ident	Core identifier
tag	struct/union tag
n, i	
<*impl-const*>	
x, y, *ident*	
intval	integer value
$floatval$	floating value
memval	
member	C struct/union member name
τ	
bty	
annots	
Mem_mem_iv_constraint	
ub-name	
string	

n	
bool	
Loc_t	

memory-order	
linux-memory-order	
thread-id	

oTy	::=		types for C objects
	|	`integer`	
	|	`floating`	
	|	`pointer`	
	|	`array` (oTy)	
	|	`struct` tag	
	|	`union` tag	
bTy	::=		Core base types
	|	`unit`	unit
	|	$bool$	boolean
	|	`ctype`	Core type of C type exprs
	|	$[bTy]$	list
	|	$(\overline{bTy_i}^{\,i})$	tuple
	|	oTy	C object value
	|	`loaded` oTy	oTy or unspecified
	|	`storable`	top type for integer/float/pointer/structs (maybe union?). This is only
$coreTy$	::=		Core types
	|	bTy	pure base type
	|	`eff` bTy	effectful base type
$binop$	::=		binary operators
	|	`+`	
	|	`-`	
	|	`*`	
	|	`/`	
	|	`rem_t`	
	|	`rem_f`	
	|	`^`	
	|	`=`	
	|	`>`	
	|	`<`	
	|	`>=`	
	|	`<=`	
	|	`/\`	
	|	`\/`	
$polarity$	::=		memory action polarities
	|	`Pos`	sequenced by `let weak` and `let strong`
	|	`Neg`	only sequenced by `let strong`
$name$	::=		
	|	$ident$	Core identifier
	|	<$impl\text{-}const$>	implementation-defined constant
$ptrval$	::=		

	|	$\texttt{nullptr}\,(\tau)$	
object_value	::=		C object values
	|	*intval*	integer value
	|	*floatval*	floating-point
	|	*ptrval*	pointer value
	|	$\texttt{array}\,(\overline{loaded\_value_i}^{\,i})$	C array value
	|	$(\texttt{struct}\ tag)\{\,\overline{.member_i : \tau_i = memval_i}^{\,i}\,\}$	C struct value
	|	$(\texttt{union}\ tag)\{.member = memval\}$	C union value
loaded_value	::=		potentially unsp
	|	$\texttt{Specified}\,(object\_value)$	non-unspecifie
	|	$\texttt{Unspecified}\,(\tau)$	unspecified loa
value	::=		Core values
	|	*object_value*	C object value
	|	*loaded_value*	loaded C obje
	|	Unit	
	|	True	
	|	False	
	|	$'\tau'$	C type as valu
	|	$[value_1, .., value_i]$	
	|	$(value_1, .., value_i)$	tuple
ctor	::=		data constructor
	|	Nil *bty*	empty list
	|	Cons	list cons
	|	Tuple	tuple
	|	Array	C array
	|	Ivmax	max integer va
	|	Ivmin	min integer va
	|	Ivsizeof	sizeof value
	|	Ivalignof	alignof value
	|	IvCOMPL	bitwise compl
	|	IvAND	bitwise AND
	|	IvOR	bitwise OR
	|	IvXOR	bitwise XOR
	|	Specified	non-unspecifie
	|	Unspecified	unspecified loa
	|	Fvfromint	cast integer to
	|	Ivfromfloat	cast floating t
maybesym_base_type	::=		
	|	$\_ : bTy$	binders = {}
	|	$ident : bTy$	binders = *ident*
mu_pattern_aux	::=		

