

Preprocessed HAL/S

BNF-converter

March 6, 2023

This document was automatically generated by the *BNF-Converter*. It was generated together with the lexer, the parser, and the abstract syntax module, which guarantees that the document matches with the implementation of the language (provided no hand-hacking has taken place).

The lexical structure of HAL/S

Literals

NeqToken = ([“~”] | {“NOT”}[“ ”]*)[“=”]
LeToken = ([“~”] | {“NOT”}[“ ”]*)[“>”] | {“<=”}
GeToken = ([“~”] | {“NOT”}[“ ”]*)[“<”] | {“>=”}
NoArgUserFuncIdentifierToken = {“nf_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
BitIdentifierToken = {“b_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
BitFunctionIdentifierToken = {“bf_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
CharFunctionIdentifierToken = {“cf_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
CharIdentifierToken = {“c_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
StructIdentifierToken = {“s_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
StructFunctionIdentifierToken = {“sf_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
LabelToken = {“l_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
EventToken = {“e_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
ArithFieldToken = {“a_”}<letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
IdentifierToken = <letter>(((<letter> | <digit> | {“_”}) * (<letter> | <digit>)))?
StringToken = [“”](<letter> | <digit> | {“”}) | [“_+-% \$()|*./&~=<>#@,;:}{!?””] * [“”]
TextToken = [“”](<letter> | <digit> | [“_+-% \$()|*./&~=<>#@,;:}{!?””]) * [“”]
LevelToken = [“123456789”] | [“12”]<digit> | [“3”][“012”]
NumberToken = <digit>+
CompoundToken = (<digit> + ([“.”]<digit>*)? | [“.”]<digit>+)([“EBH”][“_”?<digit>+)*

Reserved words and symbols

The set of reserved words is the set of terminals appearing in the grammar. Those reserved words that consist of non-letter characters are called symbols, and they are treated in a different way from those that are similar to identifiers. The lexer follows rules familiar from languages like Haskell, C, and Java, including longest match and spacing conventions.

The reserved words used in HAL/S are the following:

| | | |
|-------------|-----------|-----------|
| ABS | ABVAL | ACCESS |
| AFTER | ALIGNED | AND |
| ARCCOS | ARCCOSH | ARCSIN |
| ARCSINH | ARCTAN | ARCTAN2 |
| ARCTANH | ARRAY | ASSIGN |
| AT | AUTOMATIC | BIN |
| BIT | BOOLEAN | BY |
| CALL | CANCEL | CASE |
| CAT | CEILING | CHAR |
| CHARACTER | CLOCKTIME | CLOSE |
| COLUMN | COMPOOL | CONSTANT |
| COS | COSH | DATE |
| DEC | DECLARE | DENSE |
| DEPENDENT | DET | DIV |
| DO | DOUBLE | ELSE |
| END | EQUATE | ERRGRP |
| ERRNUM | ERROR | EVENT |
| EVERY | EXCLUSIVE | EXIT |
| EXP | EXTERNAL | FALSE |
| FILE | FLOOR | FOR |
| FUNCTION | GO | HEX |
| IF | IGNORE | IN |
| INDEX | INITIAL | INTEGER |
| INVERSE | LATCHED | LENGTH |
| LINE | LJUST | LOCK |
| LOG | MATRIX | MAX |
| MIDVAL | MIN | MOD |
| NAME | NEXTIME | NONHAL |
| NOT | NULL | OCT |
| ODD | OFF | ON |
| OR | PAGE | PRIO |
| PRIORITY | PROCEDURE | PROD |
| PROGRAM | RANDOM | RANDOMG |
| READ | READALL | REENTRANT |
| REMAINDER | REMOTE | REPEAT |
| REPLACE | RESET | RETURN |
| RIGID | RJUST | ROUND |
| RUNTIME | SCALAR | SCHEDULE |
| SEND | SET | SHL |
| SHR | SIGN | SIGNAL |
| SIGNUM | SIN | SINGLE |
| SINH | SIZE | SKIP |
| SQRT | STATIC | STRUCTURE |
| SUBBIT | SUM | SYSTEM |
| TAB | TAN | TANH |
| TASK | TEMPORARY | TERMINATE |
| THEN | TO | TRACE |
| TRANSPPOSE | TRIM | TRUE |
| TRUNCATE | UNIT | UNTIL |
| UPDATE | VECTOR | WAIT |
| WHILE | WRITE | XOR |
| initialized | typeof | typeofv |

The symbols used in HAL/S are the following:

```

,      )      (
/      +      -
*      .      **T
**     #      [
]      {      }
$      @      ;
:      |      &
||     ~      <
=      >      %

```

Comments

There are no single-line comments in the grammar.
Multiple-line comments are enclosed with `/*` and `*/`.

The syntactic structure of HAL/S

Non-terminals are enclosed between `<` and `>`. The symbols `::=` (production), `|` (union) and `ε` (empty rule) belong to the BNF notation. All other symbols are terminals.

```

<DECLARE-BODY> ::= <DECLARATION-LIST>
                  | <ATTRIBUTES> , <DECLARATION-LIST>

<ATTRIBUTES> ::= <ARRAY-SPEC> <TYPE-AND-MINOR-ATTR>
                  | <ARRAY-SPEC>
                  | <TYPE-AND-MINOR-ATTR>

<DECLARATION> ::= <NAME-ID>
                  | <NAME-ID> <ATTRIBUTES>
                  | <LabelToken>
                  | <LabelToken> <TYPE-AND-MINOR-ATTR>
                  | <LabelToken> PROCEDURE <MINOR-ATTR-LIST>
                  | <LabelToken> PROCEDURE
                  | <LabelToken> FUNCTION <TYPE-AND-MINOR-ATTR>
                  | <NoArgUserFuncIdentifierToken> FUNCTION <TYPE-AND-MINOR-ATTR>
                  | <LabelToken> FUNCTION
                  | <NoArgUserFuncIdentifierToken> FUNCTION
                  | <EventToken> EVENT
                  | <EventToken> EVENT <MINOR-ATTR-LIST>
                  | <EventToken>
                  | <EventToken> <MINOR-ATTR-LIST>

<ARRAY-SPEC> ::= <ARRAY-HEAD> <LITERAL-EXP-OR-STAR> )
                  | FUNCTION
                  | PROCEDURE
                  | PROGRAM
                  | TASK

<TYPE-AND-MINOR-ATTR> ::= <TYPE-SPEC>
                          | <TYPE-SPEC> <MINOR-ATTR-LIST>
                          | <MINOR-ATTR-LIST>

<IDENTIFIER> ::= <IdentifierToken>

```

$\langle \text{SQ-DQ-NAME} \rangle ::= \langle \text{DOUBLY-QUAL-NAME-HEAD} \rangle \langle \text{LITERAL-EXP-OR-STAR} \rangle)$
 $\quad \quad \quad | \quad \langle \text{ARITH-CONV} \rangle$

$\langle \text{DOUBLY-QUAL-NAME-HEAD} \rangle ::= \text{VECTOR} ($
 $\quad \quad \quad | \quad \text{MATRIX} (\langle \text{LITERAL-EXP-OR-STAR} \rangle ,$

$\langle \text{ARITH-CONV} \rangle ::= \text{INTEGER}$
 $\quad \quad \quad | \quad \text{SCALAR}$
 $\quad \quad \quad | \quad \text{VECTOR}$
 $\quad \quad \quad | \quad \text{MATRIX}$

$\langle \text{DECLARATION-LIST} \rangle ::= \langle \text{DECLARATION} \rangle$
 $\quad \quad \quad | \quad \langle \text{DCL-LIST-COMMA} \rangle \langle \text{DECLARATION} \rangle$

