

Ligo Formal Description

Syntax

variables (x)

label (l)

constructor (c)

declaration (d) =
| *type* x *is* te (Type declaration)
| *const* x ($:$ te) = e (Constant declaration)

type expression (te) =
| $(* te_i)$ (type of tuple)
| $(| l_i \text{ of } te_i)$ (type of sum)
| $\{ l_i : te_i \}$ (type of record)
| $te1 \rightarrow te2$ (type of function)
| l (type of variable)
| $l (te_i)$ (type of built in function)

expression (e) =
| *value* (values)
| *built_in* (built-in function)
| x (variables)
| $\lambda x . expr$ (lambda)
| $e1 \ e2$ (application)
| *let* $x = e1$ *in* $e2$ (let in)
| (e_i) (tuple)
| $c \ e$ (constructor)
| $\{ l_i = e_i \}$ (record)
| $[e1_i = e2_i]$ (map)
| $[[e1_i = e2_i]]$ (big map)
| $[e_i]$ (list)
| $\{ e_i \}$ (set)
| $e(.ai)$ (accessor)
| $e1[e2]$ (look up)
| *match* e *with matching* (matching)
| $e1; e2$ (sequence)
| *while* $e1$ *do* $e2$ (loop)
| $x.(a_i) = e$ (assign)
| *SKIP* (skip)
| e *as* T (ascription)

$access(a) =$

$ \text{int}$	(for tuples)
$ \text{string}$	(for record)
$ e$	(for map)

$value(v) =$

$ \text{literal}$	(values of built-in types)
$ \text{const } v$	(values of construct types)
$ \lambda x . \text{expr}$	(lambda)

$literal =$

$ \text{unit}$	()
$ \text{bool}$	()
$ \text{int}$	()
$ \text{nat}$	()
$ \text{mutez}$	()
$ \text{string}$	()
$ \text{bytes}$	()
$ \text{address}$	()
$ \text{timestamp}$	()
$ \text{operation}$	()

$matching(m) =$

$ \{ \text{true} \Rightarrow e; \text{false} \Rightarrow e; \}$	(match bool)
$ \{ \text{nil} \Rightarrow e; \text{cons}(\text{hd} :: \text{tl}) \Rightarrow e; \}$	(match list)
$ \{ \text{none} \Rightarrow e; \text{some}(x) \Rightarrow e; \}$	(match option)
$ (x_i) \Rightarrow e$	(match tuple)
$ (\text{const}_i(x_i) \Rightarrow e_i)$	(match variant)

$matching\ value(mv) =$

$ \{ \text{true} \Rightarrow v; \text{false} \Rightarrow v; \}$	(match bool value)
$ \{ \text{nil} \Rightarrow v; \text{cons}(\text{hd} :: \text{tl}) \Rightarrow v; \}$	(match list value)
$ \{ \text{none} \Rightarrow v; \text{some}(x) \Rightarrow v; \}$	(match option value)
$ (x_i) \Rightarrow v$	(match tuple value)
$ (\text{const}_i(x_i) \Rightarrow v_i)$	(match variant value)

Evaluation of expression

base

Values are not evaluted

$x \rightarrow v$ (*corresponding value in the environment*) (E-VARIABLE)

$\text{built in } (e_i) \rightarrow \text{built in result}$ (** evaluated depending on each case **) (E-BUILTIN)

$(\lambda x . e) v \rightarrow [x \rightarrow v] e$ (E-LAMBDA)

$$\frac{e1 \rightarrow e1'}{e1\ e2 \rightarrow e1'\ e2}$$
 (E-APP1)

$$\frac{e2 \rightarrow e2'}{v1\ e2 \rightarrow v1\ e2'}$$
 (E-APP2)

$\frac{e1 \rightarrow e1'}{let\ x = e1\ in\ e2 \rightarrow let\ x = e1'\ in\ e2}$	(E-LET)
$let\ x = v1\ in\ e2 \rightarrow [x \rightarrow v1]\ e2$	(E-LETIN)
$\frac{e1 \rightarrow e1'}{e1; e2 \rightarrow e1'; e2}$	(E-SEQ)
$unit; e2 \rightarrow e2$	(E-SEQNEXT)
$\frac{e1 \rightarrow e1'}{while\ e1\ then\ e2 \rightarrow while\ e1'\ then\ e2}$	(E-LOOP)
$while\ true(= e1)\ then\ e2 \rightarrow e2; while\ e1\ then\ e2$	(E-LOOPTRUE)
$while\ false\ then\ e2 \rightarrow unit$	(E-LOOPFALSE)
$SKIP \rightarrow unit$	(E-SKIP)
$\frac{e \rightarrow e'}{e\ as\ T \rightarrow e'\ as\ T}$	(E-ASCR1)
$v\ as\ T \rightarrow v$	(E-ASCR2)

