

5.2 CIT構文規則(2023年10月)

以下は、CIT構文規則となる。図5.1が範囲、定数、その他の定義で、図5.2が、記号定義である。

<program>	→	<devers> <defuns>
<devers>	→	{ <dever> }
<defuns>	→	{ <defun> }
<defun>	→	(DEFUN <funcname> <type> <defgs> <funcblock>)
<funcname>	→	<identifier>
<funcblock>	→	"{" <devers> <statementlist> "}"
<block>	→	"{" <statementlist> "}"
<statementlist>	→	{ <statement> }
<statement>	→	<if> <switch> <while> <dowhile> <for> <expression> <return> <break>
<expression>	→	<op>
<struct>	→	(STRUCT <usertype> {<demember>})
<union>	→	(UNION <usertype> {<demember>})
<usertype>	→	<identifier>
<demember>	→	(<memname> [<type> <bittype>])
<memname>	→	<identifier>
<dever>	→	(DEVER <varname> <type> <initval> _{opt})
<varname>	→	<identifier>
<defgs>	→	{ <defg> }
<defg>	→	(DEFG <varname> <type>)
<type>	→	{ "(" "*" <arytype> } { <usertype> <typename> }
<arytype>	→	"[" <arysize> _{opt} "]"
<bittype>	→	int : <bitsize>

<typename >	→	char int float double
<fullvarname>	→	<varname> { <next> }
<next>	→	<suffix> "*" "&" <nextmember>
<nextmember>	→	"." <memname>
<suffix>	→	"[" {<digit>} "]"
<initval>	→	<constant> <initvals> <funcname>
<initvals>	→	"{" <initval> {<delimiter> <initval> } "
<callfunc>	→	(<funcname> {<src>}) (<fullvarname> {<src>})
<if>	→	(IF <expression> <block> <elseblock> _{opt})
<elseblock>	→	(ELSE <block>)
<switch>	→	(SWITCH <fullvarname> { <case> <default> })
		繰り返しについて、<case>は、1つ以上となる。<default>は、1つ以下となる
<case>	→	(CASE <constant> <statementlist>)
<default>	→	(DEFAULT <statementlist>)
<while>	→	(WHILE <expression> <block>)
<dowhile>	→	(DOWHILE <block> <expression>)
<for>	→	(FOR <expression> _A <expression> _B <expression> _C <block>)
<return>	→	(RETURN <src>)
<break>	→	(BREAK)
<label>	→	(LABEL <labelname>)
<labelname>	→	<identifier>
<goto>	→	(GOTO <labelname>)
<assignment>	→	[(T(=) <dst> <src>)] [(T(=) <dst> <funcname>)]
<src>	→	<fullvarname> <constant> <callfunc> <op>

<dst>	→	<fullvarname>
<op>	→	<1op> <2op> <3op> <src>
<1op>	→	(T(POST++) <dst>)
<2op>	→	<assignment> (T(+) <op> _L <op> _R) (T(-) <op> _L <op> _R) (T(*) <op> _L <op> _R) (T(/) <op> _L <op> _R) (T(%) <op> _L <op> _R) (T(<<) <op> _L <op> _R) (T(>>) <op> _L <op> _R) (T(==) <op> _L <op> _R) (T(!=) <op> _L <op> _R) (T(<) <op> _L <op> _R) (T(>) <op> _L <op> _R) (T(<=) <op> _L <op> _R) (T(>=) <op> _L <op> _R) (T(&&) <op> _L <op> _R) (T() <op> _L <op> _R) (T(&) <op> _L <op> _R) (T() <op> _L <op> _R) (T(^) <op> _L <op> _R)
<3op>	→	(T(?) <op> <fullvarname> _A <fullvarname> _B)
<identifier>	→	<alpha> { <alpha> <digit> }
<delimiter>	→	" " "\t" "\r\n"
<alpha>	→	a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z -
<digit>	→	0 1 2 3 4 5 6 7 8 9
<constant>	→	<iconstant> <cconstant> <strconstant> <fconstant> <dconstant>
<strconstant>	→	\ " {<ascii_20_7e>} \ "
<cconstant>	→	' <ascii_20_7e> '

<arysize>		1～65535 の数値
<bitsize>		1～32 の数値
<iconstant>		-2147483647 から 2147483647の範囲の数値。0xをつけることにより16進数での指定可能
<fconstant>		1.175494e-38から3.402823e+38の範囲の数値
<dconstant>		2.225074e-308から1.797693e+308の範囲の数値
<ascii_20_7e>		ASCIIのコード範囲0x20～0x7Fのうちどれか

図5.1 範囲、定数、その他の定義

図5.2 CIT構文規則の記号定義

$x \rightarrow r$: x は r で置き換えられる
$(x \ y \ z)$: $(\)$ は終端記号であるが、囲まれた記号間には、スペース、タブ、改行のうちどれかが一つ以上あるものとする。
$x \ y$: 記号列 x の後ろに記号列 y を置いた記号列(接続)
$x \mid y$: 記号列 x または記号列 y (選択)
$\{x\}$: 記号列 x の0回以上の繰り返し(反復)
$\{x\}_1$: 記号列 x の1回以上の繰り返し(反復)
x_{opt}	: $x \mid \varepsilon$ (換言すれば 記号列 x は、指定しない場合がある。)
x_c	: 記号列 x は、 c という特定の x である。
< n >	: 非終端記号
" t "	: " で囲まれた記号 t は、終端記号となる
\"	: " が終端記号となる
$[x \ y]$: $[\]$ で囲まれた部分をひとつにまとめてあつかうものとする
ε	: 空記号

図5.2 CIT構文規則の記号定義