$\langle \text{NAME-ID} \rangle ::= \langle \text{IDENTIFIER} \rangle$
 $\quad \quad \quad | \quad \langle \text{IDENTIFIER} \rangle \text{ NAME}$
 $\quad \quad \quad | \quad \langle \text{BIT-ID} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-ID} \rangle$
 $\quad \quad \quad | \quad \langle \text{BitFunctionIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{CharFunctionIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{StructIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{StructFunctionIdentifierToken} \rangle$

$\langle \text{ARITH-EXP} \rangle ::= \langle \text{TERM} \rangle$
 $\quad \quad \quad | \quad \langle \text{PLUS} \rangle \langle \text{TERM} \rangle$
 $\quad \quad \quad | \quad \langle \text{MINUS} \rangle \langle \text{TERM} \rangle$
 $\quad \quad \quad | \quad \langle \text{ARITH-EXP} \rangle \langle \text{PLUS} \rangle \langle \text{TERM} \rangle$
 $\quad \quad \quad | \quad \langle \text{ARITH-EXP} \rangle \langle \text{MINUS} \rangle \langle \text{TERM} \rangle$

$\langle \text{TERM} \rangle ::= \langle \text{PRODUCT} \rangle$
 $\quad \quad \quad | \quad \langle \text{PRODUCT} \rangle / \langle \text{TERM} \rangle$

$\langle \text{PLUS} \rangle ::= +$

$\langle \text{MINUS} \rangle ::= -$

$\langle \text{PRODUCT} \rangle ::= \langle \text{FACTOR} \rangle$
 $\quad \quad \quad | \quad \langle \text{FACTOR} \rangle * \langle \text{PRODUCT} \rangle$
 $\quad \quad \quad | \quad \langle \text{FACTOR} \rangle . \langle \text{PRODUCT} \rangle$
 $\quad \quad \quad | \quad \langle \text{FACTOR} \rangle \langle \text{PRODUCT} \rangle$

$\langle \text{FACTOR} \rangle ::= \langle \text{PRIMARY} \rangle$
 $\quad \quad \quad | \quad \langle \text{PRIMARY} \rangle \langle \text{EXPONENTIATION} \rangle \langle \text{FACTOR} \rangle$
 $\quad \quad \quad | \quad \langle \text{PRIMARY} \rangle **T$

$\langle \text{EXPONENTIATION} \rangle ::= **$

$\langle \text{PRIMARY} \rangle ::= \langle \text{ARITH-VAR} \rangle$
 $\quad \quad \quad | \quad \langle \text{PRE-PRIMARY} \rangle$
 $\quad \quad \quad | \quad \langle \text{MODIFIED-ARITH-FUNC} \rangle$
 $\quad \quad \quad | \quad \langle \text{PRE-PRIMARY} \rangle \langle \text{QUALIFIER} \rangle$

$\langle \text{ARITH-VAR} \rangle ::= \langle \text{ARITH-ID} \rangle$
 $\quad | \langle \text{ARITH-ID} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad | [\langle \text{ARITH-ID} \rangle]$
 $\quad | [\langle \text{ARITH-ID} \rangle] \langle \text{SUBSCRIPT} \rangle$
 $\quad | \{ \langle \text{QUAL-STRUCT} \rangle \}$
 $\quad | \{ \langle \text{QUAL-STRUCT} \rangle \} \langle \text{SUBSCRIPT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{ARITH-ID} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{ARITH-ID} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{PRE-PRIMARY} \rangle ::= (\langle \text{ARITH-EXP} \rangle)$
 $\quad | \langle \text{NUMBER} \rangle$
 $\quad | \langle \text{COMPOUND-NUMBER} \rangle$
 $\quad | \langle \text{ARITH-FUNC} \rangle (\langle \text{CALL-LIST} \rangle)$
 $\quad | \text{typeof} (\langle \text{CALL-LIST} \rangle)$
 $\quad | \text{typeofv} (\langle \text{CALL-LIST} \rangle)$
 $\quad | \langle \text{SHAPING-HEAD} \rangle$
 $\quad | \langle \text{SHAPING-HEAD} \rangle , *)$
 $\quad | \langle \text{LabelToken} \rangle (\langle \text{CALL-LIST} \rangle)$

$\langle \text{NUMBER} \rangle ::= \langle \text{SIMPLE-NUMBER} \rangle$
 $\quad | \langle \text{LEVEL} \rangle$

$\langle \text{LEVEL} \rangle ::= \langle \text{LevelToken} \rangle$

$\langle \text{COMPOUND-NUMBER} \rangle ::= \langle \text{CompoundToken} \rangle$

$\langle \text{SIMPLE-NUMBER} \rangle ::= \langle \text{NumberToken} \rangle$

$\langle \text{MODIFIED-ARITH-FUNC} \rangle ::= \langle \text{NO-ARG-ARITH-FUNC} \rangle$
 $\quad | \langle \text{NO-ARG-ARITH-FUNC} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{NO-ARG-ARITH-FUNC} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{NO-ARG-ARITH-FUNC} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{SHAPING-HEAD} \rangle ::= \text{INTEGER} (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{SCALAR} (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{VECTOR} (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{MATRIX} (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{INTEGER} \langle \text{SUBSCRIPT} \rangle (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{SCALAR} \langle \text{SUBSCRIPT} \rangle (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{VECTOR} \langle \text{SUBSCRIPT} \rangle (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \text{MATRIX} \langle \text{SUBSCRIPT} \rangle (\langle \text{REPEATED-CONSTANT} \rangle$
 $\quad | \langle \text{SHAPING-HEAD} \rangle , \langle \text{REPEATED-CONSTANT} \rangle$

$\langle \text{CALL-LIST} \rangle ::= \langle \text{LIST-EXP} \rangle$
 $\quad | \langle \text{CALL-LIST} \rangle , \langle \text{LIST-EXP} \rangle$

$\langle \text{LIST-EXP} \rangle ::= \langle \text{EXPRESSION} \rangle$
 $\quad | \langle \text{ARITH-EXP} \rangle \# \langle \text{EXPRESSION} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle$

$\langle \text{EXPRESSION} \rangle ::= \langle \text{ARITH-EXP} \rangle$
 $\quad | \langle \text{BIT-EXP} \rangle$
 $\quad | \langle \text{CHAR-EXP} \rangle$
 $\quad | \langle \text{NAME-EXP} \rangle$
 $\quad | \langle \text{STRUCTURE-EXP} \rangle$

$$\begin{aligned}
 \langle \text{ARITH-ID} \rangle & ::= \langle \text{IDENTIFIER} \rangle \\
 & | \langle \text{ArithFieldToken} \rangle \\
 \langle \text{NO-ARG-ARITH-FUNC} \rangle & ::= \langle \text{NoArgUserFuncIdentifierToken} \rangle \\
 & | \text{CLOCKTIME} \\
 & | \text{DATE} \\
 & | \text{ERRGRP} \\
 & | \text{ERRNUM} \\
 & | \text{PRIO} \\
 & | \text{RANDOM} \\
 & | \text{RANDOMG} \\
 & | \text{RUNTIME}
 \end{aligned}$$