data structure

$\frac{e \rightarrow e'}{c\ e \rightarrow c\ e'}$	(E-CONST)
$\frac{e_j \rightarrow e'_j}{(v_i, e_j, e_k) \rightarrow (v_i, e'_j, e_k)}$	(E-TUPLES)
$\frac{e_j \rightarrow e'_j}{\{l_i = v_i, l_j = e_j, l_k = e_k\} \rightarrow \{l_i = v_i, l_j = e'_j, l_k = e_k\}}$	(E-RECORDS)
$\frac{e2_j \rightarrow e2'_j}{[e1_i = v_i, e1_j = e2_j, e1_k = e2_k] \rightarrow [e1_i = v_i, e1_j = e2'_j, e1_k = e2_k]}$	(E-MAP)
$\frac{e2_j \rightarrow e2'_j}{[[e1_i = v_i, e1_j = e2_j, e1_k = e2_k]] \rightarrow [[e1_i = v_i, e1_j = e2'_j, e1_k = e2_k]]}$	(E-BIGMAP)
$\frac{e_j \rightarrow e'_j}{[v_i, e_j, e_k] \rightarrow [v_i, e'_j, e_k]}$	(E-LIST)
$\frac{e_j \rightarrow e'_j}{\{v_i, e_j, e_k\} \rightarrow \{v_i, e'_j, e_k\}}$	(E-SET)
$\frac{e \rightarrow e'}{e(.a_i) \rightarrow e'(.a_i)}$	(E-ACCESS)

look up

$(v_i)[j] \rightarrow v_j$	(E-LUPTUPLE)
$\{l_i = v_i\}[l_j] \rightarrow v_j$	(E-LUPRECORD)
$[e_i = v_i][e_j] \rightarrow v_j$	(E-LUPMAP)
$[[e_i = v_i]][e_j] \rightarrow v_j$	(E-LUPBIGMAP)
$[v_i][j] \rightarrow v_j$	(E-LUPLIST)
$\{v_i\}[j] \rightarrow v_j$	(E-LUPSET)
$\frac{e \rightarrow e'}{x(.a_i) = e \rightarrow x(.a_i) = e'}$	(E-ASSIGN)
$x(.a_i) = v \rightarrow x'(.a_i)\ with\ x'\ as\ x\ with\ field\ (.a_i)\ replace\ by\ v$	(E-ASSIGN2)

matching

$$\begin{array}{c}
\frac{e \rightarrow e'}{\text{match } e \text{ with } m \rightarrow \text{match } e' \text{ with } m} \quad (\text{E-MATCH1}) \\
\frac{m \rightarrow m'}{\text{match } v \text{ with } m \rightarrow \text{match } v \text{ with } m'} \quad (\text{E-MATCH2}) \\
\text{match } v \text{ with } mv \rightarrow v' \text{ if } v \Rightarrow v' \text{ in } mv \quad (\text{E-MATCH}) \\
\frac{e1 \rightarrow e1'}{\{ \text{true} \Rightarrow e1; \text{false} \Rightarrow e2; \} \rightarrow \{ \text{true} \Rightarrow e1'; \text{false} \Rightarrow e2; \}} \quad (\text{E-MAcTHBOOL1}) \\
\frac{e2 \rightarrow e2'}{\{ \text{true} \Rightarrow v1; \text{false} \Rightarrow e2; \} \rightarrow \{ \text{true} \Rightarrow v1; \text{false} \Rightarrow e2'; \}} \quad (\text{E-MAcTHBOOL2}) \\
\frac{e1 \rightarrow e1'}{\{ \text{nil} \Rightarrow e1; \text{cons}(hd :: tl) \Rightarrow e2; \} \rightarrow \{ \text{nil} \Rightarrow e1'; \text{cons}(hd :: tl) \Rightarrow e2; \}} \quad (\text{E-MATCHLIST1}) \\
\frac{e2 \rightarrow e2'}{\{ \text{nil} \Rightarrow v1; \text{cons}(hd :: tl) \Rightarrow e2; \} \rightarrow \{ \text{nil} \Rightarrow v1; \text{cons}(hd :: tl) \Rightarrow e2'; \}} \quad (\text{E-MATCHLIST2}) \\
\frac{e1 \rightarrow e1'}{\{ \text{none} \Rightarrow e1; \text{some}(x) \Rightarrow e2; \} \rightarrow \{ \text{none} \Rightarrow e1'; \text{some}(x) \Rightarrow e2; \}} \quad (\text{E-MATCHOPT1}) \\
\frac{e2 \rightarrow e2'}{\{ \text{none} \Rightarrow v1; \text{some}(x) \Rightarrow e2; \} \rightarrow \{ \text{none} \Rightarrow v1'; \text{some}(x) \Rightarrow e2'; \}} \quad (\text{E-MATCHOPT2}) \\
\frac{e \rightarrow e'}{(x_i) \Rightarrow e \rightarrow (x_i) \Rightarrow e'} \quad (\text{E-MATCHTUPLE}) \\
\\
\frac{e_j \rightarrow e'_j}{(c_i(x_i) \Rightarrow v_i, c_j(x_j) \Rightarrow e_j, c_k(x_k) \Rightarrow e_k) \rightarrow (c_i(x_i) \Rightarrow v_i, c_j(x_j) \Rightarrow e'_j, c_k(x_k) \Rightarrow e_k)} \quad (\text{E-MATCHVARIANT})
\end{array}$$

Derive form

$$\begin{array}{l}
e1; e2 \iff (\lambda x : \text{Unit}.e1) e2 \text{ with } x \text{ not a free variable in } e1 \\
\text{let } x = e1 \text{ in } e2 \iff (\lambda x : T1.e2) e1
\end{array}$$