ 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, 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-boolean-and . 0)),
           push (p-frame (list (cons ('ans,
                                     value (car (call-actuals (stmt)),
                                               bindings (top (ctrl-stk))),
                                     cons ('b1,
                                           value (cadr (call-actuals (stmt)),
                                               bindings (top (ctrl-stk))),
                                     cons ('b2,
                                           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-boolean-and-steps-5-8
 $((n \neq 0)$
 $\wedge (\neg \text{resources-inadequatep} (stmt,$


```

                                proc-list,
                                list (length (temp-stk),
                                        p-ctrl-stk-size (ctrl-stk)))
^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-boolean-and)
^ 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-state (tag ('pc,
                                                '(mg-boolean-and . 0)),
                                                push (p-frame (list (cons ('ans,
                                                                            value (car (call-actuals (stmt)),
                                                                              bindings (top (ctrl-stk))),
                                                                              cons ('b1,
                                                                              value (cadr (call-actuals (stmt)),
                                                                              bindings (top (ctrl-stk))),
                                                                              cons ('b2,
                                                                              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-boolean-and . 4)),
  push (p-frame (list (cons ('ans,
    value (car (call-actuals (stmt)),
      bindings (top (ctrl-stk))))),
    cons ('b1,
      value (cadr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    cons ('b2,
      value (caddr (call-actuals (stmt)),
        bindings (top (ctrl-stk))))),
    tag ('pc,
      cons (subr, length (code (cinfo)
        + 4))),
    ctrl-stk),
  push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
    mg-alist (mg-state))))),
  push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
    mg-alist (mg-state))))),
  map-down-values (mg-alist (mg-state),
    bindings (top (ctrl-stk)),
    temp-stk)),
  translate-proc-list (proc-list),
  list (list ('c-c,
    mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
  MG-MAX-CTRL-STK-SIZE,
  MG-MAX-TEMP-STK-SIZE,
  MG-WORD-SIZE,
  'run))

```

THEOREM: mg-boolean-and-steps-9-11

```

((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-boolean-and)
 ∧ 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-step (p-step (p-state (tag ('pc, '(mg-boolean-and . 4)),
                                     push (p-frame (list (cons ('ans,
                                                                value (car (call-actuals (stmt)),
                                                                bindings (top (ctrl-stk)))),
                                                                cons ('b1,
                                                                value (cadr (call-actuals (stmt)),
                                                                bindings (top (ctrl-stk)))),
                                                                cons ('b2,
                                                                value (caddr (call-actuals (stmt)),
                                                                bindings (top (ctrl-stk))))),
                                     tag ('pc,
                                         cons (subr,
                                               length (code (cinfo))
                                               + 4))),
                                     ctrl-stk),
                                     push (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                                     push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                                     map-down-values (mg-alist (mg-state),
                                                     bindings (top (ctrl-stk)),
                                                     temp-stk))),
                                     translate-proc-list (proc-list),
                                     list (list ('c-c,
                                                 mg-cond-to-p-nat (cc (mg-state),
                                                                     t-cond-list))),
                                     MG-MAX-CTRL-STK-SIZE,
                                     MG-MAX-TEMP-STK-SIZE,
                                     MG-WORD-SIZE,
                                     'run))))))
= p-state (tag ('pc, '(mg-boolean-and . 7)),
          push (p-frame (list (cons ('ans,
                                     value (car (call-actuals (stmt)),

```

```

bindings (top (ctrl-stk))),
cons ('b1,
      value (cadr (call-actuals (stmt)),
              bindings (top (ctrl-stk))),
cons ('b2,
      value (caddr (call-actuals (stmt)),
              bindings (top (ctrl-stk))),
tag ('pc,
     cons (subr, length (code (cinfo)
                             + 4))),
ctrl-stk),
rput (tag ('bool,
           and-bool (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))),
                     untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt)),
                                                                mg-alist (mg-state))))))),
      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-boolean-and-step-12

```

((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-boolean-and)
 ∧ 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-boolean-and . 7)),
                    push (p-frame (list (cons ('ans,
                                             value (car (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))),
                                             cons ('b1,
                                             value (cadr (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))),
                                             cons ('b2,
                                             value (caddr (call-actuals (stmt)),
                                             bindings (top (ctrl-stk))))),
                    tag ('pc,
                        cons (subr, length (code (cinfo))
                            + 4))),
                    ctrl-stk),
    rput (tag ('bool,
              and-bool (untag (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt))
                                                                    mg-alist (mg-state))))),
                          untag (mg-to-p-simple-literal (caddr (assoc (caddr (call-actuals (stmt))
                                                                    mg-alist (mg-state))))))),
          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-boolean-and-exact-time-lemma