| | | |
|-------------------------------------|-----|---|
| $\langle \text{ARITH-FUNC} \rangle$ | ::= | NEXTIME ABS CEILING DIV FLOOR MIDVAL MOD ODD REMAINDER ROUND SIGN SIGNUM TRUNCATE ARCCOS ARCCOSH ARCSIN ARCSINH ARCTAN2 ARCTAN ARCTANH COS COSH EXP LOG SIN SINH SQRT TAN TANH SHL SHR ABVAL DET TRACE UNIT INDEX LENGTH INVERSE TRANSPOSE PROD SUM SIZE MAX MIN |
| $\langle \text{SUBSCRIPT} \rangle$ | ::= | $\langle \text{SUB-HEAD} \rangle$) $\langle \text{QUALIFIER} \rangle$ \$ $\langle \text{NUMBER} \rangle$ \$ $\langle \text{ARITH-VAR} \rangle$ |
| $\langle \text{QUALIFIER} \rangle$ | ::= | \$ (@ $\langle \text{PREC-SPEC} \rangle$) \$ ($\langle \text{SCALE-HEAD} \rangle$ $\langle \text{ARITH-EXP} \rangle$) \$ (@ $\langle \text{PREC-SPEC} \rangle$, $\langle \text{SCALE-HEAD} \rangle$ $\langle \text{ARITH-EXP} \rangle$) \$ (@ $\langle \text{RADIX} \rangle$) |

$$\begin{aligned}
\langle \text{SCALE-HEAD} \rangle & ::= \text{@} \\
& \quad | \quad \text{@ @} \\
\langle \text{PREC-SPEC} \rangle & ::= \text{SINGLE} \\
& \quad | \quad \text{DOUBLE} \\
\langle \text{SUB-START} \rangle & ::= \$ (\\
& \quad | \quad \$ (\text{@} \langle \text{PREC-SPEC} \rangle , \\
& \quad | \quad \langle \text{SUB-HEAD} \rangle ; \\
& \quad | \quad \langle \text{SUB-HEAD} \rangle : \\
& \quad | \quad \langle \text{SUB-HEAD} \rangle , \\
\langle \text{SUB-HEAD} \rangle & ::= \langle \text{SUB-START} \rangle \\
& \quad | \quad \langle \text{SUB-START} \rangle \langle \text{SUB} \rangle \\
\langle \text{SUB} \rangle & ::= \langle \text{SUB-EXP} \rangle \\
& \quad | \quad * \\
& \quad | \quad \langle \text{SUB-RUN-HEAD} \rangle \langle \text{SUB-EXP} \rangle \\
& \quad | \quad \langle \text{ARITH-EXP} \rangle \text{ AT } \langle \text{SUB-EXP} \rangle \\
\langle \text{SUB-RUN-HEAD} \rangle & ::= \langle \text{SUB-EXP} \rangle \text{ TO} \\
\langle \text{SUB-EXP} \rangle & ::= \langle \text{ARITH-EXP} \rangle \\
& \quad | \quad \langle \text{POUND-EXPRESSION} \rangle \\
\langle \text{POUND-EXPRESSION} \rangle & ::= \# \\
& \quad | \quad \langle \text{POUND-EXPRESSION} \rangle \langle \text{PLUS} \rangle \langle \text{TERM} \rangle \\
& \quad | \quad \langle \text{POUND-EXPRESSION} \rangle \langle \text{MINUS} \rangle \langle \text{TERM} \rangle \\
\langle \text{BIT-EXP} \rangle & ::= \langle \text{BIT-FACTOR} \rangle \\
& \quad | \quad \langle \text{BIT-EXP} \rangle \langle \text{OR} \rangle \langle \text{BIT-FACTOR} \rangle \\
\langle \text{BIT-FACTOR} \rangle & ::= \langle \text{BIT-CAT} \rangle \\
& \quad | \quad \langle \text{BIT-FACTOR} \rangle \langle \text{AND} \rangle \langle \text{BIT-CAT} \rangle \\
\langle \text{BIT-CAT} \rangle & ::= \langle \text{BIT-PRIM} \rangle \\
& \quad | \quad \langle \text{BIT-CAT} \rangle \langle \text{CAT} \rangle \langle \text{BIT-PRIM} \rangle \\
& \quad | \quad \langle \text{NOT} \rangle \langle \text{BIT-PRIM} \rangle \\
& \quad | \quad \langle \text{BIT-CAT} \rangle \langle \text{CAT} \rangle \langle \text{NOT} \rangle \langle \text{BIT-PRIM} \rangle \\
\langle \text{OR} \rangle & ::= \langle \text{CHAR-VERTICAL-BAR} \rangle \\
& \quad | \quad \text{OR} \\
\langle \text{CHAR-VERTICAL-BAR} \rangle & ::= | \\
\langle \text{AND} \rangle & ::= \& \\
& \quad | \quad \text{AND}
\end{aligned}$$

$\langle \text{BIT-PRIM} \rangle ::= \langle \text{BIT-VAR} \rangle$
 $\quad | \langle \text{LABEL-VAR} \rangle$
 $\quad | \langle \text{EVENT-VAR} \rangle$
 $\quad | \langle \text{BIT-CONST} \rangle$
 $\quad | (\langle \text{BIT-EXP} \rangle)$
 $\quad | \langle \text{SUBBIT-HEAD} \rangle \langle \text{EXPRESSION} \rangle$
 $\quad | \langle \text{BIT-FUNC-HEAD} \rangle (\langle \text{CALL-LIST} \rangle)$
 $\quad | \text{initialized} (\langle \text{CALL-LIST} \rangle)$
 $\quad | [\langle \text{BIT-VAR} \rangle]$
 $\quad | [\langle \text{BIT-VAR} \rangle] \langle \text{SUBSCRIPT} \rangle$
 $\quad | \{ \langle \text{BIT-VAR} \rangle \}$
 $\quad | \{ \langle \text{BIT-VAR} \rangle \} \langle \text{SUBSCRIPT} \rangle$

$\langle \text{CAT} \rangle ::= ||$
 $\quad | \text{CAT}$

$\langle \text{NOT} \rangle ::= \text{NOT}$
 $\quad | \sim$

$\langle \text{BIT-VAR} \rangle ::= \langle \text{BIT-ID} \rangle$
 $\quad | \langle \text{BIT-ID} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{BIT-ID} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{BIT-ID} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{LABEL-VAR} \rangle ::= \langle \text{LABEL} \rangle$
 $\quad | \langle \text{LABEL} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{LABEL} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{LABEL} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{EVENT-VAR} \rangle ::= \langle \text{EVENT} \rangle$
 $\quad | \langle \text{EVENT} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{EVENT} \rangle$
 $\quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{EVENT} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{BIT-CONST-HEAD} \rangle ::= \langle \text{RADIX} \rangle$
 $\quad | \langle \text{RADIX} \rangle (\langle \text{NUMBER} \rangle)$

$\langle \text{BIT-CONST} \rangle ::= \langle \text{BIT-CONST-HEAD} \rangle \langle \text{CHAR-STRING} \rangle$
 $\quad | \text{TRUE}$
 $\quad | \text{FALSE}$
 $\quad | \text{ON}$
 $\quad | \text{OFF}$

$\langle \text{RADIX} \rangle ::= \text{HEX}$
 $\quad | \text{OCT}$
 $\quad | \text{BIN}$
 $\quad | \text{DEC}$

$\langle \text{CHAR-STRING} \rangle ::= \langle \text{StringToken} \rangle$

$\langle \text{SUBBIT-HEAD} \rangle ::= \langle \text{SUBBIT-KEY} \rangle ($
 $\quad | \langle \text{SUBBIT-KEY} \rangle \langle \text{SUBSCRIPT} \rangle ($

$\langle \text{SUBBIT-KEY} \rangle ::= \text{SUBBIT}$

$\langle \text{BIT-FUNC-HEAD} \rangle ::= \langle \text{BIT-FUNC} \rangle$
 $\quad | \text{BIT}$
 $\quad | \text{BIT} \langle \text{SUB-OR-QUALIFIER} \rangle$

$\langle \text{BIT-ID} \rangle ::= \langle \text{BitIdentifierToken} \rangle$

$\langle \text{LABEL} \rangle ::= \langle \text{LabelToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{BitFunctionIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{CharFunctionIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{StructFunctionIdentifierToken} \rangle$

$\langle \text{BIT-FUNC} \rangle ::= \text{XOR}$
 $\quad \quad \quad | \quad \langle \text{BitFunctionIdentifierToken} \rangle$

$\langle \text{EVENT} \rangle ::= \langle \text{EventToken} \rangle$

$\langle \text{SUB-OR-QUALIFIER} \rangle ::= \langle \text{SUBSCRIPT} \rangle$
 $\quad \quad \quad | \quad \langle \text{BIT-QUALIFIER} \rangle$

$\langle \text{BIT-QUALIFIER} \rangle ::= < \$ (@ \langle \text{RADIX} \rangle)$

$\langle \text{CHAR-EXP} \rangle ::= \langle \text{CHAR-PRIM} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-EXP} \rangle \langle \text{CAT} \rangle \langle \text{CHAR-PRIM} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-EXP} \rangle \langle \text{CAT} \rangle \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \langle \text{ARITH-EXP} \rangle \langle \text{CAT} \rangle \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \langle \text{ARITH-EXP} \rangle \langle \text{CAT} \rangle \langle \text{CHAR-PRIM} \rangle$

$\langle \text{CHAR-PRIM} \rangle ::= \langle \text{CHAR-VAR} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-CONST} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-FUNC-HEAD} \rangle (\langle \text{CALL-LIST} \rangle)$
 $\quad \quad \quad | \quad (\langle \text{CHAR-EXP} \rangle)$

$\langle \text{CHAR-FUNC-HEAD} \rangle ::= \langle \text{CHAR-FUNC} \rangle$
 $\quad \quad \quad | \quad \text{CHARACTER} \langle \text{SUB-OR-QUALIFIER} \rangle$

$\langle \text{CHAR-VAR} \rangle ::= \langle \text{CHAR-ID} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-ID} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\quad \quad \quad | \quad \langle \text{QUAL-STRUCT} \rangle . \langle \text{CHAR-ID} \rangle$
 $\quad \quad \quad | \quad \langle \text{QUAL-STRUCT} \rangle . \langle \text{CHAR-ID} \rangle \langle \text{SUBSCRIPT} \rangle$

$\langle \text{CHAR-CONST} \rangle ::= \langle \text{CHAR-STRING} \rangle$
 $\quad \quad \quad | \quad \text{CHAR} (\langle \text{NUMBER} \rangle) \langle \text{CHAR-STRING} \rangle$

$\langle \text{CHAR-FUNC} \rangle ::= \text{LJUST}$
 $\quad \quad \quad | \quad \text{RJUST}$
 $\quad \quad \quad | \quad \text{TRIM}$
 $\quad \quad \quad | \quad \langle \text{CharFunctionIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \text{CHARACTER}$

$\langle \text{CHAR-ID} \rangle ::= \langle \text{CharIdentifierToken} \rangle$

$\langle \text{NAME-EXP} \rangle ::= \langle \text{NAME-KEY} \rangle (\langle \text{NAME-VAR} \rangle)$
 $\quad \quad \quad | \quad \text{NULL}$
 $\quad \quad \quad | \quad \langle \text{NAME-KEY} \rangle (\text{NULL})$

$\langle \text{NAME-KEY} \rangle ::= \text{NAME}$

$\langle \text{NAME-VAR} \rangle ::= \langle \text{VARIABLE} \rangle$
 $\quad \quad \quad | \quad \langle \text{MODIFIED-ARITH-FUNC} \rangle$
 $\quad \quad \quad | \quad \langle \text{LABEL-VAR} \rangle$

$\langle \text{VARIABLE} \rangle ::= \langle \text{ARITH-VAR} \rangle$
 $\quad \quad \quad | \langle \text{BIT-VAR} \rangle$
 $\quad \quad \quad | \langle \text{SUBBIT-HEAD} \rangle \langle \text{VARIABLE} \rangle$
 $\quad \quad \quad | \langle \text{CHAR-VAR} \rangle$
 $\quad \quad \quad | \langle \text{NAME-KEY} \rangle (\langle \text{NAME-VAR} \rangle)$
 $\quad \quad \quad | \langle \text{EVENT-VAR} \rangle$
 $\quad \quad \quad | \langle \text{STRUCTURE-VAR} \rangle$
 $\langle \text{STRUCTURE-EXP} \rangle ::= \langle \text{STRUCTURE-VAR} \rangle$
 $\quad \quad \quad | \langle \text{STRUCT-FUNC-HEAD} \rangle (\langle \text{CALL-LIST} \rangle)$
 $\quad \quad \quad | \langle \text{STRUC-INLINE-DEF} \rangle \langle \text{CLOSING} \rangle ;$
 $\quad \quad \quad | \langle \text{STRUC-INLINE-DEF} \rangle \langle \text{BLOCK-BODY} \rangle \langle \text{CLOSING} \rangle ;$
 $\langle \text{STRUCT-FUNC-HEAD} \rangle ::= \langle \text{STRUCT-FUNC} \rangle$
 $\langle \text{STRUCTURE-VAR} \rangle ::= \langle \text{QUAL-STRUCT} \rangle \langle \text{SUBSCRIPT} \rangle$
 $\langle \text{STRUCT-FUNC} \rangle ::= \langle \text{StructFunctionIdentifierToken} \rangle$
 $\langle \text{QUAL-STRUCT} \rangle ::= \langle \text{STRUCTURE-ID} \rangle$
 $\quad \quad \quad | \langle \text{QUAL-STRUCT} \rangle . \langle \text{STRUCTURE-ID} \rangle$
 $\langle \text{STRUCTURE-ID} \rangle ::= \langle \text{StructIdentifierToken} \rangle$
 $\langle \text{ASSIGNMENT} \rangle ::= \langle \text{VARIABLE} \rangle \langle \text{EQUALS} \rangle \langle \text{EXPRESSION} \rangle$
 $\quad \quad \quad | \langle \text{VARIABLE} \rangle , \langle \text{ASSIGNMENT} \rangle$
 $\quad \quad \quad | \langle \text{QUAL-STRUCT} \rangle \langle \text{EQUALS} \rangle \langle \text{EXPRESSION} \rangle$
 $\quad \quad \quad | \langle \text{QUAL-STRUCT} \rangle , \langle \text{ASSIGNMENT} \rangle$
 $\langle \text{EQUALS} \rangle ::= =$
 $\langle \text{STATEMENT} \rangle ::= \langle \text{BASIC-STATEMENT} \rangle$
 $\quad \quad \quad | \langle \text{OTHER-STATEMENT} \rangle$
 $\quad \quad \quad | \langle \text{INLINE-DEFINITION} \rangle$

```

<BASIC-STATEMENT> ::= <ASSIGNMENT> ;
| <LABEL-DEFINITION> <BASIC-STATEMENT>
| EXIT ;
| EXIT <LABEL> ;
| REPEAT ;
| REPEAT <LABEL> ;
| GO TO <LABEL> ;
| ;
| <CALL-KEY> ;
| <CALL-KEY> ( <CALL-LIST> ) ;
| <CALL-KEY> <ASSIGN> ( <CALL-ASSIGN-LIST> ) ;
| <CALL-KEY> ( <CALL-LIST> ) <ASSIGN> ( <CALL-ASSIGN-LIST> ) ;
| RETURN ;
| RETURN <EXPRESSION> ;
| <DO-GROUP-HEAD> <ENDING> ;
| <READ-KEY> ;
| <READ-PHRASE> ;
| <WRITE-KEY> ;
| <WRITE-PHRASE> ;
| <FILE-EXP> <EQUALS> <EXPRESSION> ;
| <VARIABLE> <EQUALS> <FILE-EXP> ;
| <FILE-EXP> <EQUALS> <QUAL-STRUCT> ;
| <WAIT-KEY> FOR DEPENDENT ;
| <WAIT-KEY> <ARITH-EXP> ;
| <WAIT-KEY> UNTIL <ARITH-EXP> ;
| <WAIT-KEY> FOR <BIT-EXP> ;
| <TERMINATOR> ;
| <TERMINATOR> <TERMINATE-LIST> ;
| UPDATE PRIORITY TO <ARITH-EXP> ;
| UPDATE PRIORITY <LABEL-VAR> TO <ARITH-EXP> ;
| <SCHEDULE-PHRASE> ;
| <SCHEDULE-PHRASE> <SCHEDULE-CONTROL> ;
| <SIGNAL-CLAUSE> ;
| SEND ERROR <SUBSCRIPT> ;
| SEND ERROR ;
| <ON-CLAUSE> ;
| <ON-CLAUSE> AND <SIGNAL-CLAUSE> ;
| OFF ERROR <SUBSCRIPT> ;
| OFF ERROR ;
| <PERCENT-MACRO-NAME> ;
| <PERCENT-MACRO-HEAD> <PERCENT-MACRO-ARG> ) ;

<OTHER-STATEMENT> ::= <IF-STATEMENT>
| <ON-PHRASE> <STATEMENT>
| <LABEL-DEFINITION> <OTHER-STATEMENT>

<IF-STATEMENT> ::= <IF-CLAUSE> <STATEMENT>
| <TRUE-PART> <STATEMENT>

<IF-CLAUSE> ::= <IF> <RELATIONAL-EXP> <THEN>
| <IF> <BIT-EXP> <THEN>

<TRUE-PART> ::= <IF-CLAUSE> <BASIC-STATEMENT> ELSE

<IF> ::= IF