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep} (stmt, \\
& \qquad \qquad \qquad \text{proc-list}, \\
& \qquad \qquad \qquad \text{list} (\text{length} (temp-stk), \\
& \qquad \qquad \qquad \text{p-ctrl-stk-size} (ctrl-stk)))) \\
& \wedge (\text{car} (stmt) = \text{'predefined-proc-call-mg}) \\
& \wedge (\text{call-name} (stmt) = \text{'mg-boolean-and}) \\
& \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)), \\
& \qquad \qquad \qquad \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), \\
& \qquad \qquad \qquad \text{bindings} (\text{top} (ctrl-stk)), \\
& \qquad \qquad \qquad \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} (\text{map-down} (mg-state, \\
& \qquad \qquad \qquad \text{proc-list}, \\
& \qquad \qquad \qquad \text{ctrl-stk}, \\
& \qquad \qquad \qquad \text{temp-stk}, \\
& \qquad \qquad \qquad \text{tag} (\text{'pc}, \text{cons} (subr, \text{length} (\text{code} (cinfo))), \\
& \qquad \qquad \qquad \text{t-cond-list}), \\
& \qquad \qquad \text{clock} (stmt, proc-list, mg-state, n)) \\
& = \text{p-state} (\text{tag} (\text{'pc}, \\
& \qquad \qquad \text{cons} (subr, \\
& \qquad \qquad \text{if normal} (\text{mg-meaning-r} (stmt, \\
& \qquad \qquad \qquad \text{proc-list}, \\
& \qquad \qquad \qquad \text{mg-state}, \\
& \qquad \qquad \qquad n, \\
& \qquad \qquad \qquad \text{list} (\text{length} (temp-stk), \\
& \qquad \qquad \qquad \text{p-ctrl-stk-size} (ctrl-stk)))) \\
& \qquad \qquad \text{then length} (\text{code} (\text{translate} (cinfo, \\
& \qquad \qquad \qquad \text{t-cond-list}, \\
& \qquad \qquad \qquad \text{stmt}, \\
& \qquad \qquad \qquad \text{proc-list})) \\
& \qquad \qquad \text{else find-label} (\text{fetch-label} (\text{cc} (\text{mg-meaning-r} (stmt, \\
& \qquad \qquad \qquad \text{proc-list},
\end{aligned}$$


```

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

```

THEOREM: mg-boolean-not-step-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-boolean-not)
 ∧ 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))

```

$$\begin{aligned}
& \rightarrow (\text{p-step } (\text{p-state } (\text{tag } ('pc, \text{cons } (subr, \text{length } (\text{code } (cinfo)) + 2)), \\
& \quad ctrl-stk, \\
& \quad \text{push } (\text{value } (\text{cadr } (\text{call-actuals } (stmt))), \\
& \quad \quad \text{bindings } (\text{top } (ctrl-stk))), \\
& \quad \text{push } (\text{value } (\text{car } (\text{call-actuals } (stmt))), \\
& \quad \quad \text{bindings } (\text{top } (ctrl-stk))), \\
& \quad \text{map-down-values } (\text{mg-alist } (mg-state), \\
& \quad \quad \text{bindings } (\text{top } (ctrl-stk)), \\
& \quad \quad \text{temp-stk}))), \\
& \quad \text{translate-proc-list } (proc-list), \\
& \quad \text{list } (\text{list } ('c-c, \\
& \quad \quad \text{mg-cond-to-p-nat } (\text{cc } (mg-state), t-cond-list))), \\
& \quad \text{MG-MAX-CTRL-STK-SIZE,} \\
& \quad \text{MG-MAX-TEMP-STK-SIZE,} \\
& \quad \text{MG-WORD-SIZE,} \\
& \quad 'run)) \\
& = \text{p-state } (\text{tag } ('pc, 'mg-boolean-not . 0), \\
& \quad \text{push } (\text{p-frame } (\text{list } (\text{cons } ('ans, \\
& \quad \quad \text{value } (\text{car } (\text{call-actuals } (stmt))), \\
& \quad \quad \text{bindings } (\text{top } (ctrl-stk))), \\
& \quad \quad \text{cons } ('b1, \\
& \quad \quad \text{value } (\text{cadr } (\text{call-actuals } (stmt))), \\
& \quad \quad \text{bindings } (\text{top } (ctrl-stk))), \\
& \quad \quad \text{tag } ('pc, \\
& \quad \quad \text{cons } (subr, \text{length } (\text{code } (cinfo)) \\
& \quad \quad \quad + 3))), \\
& \quad \quad 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}), \\
& \quad \text{translate-proc-list } (proc-list), \\
& \quad \text{list } (\text{list } ('c-c, \\
& \quad \quad \text{mg-cond-to-p-nat } (\text{cc } (mg-state), t-cond-list))), \\
& \quad \text{MG-MAX-CTRL-STK-SIZE,} \\
& \quad \text{MG-MAX-TEMP-STK-SIZE,} \\
& \quad \text{MG-WORD-SIZE,} \\
& \quad 'run))
\end{aligned}$$

THEOREM: mg-boolean-not-steps-4-5

$$\begin{aligned}
& ((n \neq 0) \\
& \wedge (\neg \text{resources-inadequatep } (stmt, \\
& \quad \quad \quad \text{proc-list,} \\
& \quad \quad \quad \text{list } (\text{length } (temp-stk), \\
& \quad \quad \quad \text{p-ctrl-stk-size } (ctrl-stk))))
\end{aligned}$$