```

```

<THEN> ::= THEN

<RELATIONAL-EXP> ::= <RELATIONAL-FACTOR>
                  | <RELATIONAL-EXP> <OR> <RELATIONAL-FACTOR>

<RELATIONAL-FACTOR> ::= <REL-PRIM>
                       | <RELATIONAL-FACTOR> <AND> <REL-PRIM>

<REL-PRIM> ::= ( <RELATIONAL-EXP> )
             | <NOT> ( <RELATIONAL-EXP> )
             | <COMPARISON>

<COMPARISON> ::= <ARITH-EXP> <EQUALS> <ARITH-EXP>
                | <CHAR-EXP> <EQUALS> <CHAR-EXP>
                | <BIT-CAT> <EQUALS> <BIT-CAT>
                | <STRUCTURE-EXP> <EQUALS> <STRUCTURE-EXP>
                | <NAME-EXP> <EQUALS> <NAME-EXP>
                | <ARITH-EXP> <NeqToken> <ARITH-EXP>
                | <CHAR-EXP> <NeqToken> <CHAR-EXP>
                | <BIT-CAT> <NeqToken> <BIT-CAT>
                | <STRUCTURE-EXP> <NeqToken> <STRUCTURE-EXP>
                | <NAME-EXP> <NeqToken> <NAME-EXP>
                | <ARITH-EXP> <<> <ARITH-EXP>
                | <CHAR-EXP> <<> <CHAR-EXP>
                | <BIT-CAT> <<> <BIT-CAT>
                | <STRUCTURE-EXP> <<> <STRUCTURE-EXP>
                | <NAME-EXP> <<> <NAME-EXP>
                | <ARITH-EXP> <>> <ARITH-EXP>
                | <CHAR-EXP> <>> <CHAR-EXP>
                | <BIT-CAT> <>> <BIT-CAT>
                | <STRUCTURE-EXP> <>> <STRUCTURE-EXP>
                | <NAME-EXP> <>> <NAME-EXP>
                | <ARITH-EXP> <LeToken> <ARITH-EXP>
                | <CHAR-EXP> <LeToken> <CHAR-EXP>
                | <BIT-CAT> <LeToken> <BIT-CAT>
                | <STRUCTURE-EXP> <LeToken> <STRUCTURE-EXP>
                | <NAME-EXP> <LeToken> <NAME-EXP>
                | <ARITH-EXP> <GeToken> <ARITH-EXP>
                | <CHAR-EXP> <GeToken> <CHAR-EXP>
                | <BIT-CAT> <GeToken> <BIT-CAT>
                | <STRUCTURE-EXP> <GeToken> <STRUCTURE-EXP>
                | <NAME-EXP> <GeToken> <NAME-EXP>

<ANY-STATEMENT> ::= <STATEMENT>
                  | <BLOCK-DEFINITION>

<ON-PHRASE> ::= ON ERROR <SUBSCRIPT>
               | ON ERROR

<ON-CLAUSE> ::= ON ERROR <SUBSCRIPT> SYSTEM
               | ON ERROR <SUBSCRIPT> IGNORE
               | ON ERROR SYSTEM
               | ON ERROR IGNORE

<LABEL-DEFINITION> ::= <LABEL> :

```

$\langle \text{CALL-KEY} \rangle ::= \text{CALL } \langle \text{LABEL-VAR} \rangle$

$\langle \text{ASSIGN} \rangle ::= \text{ASSIGN}$

$\langle \text{CALL-ASSIGN-LIST} \rangle ::=$

- $\langle \text{VARIABLE} \rangle$
- $| \quad \langle \text{CALL-ASSIGN-LIST} \rangle , \langle \text{VARIABLE} \rangle$
- $| \quad \langle \text{QUAL-STRUCT} \rangle$
- $| \quad \langle \text{CALL-ASSIGN-LIST} \rangle , \langle \text{QUAL-STRUCT} \rangle$

$\langle \text{DO-GROUP-HEAD} \rangle ::=$

- DO ;
- $| \quad \text{DO } \langle \text{FOR-LIST} \rangle ;$
- $| \quad \text{DO } \langle \text{FOR-LIST} \rangle \langle \text{WHILE-CLAUSE} \rangle ;$
- $| \quad \text{DO } \langle \text{WHILE-CLAUSE} \rangle ;$
- $| \quad \text{DO CASE } \langle \text{ARITH-EXP} \rangle ;$
- $| \quad \langle \text{CASE-ELSE} \rangle \langle \text{STATEMENT} \rangle$
- $| \quad \langle \text{DO-GROUP-HEAD} \rangle \langle \text{ANY-STATEMENT} \rangle$
- $| \quad \langle \text{DO-GROUP-HEAD} \rangle \langle \text{TEMPORARY-STMT} \rangle$

$\langle \text{ENDING} \rangle ::=$

- END
- $| \quad \text{END } \langle \text{LABEL} \rangle$
- $| \quad \langle \text{LABEL-DEFINITION} \rangle \langle \text{ENDING} \rangle$

$\langle \text{READ-KEY} \rangle ::=$

- $\text{READ } (\langle \text{NUMBER} \rangle)$
- $| \quad \text{READALL } (\langle \text{NUMBER} \rangle)$

$\langle \text{WRITE-KEY} \rangle ::= \text{WRITE } (\langle \text{NUMBER} \rangle)$

$\langle \text{READ-PHRASE} \rangle ::=$

- $\langle \text{READ-KEY} \rangle \langle \text{READ-ARG} \rangle$
- $| \quad \langle \text{READ-PHRASE} \rangle , \langle \text{READ-ARG} \rangle$

$\langle \text{WRITE-PHRASE} \rangle ::=$

- $\langle \text{WRITE-KEY} \rangle \langle \text{WRITE-ARG} \rangle$
- $| \quad \langle \text{WRITE-PHRASE} \rangle , \langle \text{WRITE-ARG} \rangle$

$\langle \text{READ-ARG} \rangle ::=$

- $\langle \text{VARIABLE} \rangle$
- $| \quad \langle \text{IO-CONTROL} \rangle$

$\langle \text{WRITE-ARG} \rangle ::=$

- $\langle \text{EXPRESSION} \rangle$
- $| \quad \langle \text{IO-CONTROL} \rangle$
- $| \quad \langle \text{StructIdentifierToken} \rangle$

$\langle \text{FILE-EXP} \rangle ::= \langle \text{FILE-HEAD} \rangle , \langle \text{ARITH-EXP} \rangle$

$\langle \text{FILE-HEAD} \rangle ::= \text{FILE } (\langle \text{NUMBER} \rangle$

$\langle \text{IO-CONTROL} \rangle ::=$

- $\text{SKIP } (\langle \text{ARITH-EXP} \rangle)$
- $| \quad \text{TAB } (\langle \text{ARITH-EXP} \rangle)$
- $| \quad \text{COLUMN } (\langle \text{ARITH-EXP} \rangle)$
- $| \quad \text{LINE } (\langle \text{ARITH-EXP} \rangle)$
- $| \quad \text{PAGE } (\langle \text{ARITH-EXP} \rangle)$

$\langle \text{WAIT-KEY} \rangle ::= \text{WAIT}$

$\langle \text{TERMINATOR} \rangle ::=$

- TERMINATE
- $| \quad \text{CANCEL}$

$\langle \text{TERMINATE-LIST} \rangle ::=$

- $\langle \text{LABEL-VAR} \rangle$
- $| \quad \langle \text{TERMINATE-LIST} \rangle , \langle \text{LABEL-VAR} \rangle$

$\langle \text{SCHEDULE-HEAD} \rangle ::= \text{SCHEDULE } \langle \text{LABEL-VAR} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{SCHEDULE-HEAD} \rangle \text{ AT } \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{SCHEDULE-HEAD} \rangle \text{ IN } \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{SCHEDULE-HEAD} \rangle \text{ ON } \langle \text{BIT-EXP} \rangle$
 $\langle \text{SCHEDULE-PHRASE} \rangle ::= \langle \text{SCHEDULE-HEAD} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{SCHEDULE-HEAD} \rangle \text{ PRIORITY } (\langle \text{ARITH-EXP} \rangle)$
 $\quad \quad \quad | \quad \quad \quad \langle \text{SCHEDULE-PHRASE} \rangle \text{ DEPENDENT}$
 $\langle \text{SCHEDULE-CONTROL} \rangle ::= \langle \text{STOPPING} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{TIMING} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{TIMING} \rangle \langle \text{STOPPING} \rangle$
 $\langle \text{TIMING} \rangle ::= \langle \text{REPEAT} \rangle \text{ EVERY } \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{REPEAT} \rangle \text{ AFTER } \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{REPEAT} \rangle$
 $\langle \text{REPEAT} \rangle ::= , \text{ REPEAT}$
 $\langle \text{STOPPING} \rangle ::= \langle \text{WHILE-KEY} \rangle \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{WHILE-KEY} \rangle \langle \text{BIT-EXP} \rangle$
 $\langle \text{SIGNAL-CLAUSE} \rangle ::= \text{SET } \langle \text{EVENT-VAR} \rangle$
 $\quad \quad \quad | \quad \quad \quad \text{RESET } \langle \text{EVENT-VAR} \rangle$
 $\quad \quad \quad | \quad \quad \quad \text{SIGNAL } \langle \text{EVENT-VAR} \rangle$
 $\langle \text{PERCENT-MACRO-NAME} \rangle ::= \% \langle \text{IDENTIFIER} \rangle$
 $\langle \text{PERCENT-MACRO-HEAD} \rangle ::= \langle \text{PERCENT-MACRO-NAME} \rangle ($
 $\quad \quad \quad | \quad \quad \quad \langle \text{PERCENT-MACRO-HEAD} \rangle \langle \text{PERCENT-MACRO-ARG} \rangle ,$
 $\langle \text{PERCENT-MACRO-ARG} \rangle ::= \langle \text{NAME-VAR} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{CONSTANT} \rangle$
 $\langle \text{CASE-ELSE} \rangle ::= \text{DO CASE } \langle \text{ARITH-EXP} \rangle ; \text{ ELSE}$
 $\langle \text{WHILE-KEY} \rangle ::= \text{WHILE}$
 $\quad \quad \quad | \quad \quad \quad \text{UNTIL}$
 $\langle \text{WHILE-CLAUSE} \rangle ::= \langle \text{WHILE-KEY} \rangle \langle \text{BIT-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{WHILE-KEY} \rangle \langle \text{RELATIONAL-EXP} \rangle$
 $\langle \text{FOR-LIST} \rangle ::= \langle \text{FOR-KEY} \rangle \langle \text{ARITH-EXP} \rangle \langle \text{ITERATION-CONTROL} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{FOR-KEY} \rangle \langle \text{ITERATION-BODY} \rangle$
 $\langle \text{ITERATION-BODY} \rangle ::= \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{ITERATION-BODY} \rangle , \langle \text{ARITH-EXP} \rangle$
 $\langle \text{ITERATION-CONTROL} \rangle ::= \text{TO } \langle \text{ARITH-EXP} \rangle$
 $\quad \quad \quad | \quad \quad \quad \text{TO } \langle \text{ARITH-EXP} \rangle \text{ BY } \langle \text{ARITH-EXP} \rangle$
 $\langle \text{FOR-KEY} \rangle ::= \text{FOR } \langle \text{ARITH-VAR} \rangle \langle \text{EQUALS} \rangle$
 $\quad \quad \quad | \quad \quad \quad \text{FOR TEMPORARY } \langle \text{IDENTIFIER} \rangle =$
 $\langle \text{TEMPORARY-STMT} \rangle ::= \text{TEMPORARY } \langle \text{DECLARE-BODY} \rangle ;$
 $\langle \text{CONSTANT} \rangle ::= \langle \text{NUMBER} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{COMPOUND-NUMBER} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{BIT-CONST} \rangle$
 $\quad \quad \quad | \quad \quad \quad \langle \text{CHAR-CONST} \rangle$