```

^ (car (stmt) = 'predefined-proc-call-mg)
^ (call-name (stmt) = 'mg-boolean-not)
^ 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-state (tag ('pc, '(mg-boolean-not . 0)),
                              push (p-frame (list (cons ('ans,
                                                         value (car (call-actuals (stmt))),
                                                         bindings (top (ctrl-stk))))),
                                                         cons ('b1,
                                                         value (cadr (call-actuals (stmt))),
                                                         bindings (top (ctrl-stk))))),
                              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-boolean-not . 2)),
          push (p-frame (list (cons ('ans,
                                    value (car (call-actuals (stmt))),
                                    bindings (top (ctrl-stk))))),

```

```

cons ('b1,
      value (cadr (call-actuals (stmt)),
             bindings (top (ctrl-stk)))))
tag ('pc,
     cons (subr, length (code (cinfo)
                             + 3))),
ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                           mg-alist (mg-state)))))
     map-down-values (mg-alist (mg-state),
                      bindings (top (ctrl-stk)),
                      temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
           mg-cond-to-p-nat (cc (mg-state), t-cond-list))),
MG-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

```

THEOREM: mg-boolean-not-steps-6-8

```

((n ≠ 0)
 ∧ (¬ resources-inadequatep (stmt,
                             proc-list,
                             list (length (temp-stk),
                                     p-ctrl-stk-size (ctrl-stk))))
 ∧ (car (stmt) = 'predefined-proc-call-mg)
 ∧ (call-name (stmt) = 'mg-boolean-not)
 ∧ ok-mg-statement (stmt, r-cond-list, name-alist, proc-list)
 ∧ ok-mg-def-plistp (proc-list)
 ∧ ok-mg-statep (mg-state, r-cond-list)
 ∧ (code (translate-def-body (assoc (subr, proc-list), proc-list))
      = append (code (translate (cinfo, t-cond-list, stmt, proc-list)),
               code2))
 ∧ user-defined-procp (subr, proc-list)
 ∧ listp (ctrl-stk)
 ∧ all-cars-unique (mg-alist (mg-state))
 ∧ signatures-match (mg-alist (mg-state), name-alist)
 ∧ mg-vars-list-ok-in-p-state (mg-alist (mg-state),
                              bindings (top (ctrl-stk)),
                              temp-stk)
 ∧ no-p-aliasing (bindings (top (ctrl-stk)), mg-alist (mg-state))
 ∧ normal (mg-state))
 → (p-step (p-step (p-step (p-state (tag ('pc, '(mg-boolean-not . 2)),

```