$\langle \text{ARRAY-HEAD} \rangle ::= \text{ARRAY (}$
 $\quad | \quad \langle \text{ARRAY-HEAD} \rangle \langle \text{LITERAL-EXP-OR-STAR} \rangle ,$
 $\langle \text{MINOR-ATTR-LIST} \rangle ::= \langle \text{MINOR-ATTRIBUTE} \rangle$
 $\quad | \quad \langle \text{MINOR-ATTR-LIST} \rangle \langle \text{MINOR-ATTRIBUTE} \rangle$
 $\langle \text{MINOR-ATTRIBUTE} \rangle ::= \text{STATIC}$
 $\quad | \quad \text{AUTOMATIC}$
 $\quad | \quad \text{DENSE}$
 $\quad | \quad \text{ALIGNED}$
 $\quad | \quad \text{ACCESS}$
 $\quad | \quad \text{LOCK (} \langle \text{LITERAL-EXP-OR-STAR} \rangle \text{)}$
 $\quad | \quad \text{REMOTE}$
 $\quad | \quad \text{RIGID}$
 $\quad | \quad \langle \text{INIT-OR-CONST-HEAD} \rangle \langle \text{REPEATED-CONSTANT} \rangle \text{)}$
 $\quad | \quad \langle \text{INIT-OR-CONST-HEAD} \rangle * \text{)}$
 $\quad | \quad \text{LATCHED}$
 $\quad | \quad \text{NONHAL (} \langle \text{LEVEL} \rangle \text{)}$
 $\langle \text{INIT-OR-CONST-HEAD} \rangle ::= \text{INITIAL (}$
 $\quad | \quad \text{CONSTANT (}$
 $\quad | \quad \langle \text{INIT-OR-CONST-HEAD} \rangle \langle \text{REPEATED-CONSTANT} \rangle ,$
 $\langle \text{REPEATED-CONSTANT} \rangle ::= \langle \text{EXPRESSION} \rangle$
 $\quad | \quad \langle \text{REPEAT-HEAD} \rangle \langle \text{VARIABLE} \rangle$
 $\quad | \quad \langle \text{REPEAT-HEAD} \rangle \langle \text{CONSTANT} \rangle$
 $\quad | \quad \langle \text{NESTED-REPEAT-HEAD} \rangle \langle \text{REPEATED-CONSTANT} \rangle \text{)}$
 $\quad | \quad \langle \text{REPEAT-HEAD} \rangle$
 $\langle \text{REPEAT-HEAD} \rangle ::= \langle \text{ARITH-EXP} \rangle \#$
 $\langle \text{NESTED-REPEAT-HEAD} \rangle ::= \langle \text{REPEAT-HEAD} \rangle ($
 $\quad | \quad \langle \text{NESTED-REPEAT-HEAD} \rangle \langle \text{REPEATED-CONSTANT} \rangle ,$
 $\langle \text{DCL-LIST-COMMA} \rangle ::= \langle \text{DECLARATION-LIST} \rangle ,$
 $\langle \text{LITERAL-EXP-OR-STAR} \rangle ::= \langle \text{ARITH-EXP} \rangle$
 $\quad | \quad *$
 $\langle \text{TYPE-SPEC} \rangle ::= \langle \text{STRUCT-SPEC} \rangle$
 $\quad | \quad \langle \text{BIT-SPEC} \rangle$
 $\quad | \quad \langle \text{CHAR-SPEC} \rangle$
 $\quad | \quad \langle \text{ARITH-SPEC} \rangle$
 $\quad | \quad \text{EVENT}$
 $\langle \text{BIT-SPEC} \rangle ::= \text{BOOLEAN}$
 $\quad | \quad \text{BIT (} \langle \text{LITERAL-EXP-OR-STAR} \rangle \text{)}$
 $\langle \text{CHAR-SPEC} \rangle ::= \text{CHARACTER (} \langle \text{LITERAL-EXP-OR-STAR} \rangle \text{)}$
 $\langle \text{STRUCT-SPEC} \rangle ::= \langle \text{STRUCT-TEMPLATE} \rangle \langle \text{STRUCT-SPEC-BODY} \rangle$
 $\langle \text{STRUCT-SPEC-BODY} \rangle ::= - \text{STRUCTURE}$
 $\quad | \quad \langle \text{STRUCT-SPEC-HEAD} \rangle \langle \text{LITERAL-EXP-OR-STAR} \rangle \text{)}$
 $\langle \text{STRUCT-TEMPLATE} \rangle ::= \langle \text{STRUCTURE-ID} \rangle$

$\langle \text{STRUCT-SPEC-HEAD} \rangle ::= - \text{STRUCTURE} ($
 $\langle \text{ARITH-SPEC} \rangle ::= \langle \text{PREC-SPEC} \rangle$
 $\quad | \quad \langle \text{SQ-DQ-NAME} \rangle$
 $\quad | \quad \langle \text{SQ-DQ-NAME} \rangle \langle \text{PREC-SPEC} \rangle$
 $\langle \text{COMPILATION} \rangle ::= \langle \text{ANY-STATEMENT} \rangle$
 $\quad | \quad \langle \text{COMPILATION} \rangle \langle \text{ANY-STATEMENT} \rangle$
 $\quad | \quad \langle \text{DECLARE-STATEMENT} \rangle$
 $\quad | \quad \langle \text{COMPILATION} \rangle \langle \text{DECLARE-STATEMENT} \rangle$
 $\quad | \quad \langle \text{STRUCTURE-STMT} \rangle$
 $\quad | \quad \langle \text{COMPILATION} \rangle \langle \text{STRUCTURE-STMT} \rangle$
 $\quad | \quad \langle \text{REPLACE-STMT} \rangle ;$
 $\quad | \quad \langle \text{COMPILATION} \rangle \langle \text{REPLACE-STMT} \rangle ;$
 $\quad | \quad \langle \text{INIT-OR-CONST-HEAD} \rangle \langle \text{EXPRESSION} \rangle)$
 $\langle \text{BLOCK-DEFINITION} \rangle ::= \langle \text{BLOCK-STMT} \rangle \langle \text{CLOSING} \rangle ;$
 $\quad | \quad \langle \text{BLOCK-STMT} \rangle \langle \text{BLOCK-BODY} \rangle \langle \text{CLOSING} \rangle ;$
 $\langle \text{BLOCK-STMT} \rangle ::= \langle \text{BLOCK-STMT-TOP} \rangle ;$
 $\langle \text{BLOCK-STMT-TOP} \rangle ::= \langle \text{BLOCK-STMT-TOP} \rangle \text{ACCESS}$
 $\quad | \quad \langle \text{BLOCK-STMT-TOP} \rangle \text{RIGID}$
 $\quad | \quad \langle \text{BLOCK-STMT-HEAD} \rangle$
 $\quad | \quad \langle \text{BLOCK-STMT-HEAD} \rangle \text{EXCLUSIVE}$
 $\quad | \quad \langle \text{BLOCK-STMT-HEAD} \rangle \text{REENTRANT}$
 $\langle \text{BLOCK-STMT-HEAD} \rangle ::= \langle \text{LABEL-EXTERNAL} \rangle \text{PROGRAM}$
 $\quad | \quad \langle \text{LABEL-EXTERNAL} \rangle \text{COMPOOL}$
 $\quad | \quad \langle \text{LABEL-DEFINITION} \rangle \text{TASK}$
 $\quad | \quad \langle \text{LABEL-DEFINITION} \rangle \text{UPDATE}$
 $\quad | \quad \text{UPDATE}$
 $\quad | \quad \langle \text{FUNCTION-NAME} \rangle$
 $\quad | \quad \langle \text{FUNCTION-NAME} \rangle \langle \text{FUNC-STMT-BODY} \rangle$
 $\quad | \quad \langle \text{PROCEDURE-NAME} \rangle$
 $\quad | \quad \langle \text{PROCEDURE-NAME} \rangle \langle \text{PROC-STMT-BODY} \rangle$
 $\langle \text{LABEL-EXTERNAL} \rangle ::= \langle \text{LABEL-DEFINITION} \rangle$
 $\quad | \quad \langle \text{LABEL-DEFINITION} \rangle \text{EXTERNAL}$
 $\langle \text{CLOSING} \rangle ::= \text{CLOSE}$
 $\quad | \quad \text{CLOSE} \langle \text{LABEL} \rangle$
 $\quad | \quad \text{CLOSE} \langle \text{NoArgUserFuncIdentifierToken} \rangle$
 $\quad | \quad \langle \text{LABEL-DEFINITION} \rangle \langle \text{CLOSING} \rangle$
 $\langle \text{BLOCK-BODY} \rangle ::= \langle \text{DECLARE-GROUP} \rangle$
 $\quad | \quad \langle \text{ANY-STATEMENT} \rangle$
 $\quad | \quad \langle \text{BLOCK-BODY} \rangle \langle \text{ANY-STATEMENT} \rangle$
 $\langle \text{FUNCTION-NAME} \rangle ::= \langle \text{LABEL-EXTERNAL} \rangle \text{FUNCTION}$
 $\quad | \quad \langle \text{NoArgUserFuncIdentifierToken} \rangle : \text{FUNCTION}$
 $\quad | \quad \langle \text{NoArgUserFuncIdentifierToken} \rangle : \text{EXTERNAL FUNCTION}$
 $\langle \text{PROCEDURE-NAME} \rangle ::= \langle \text{LABEL-EXTERNAL} \rangle \text{PROCEDURE}$
 $\langle \text{FUNC-STMT-BODY} \rangle ::= \langle \text{PARAMETER-LIST} \rangle$
 $\quad | \quad \langle \text{TYPE-SPEC} \rangle$
 $\quad | \quad \langle \text{PARAMETER-LIST} \rangle \langle \text{TYPE-SPEC} \rangle$

$\langle \text{PROC-STMT-BODY} \rangle ::= \langle \text{PARAMETER-LIST} \rangle$
 $\quad \quad \quad | \quad \langle \text{ASSIGN-LIST} \rangle$
 $\quad \quad \quad | \quad \langle \text{PARAMETER-LIST} \rangle \langle \text{ASSIGN-LIST} \rangle$
 $\langle \text{DECLARE-GROUP} \rangle ::= \langle \text{DECLARE-ELEMENT} \rangle$
 $\quad \quad \quad | \quad \langle \text{DECLARE-GROUP} \rangle \langle \text{DECLARE-ELEMENT} \rangle$
 $\langle \text{DECLARE-ELEMENT} \rangle ::= \langle \text{DECLARE-STATEMENT} \rangle$
 $\quad \quad \quad | \quad \langle \text{REPLACE-STMT} \rangle ;$
 $\quad \quad \quad | \quad \langle \text{STRUCTURE-STMT} \rangle$
 $\quad \quad \quad | \quad \text{EQUATE EXTERNAL } \langle \text{IDENTIFIER} \rangle \text{ TO } \langle \text{VARIABLE} \rangle ;$
 $\langle \text{PARAMETER} \rangle ::= \langle \text{IdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{BitIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{CharIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{StructIdentifierToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{EventToken} \rangle$
 $\quad \quad \quad | \quad \langle \text{LabelToken} \rangle$
 $\langle \text{PARAMETER-LIST} \rangle ::= \langle \text{PARAMETER-HEAD} \rangle \langle \text{PARAMETER} \rangle)$
 $\langle \text{PARAMETER-HEAD} \rangle ::= ($
 $\quad \quad \quad | \quad \langle \text{PARAMETER-HEAD} \rangle \langle \text{PARAMETER} \rangle ,$
 $\langle \text{DECLARE-STATEMENT} \rangle ::= \text{DECLARE } \langle \text{DECLARE-BODY} \rangle ;$
 $\langle \text{ASSIGN-LIST} \rangle ::= \langle \text{ASSIGN} \rangle \langle \text{PARAMETER-LIST} \rangle$
 $\langle \text{TEXT} \rangle ::= \langle \text{TextToken} \rangle$
 $\langle \text{REPLACE-STMT} \rangle ::= \text{REPLACE } \langle \text{REPLACE-HEAD} \rangle \text{ BY } \langle \text{TEXT} \rangle$
 $\langle \text{REPLACE-HEAD} \rangle ::= \langle \text{IDENTIFIER} \rangle$
 $\quad \quad \quad | \quad \langle \text{IDENTIFIER} \rangle (\langle \text{ARG-LIST} \rangle)$
 $\langle \text{ARG-LIST} \rangle ::= \langle \text{IDENTIFIER} \rangle$
 $\quad \quad \quad | \quad \langle \text{ARG-LIST} \rangle , \langle \text{IDENTIFIER} \rangle$
 $\langle \text{STRUCTURE-STMT} \rangle ::= \text{STRUCTURE } \langle \text{STRUCT-STMT-HEAD} \rangle \langle \text{STRUCT-STMT-TAIL} \rangle$
 $\langle \text{STRUCT-STMT-HEAD} \rangle ::= \langle \text{STRUCTURE-ID} \rangle : \langle \text{LEVEL} \rangle$
 $\quad \quad \quad | \quad \langle \text{STRUCTURE-ID} \rangle \langle \text{MINOR-ATTR-LIST} \rangle : \langle \text{LEVEL} \rangle$
 $\quad \quad \quad | \quad \langle \text{STRUCT-STMT-HEAD} \rangle \langle \text{DECLARATION} \rangle , \langle \text{LEVEL} \rangle$
 $\langle \text{STRUCT-STMT-TAIL} \rangle ::= \langle \text{DECLARATION} \rangle ;$
 $\langle \text{INLINE-DEFINITION} \rangle ::= \langle \text{ARITH-INLINE} \rangle$
 $\quad \quad \quad | \quad \langle \text{BIT-INLINE} \rangle$
 $\quad \quad \quad | \quad \langle \text{CHAR-INLINE} \rangle$
 $\quad \quad \quad | \quad \langle \text{STRUCTURE-EXP} \rangle$
 $\langle \text{ARITH-INLINE} \rangle ::= \langle \text{ARITH-INLINE-DEF} \rangle \langle \text{CLOSING} \rangle ;$
 $\quad \quad \quad | \quad \langle \text{ARITH-INLINE-DEF} \rangle \langle \text{BLOCK-BODY} \rangle \langle \text{CLOSING} \rangle ;$
 $\langle \text{ARITH-INLINE-DEF} \rangle ::= \text{FUNCTION } \langle \text{ARITH-SPEC} \rangle ;$
 $\quad \quad \quad | \quad \text{FUNCTION} ;$

$$\begin{aligned}
\langle \text{BIT-INLINE} \rangle & ::= \langle \text{BIT-INLINE-DEF} \rangle \langle \text{CLOSING} \rangle ; \\
& | \quad \langle \text{BIT-INLINE-DEF} \rangle \langle \text{BLOCK-BODY} \rangle \langle \text{CLOSING} \rangle ; \\
\langle \text{BIT-INLINE-DEF} \rangle & ::= \text{FUNCTION } \langle \text{BIT-SPEC} \rangle ; \\
\langle \text{CHAR-INLINE} \rangle & ::= \langle \text{CHAR-INLINE-DEF} \rangle \langle \text{CLOSING} \rangle ; \\
& | \quad \langle \text{CHAR-INLINE-DEF} \rangle \langle \text{BLOCK-BODY} \rangle \langle \text{CLOSING} \rangle ; \\
\langle \text{CHAR-INLINE-DEF} \rangle & ::= \text{FUNCTION } \langle \text{CHAR-SPEC} \rangle ; \\
\langle \text{STRUC-INLINE-DEF} \rangle & ::= \text{FUNCTION } \langle \text{STRUCT-SPEC} \rangle ;
\end{aligned}$$