```

push (p-frame (list (cons ('ans,
                           value (car (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))))
                    cons ('b1,
                           value (cadr (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))))
      tag ('pc,
           cons (subr,
                 length (code (cinfo))
                       + 3))),
      ctrl-stk),
push (mg-to-p-simple-literal (caddr (assoc (cadr (call-actuals (stmt)),
                                           mg-alist (mg-state))))),
      map-down-values (mg-alist (mg-state),
                       bindings (top (ctrl-stk)),
                       temp-stk)),
translate-proc-list (proc-list),
list (list ('c-c,
           mg-cond-to-p-nat (cc (mg-state),
                               t-cond-list))),
      MG-MAX-CTRL-STK-SIZE,
      MG-MAX-TEMP-STK-SIZE,
      MG-WORD-SIZE,
      'run))))
= p-state (tag ('pc, ' (mg-boolean-not . 5)),
           push (p-frame (list (cons ('ans,
                                       value (car (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))))
                                cons ('b1,
                                       value (cadr (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))))
              tag ('pc,
                   cons (subr, length (code (cinfo))
                         + 3))),
                ctrl-stk),
           rput (tag ('bool,
                     not-bool (untag (mg-to-p-simple-literal (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-boolean-not-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-boolean-not)
 ∧ 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-boolean-not . 5)),
                    push (p-frame (list (cons ('ans,
                                             value (car (call-actuals (stmt)),
                                             bindings (top (ctrl-stk)))),
                                         cons ('b1,
                                             value (cadr (call-actuals (stmt)),
                                             bindings (top (ctrl-stk))))),
                                tag ('pc,
                                    cons (subr, length (code (cinfo)
                                                                + 3))),
                                ctrl-stk),
                    rput (tag ('bool,
                                not-bool (untag (mg-to-p-simple-literal (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-boolean-not-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-boolean-not)
 ∧ 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-MAX-CTRL-STK-SIZE,
MG-MAX-TEMP-STK-SIZE,
MG-WORD-SIZE,
'run))

t-cond-list))),
mg-state,
n,
list (length (temp-stk),
p-ctrl-stk-size (ctrl-stk))),

```

EVENT: Make the library "c-predefined3".

Index

- all-cars-unique, 2–4, 6, 8, 10–12, 14, 16, 18, 20, 22, 23, 25, 27, 29–32, 35, 37, 38, 40, 43–45, 47, 49, 52, 54, 55, 57, 59, 61, 63, 65, 66, 68, 69, 71, 73
- and-bool, 60, 61
- bindings, 2–9, 12–30, 32–41, 43–74
- boolean-identifierp, 42, 53, 54, 65
- boolean-literal-simple-typed-lite
ralp, 49
- boolean-literalp, 49
- call-actuals, 1–9, 12, 14–16, 18, 21–30, 32, 33, 35–38, 42–50, 53–61, 65–72
- call-name, 1, 2, 4, 6, 8, 10, 11, 14, 16, 17, 19, 21–23, 25, 26, 29–32, 34, 36, 38, 40, 42–45, 47, 49, 51, 53–55, 57, 58, 60, 63, 65, 66, 68, 69, 71, 73
- cc, 2–9, 12, 13, 15–17, 19–21, 23–26, 28, 30, 33–37, 39–53, 55–62, 64, 66–75
- clock, 20, 41, 52, 63, 74
- code, 1–41, 43–52, 54–74
- definedp, 1, 22, 42, 54, 65
- fetch-label, 13, 19, 20, 34, 39, 41, 51, 52, 62, 64, 72, 74
- find-label, 13, 19, 21, 34, 39, 41, 51, 53, 62, 64, 72, 74
- fix-small-integer, 8, 12, 15, 28, 33, 36
- idifference, 27, 28, 32, 33, 35, 36, 38
- int-identifierp, 1, 21
- iplus, 6, 8, 12, 14, 15, 18, 27, 28, 32, 33, 35, 36, 38
- label-alist, 13, 19, 20, 34, 39, 41, 51, 52, 62, 64, 72, 74
- length, 1–75
- map-down, 2, 20, 22, 41, 43, 52, 54, 63, 66, 74
- map-down-values, 2–9, 12, 13, 15–19, 21, 23–28, 30, 33–41, 43–51, 53, 55–62, 64, 66–70, 72–74
- mg-alist, 1–74
- mg-boolean-and-args-definedp, 54
- mg-boolean-and-args-have-simple
-mg-type-refps, 53
- mg-boolean-and-exact-time-lemma, 63
- mg-boolean-and-step-12, 60
- mg-boolean-and-step-4, 55
- mg-boolean-and-steps-1-3, 54
- mg-boolean-and-steps-5-8, 56
- mg-boolean-and-steps-9-11, 58
- mg-boolean-not-args-definedp, 65
- mg-boolean-not-args-have-simple
-mg-type-refps, 65
- mg-boolean-not-exact-time-lemma, 73
- mg-boolean-not-step-3, 66
- mg-boolean-not-step-9, 71
- mg-boolean-not-steps-1-2, 65
- mg-boolean-not-steps-4-5, 67
- mg-boolean-not-steps-6-8, 69
- mg-boolean-or-args-definedp, 42
- mg-boolean-or-args-have-simple
-mg-type-refps, 42
- mg-boolean-or-exact-time-lemma, 51
- mg-boolean-or-step-12, 49
- mg-boolean-or-step-4, 44
- mg-boolean-or-steps-1-3, 42
- mg-boolean-or-steps-5-8, 45
- mg-boolean-or-steps-9-11, 47
- mg-cond-to-p-nat, 2–9, 12, 13, 15–17, 19, 21, 23–26, 28, 30,

33–37, 40, 42–51, 53, 55–62, 64, 66–73, 75
 mg-integer-add-args-definedp, 1
 mg-integer-add-args-have-simple
 -mg-type-refps, 1
 mg-integer-add-exact-time-lemma, 19
 mg-integer-add-push-cc, 10
 mg-integer-add-step-10-error, 14
 mg-integer-add-step-10-nonerror, 6
 mg-integer-add-step-18-error, 17
 mg-integer-add-step-20-nonerror, 11
 mg-integer-add-step-4, 2
 mg-integer-add-steps-1-3, 1
 mg-integer-add-steps-11-15-erro
 r, 16
 mg-integer-add-steps-11-17-none
 rror, 8
 mg-integer-add-steps-5-9, 4
 mg-integer-add-sub1-cc, 10
 mg-integer-subtract-args-define
 dp, 22
 mg-integer-subtract-args-have-si
 mple-mg-type-refps, 21
 mg-integer-subtract-exact-time-
 lemma, 40
 mg-integer-subtract-push-cc, 30
 mg-integer-subtract-step-10-err
 or, 34
 mg-integer-subtract-step-10-none
 rror, 26
 mg-integer-subtract-step-18-err
 or, 38
 mg-integer-subtract-step-20-none
 rror, 32
 mg-integer-subtract-step-4, 23
 mg-integer-subtract-steps-1-3, 22
 mg-integer-subtract-steps-11-15
 -error, 36
 mg-integer-subtract-steps-11-17
 -nonerror, 28
 mg-integer-subtract-steps-5-9, 24
 mg-integer-subtract-sub1-cc, 31
 mg-max-ctrl-stk-size, 2–13, 15–19, 21,
 23, 24, 26, 28, 30–40, 42–
 51, 53, 55–62, 64, 66–73,
 75
 mg-max-temp-stk-size, 2–12, 14–19,
 21, 23, 24, 26, 28, 30–40,
 42–51, 53, 55, 56, 58–62,
 64, 66–73, 75
 mg-meaning-r, 13, 18–21, 33, 34, 39–
 42, 50–53, 62–64, 72–75
 mg-to-p-simple-literal, 5–8, 12, 14,
 15, 18, 26–28, 32, 33, 35,
 36, 38, 47, 48, 50, 58–61,
 69–71
 mg-vars-list-ok-in-p-state, 2–4, 6, 8,
 12, 14, 16, 18, 20, 22, 23,
 25, 27, 29, 32, 35, 37, 38,
 40, 43, 44, 46, 47, 49, 52,
 54, 55, 57, 59, 61, 63, 65,
 66, 68, 69, 71, 73
 mg-word-size, 2–12, 14–19, 21, 23,
 24, 26–28, 30–36, 38–40, 42,
 44–51, 53, 55, 56, 58–62,
 64, 66–73, 75
 no-p-aliasing, 2–4, 6, 8, 12, 14, 16,
 18, 20, 22, 23, 25, 27, 29,
 32, 35, 37, 38, 40, 43, 44,
 46, 47, 49, 52, 54, 55, 57,
 59, 61, 63, 65, 66, 68, 69,
 71, 73
 normal, 2–4, 6, 8, 10–14, 16, 18, 20,
 22, 23, 25, 27, 29–33, 35,
 37–41, 43, 44, 46, 47, 49,
 50, 52, 54, 55, 57, 59, 61–
 63, 65, 66, 68, 69, 71–74
 not-bool, 70, 71
 ok-mg-def-plistp, 1, 2, 4, 6, 8, 10,
 11, 14, 16, 17, 20, 22, 23,
 25, 27, 29–32, 34, 36, 38,
 40, 43–45, 47, 49, 51, 54,
 55, 57, 58, 60, 63, 65, 66,
 68, 69, 71, 73
 ok-mg-statement, 1, 2, 4, 6, 8, 10,
 11, 14, 16, 17, 20–23, 25,

26, 29–32, 34, 36, 38, 40,
 42–45, 47, 49, 51, 53–55,
 57, 58, 60, 63, 65, 66, 68,
 69, 71, 73
 ok-mg-statep, 1, 3, 4, 6, 8, 10, 11,
 14, 16, 17, 20–23, 25, 27,
 29–32, 34, 37, 38, 40, 42–
 45, 47, 49, 51, 53–55, 57,
 58, 60, 63, 65, 66, 68, 69,
 71, 73
 or-bool, 48, 50
 p, 20, 41, 52, 63, 74
 p-ctrl-stk-size, 1, 2, 4, 6, 8, 10, 11,
 13, 14, 16–24, 26, 29–34,
 36, 38–45, 47, 49–55, 57,
 58, 60, 62–67, 69, 71–75
 p-frame, 4, 5, 7, 9, 15, 17, 24–29,
 35–37, 45, 46, 48, 50, 56–
 61, 67–71
 p-state, 2–12, 14–19, 21, 23, 24, 26,
 28, 30–36, 38–40, 42, 44–
 51, 53, 55, 56, 58–62, 64,
 66–73, 75
 p-step, 2, 3, 5, 7, 9–12, 15, 17, 18,
 22, 24, 26, 28, 30–33, 35,
 38, 39, 43, 44, 46, 48, 50,
 54, 56, 58, 59, 61, 66–68,
 70, 72
 push, 2–12, 15–18, 23–33, 35–37, 39,
 43–48, 50, 55–61, 66–71
 resources-inadequatep, 1, 2, 4, 6, 8,
 10, 11, 14, 16, 17, 19, 22–
 24, 26, 29–32, 34, 36, 38,
 40, 43–45, 47, 49, 51, 54,
 55, 57, 58, 60, 63, 65–67,
 69, 71, 73
 rput, 9, 12, 30, 33, 49, 50, 60, 61,
 70, 72
 signatures-match, 1–4, 6, 8, 10–12,
 14, 16, 18, 20–23, 25, 27,
 29–32, 35, 37, 38, 40, 42–
 45, 47, 49, 52–55, 57, 59,
 61, 63, 65, 66, 68, 69, 71,
 73
 simple-typed-literalp, 49
 small-integerp, 6, 12, 14, 18, 27, 32,
 35, 38
 tag, 2–39, 41, 43–64, 66–72, 74
 top, 2–9, 12–30, 32–41, 43–74
 translate, 1, 3, 4, 6, 8, 10–14, 16,
 18–23, 25, 27, 29–34, 37–
 41, 43–45, 47, 49–52, 54,
 55, 57, 59, 60, 62–66, 68,
 69, 71–74
 translate-def-body, 1, 3, 4, 6, 8, 10–
 12, 14, 16, 18, 20, 22, 23,
 25, 27, 29–32, 34, 37, 38,
 40, 43–45, 47, 49, 51, 54,
 55, 57, 58, 60, 63, 65, 66,
 68, 69, 71, 73
 translate-proc-list, 2–13, 15–19, 21,
 23–28, 30–41, 43–51, 53, 55–
 62, 64, 66–70, 72–74
 untag, 6–9, 11, 12, 14, 15, 18, 27,
 28, 30, 32, 33, 35, 36, 38,
 39, 48–50, 60, 61, 70–72
 user-defined-procp, 1, 3, 4, 6, 8, 10–
 12, 14, 16, 18, 20, 22, 23,
 25, 27, 29–32, 34, 37, 38,
 40, 43–45, 47, 49, 52, 54,
 55, 57, 59, 60, 63, 65, 66,
 68, 69, 71, 73
 value, 2–9, 12, 14–16, 22–30, 33, 35–
 37, 43–50, 55–61, 66–72