	|	$maybesym\_base\_type$
	|	$ctor(\overline{mu\_pattern_i}^{\,i})$
$mu\_pattern$	::=	
	|	$annots\ mu\_pattern\_aux$
mu_pexpr_aux	::=	Core p
	|	$ident$
	|	<*impl-const*> imp
	|	$value$
	|	$\texttt{constrained}(\overline{Mem\_mem\_iv\_constraint_i, ident_i}^{\,i})$ cons
	|	$\texttt{undef}\ Loc\_t(ub\text{-}name)$ und
	|	$\texttt{error}(string, ident)$ imp
	|	$ctor(\overline{ident_i}^{\,i})$ data
	|	$\texttt{array\_shift}(ident_1, \tau, ident_2)$ poi
	|	$\texttt{member\_shift}(ident_1, ident_2, member)$ poi
	|	$\texttt{not}(ident)$ boo
	|	$ident_1\ binop\ ident_2$
	|	$(\texttt{struct}\ ident)\{\overline{.member_i = ident_i}^{\,i}\}$ C st
	|	$(\texttt{union}\ ident_1)\{.member = ident_2\}$ C u
	|	$\texttt{memberof}(ident_1, member, ident_2)$ C st
	|	$name(ident_1, .., ident_n)$ pur
	|	$\texttt{assert\_undef}(ident, ub\text{-}name)$
	|	$\texttt{bool\_to\_integer}(ident)$
	|	$\texttt{conv\_int}(\tau, ident)$
	|	$\texttt{wrapI}(\tau, ident)$
e	::=	
	|	$annots\ bty\ mu\_pexpr\_aux$
mu_tpexpr	::=	Core t
	|	$\texttt{case}\ ident\ \texttt{of}\ \overline{\vert mu\_pattern_i => mu\_tpexpr_i}^{\,i}\ \texttt{end}$ patt
	|	$\texttt{let}\ mu\_pattern = mu\_tpexpr_1\ \in\ mu\_tpexpr_2$ pur
	|	$\texttt{if}\ ident\ \texttt{then}\ mu\_tpexpr_1\ \texttt{else}\ mu\_tpexpr_2$ pur
	|	$\texttt{done}\ ident$ pur
mu_action_aux	::=	memo
	|	$\texttt{create}(e_1, e_2)$
	|	$\texttt{create\_readonly}(e_1, e_2, e_3)$
	|	$\texttt{alloc}(e_1, e_2)$
	|	$\texttt{kill}(bool, e)$ the
	|	$\texttt{store}(bool, e_1, e_2, e_3, memory\text{-}order)$ the
	|	$\texttt{load}(e_1, e_2, memory\text{-}order)$
	|	$\texttt{rmw}(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$
	|	$\texttt{fence}(memory\text{-}order)$
	|	$\texttt{compare\_exchange\_strong}(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$
	|	$\texttt{compare\_exchange\_weak}(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$

	$\mid$	$\texttt{linux\_fence}\,(\textit{linux-memory-order})$	
	$\mid$	$\texttt{linux\_load}\,(e_1, e_2, \textit{linux-memory-order})$	
	$\mid$	$\texttt{linux\_store}\,(e_1, e_2, e_3, \textit{linux-memory-order})$	
	$\mid$	$\texttt{linux\_rmw}\,(e_1, e_2, e_3, \textit{linux-memory-order})$	
mu_action	::=		
	$\mid$	*Loc_t mu_action_aux*	
mu_paction	::=		memory actions with po
	$\mid$	*polarity mu_action*	
	$\mid$	*mu_action*	M positive, sequenced b
	$\mid$	$\neg\,(\textit{mu\_action})$	M negative, only sequen
memop	::=		operations involving the
	$\mid$	*pointer-equality-operator*	pointer equality comp
	$\mid$	*pointer-relational-operator*	pointer relational con
	$\mid$	`ptrdiff`	pointer subtraction
	$\mid$	`intFromPtr`	cast of pointer value
	$\mid$	`ptrFromInt`	cast of integer value t
	$\mid$	`ptrValidForDeref`	dereferencing validity
	$\mid$	`ptrWellAligned`	
	$\mid$	`ptrArrayShift`	
	$\mid$	`memcpy`	
	$\mid$	`memcmp`	
	$\mid$	`realloc`	TODO: not sure abou
	$\mid$	`va_start`	
	$\mid$	`va_copy`	
	$\mid$	`va_arg`	
	$\mid$	`va_end`	
tyvarsym_base_type_pair	::=		
	$\mid$	$\textit{ident} : \textit{bTy}$	
base_type_pexpr_pair	::=		
	$\mid$	$\textit{bTy} := e$	
E	::=		(effectful) expression
	$\mid$	$\texttt{pure}\,(e)$	
	$\mid$	$\texttt{memop}\,(\textit{memop}, e_1, \ldots, e_n)$	pointer op involving
	$\mid$	*mu_paction*	memory action
	$\mid$	$\texttt{case}\,e\,\texttt{with}\,\overline{\mid \textit{mu\_pattern}_i \Rightarrow E_i}^{\,i}\,\texttt{end}$	pattern matching
	$\mid$	$\texttt{let}\,\textit{mu\_pattern} = e\,\texttt{in}\,E$	
	$\mid$	$\texttt{if}\,e\,\texttt{then}\,E_1\,\texttt{else}\,E_2$	
	$\mid$	`skip`	
	$\mid$	$\texttt{ccall}\,(e_1, e_2, \overline{e_i}^{\,i})$	C function call
	$\mid$	$\texttt{pcall}\,(\textit{name}, \overline{e_i}^{\,i})$	Core procedure call
	$\mid$	$\texttt{unseq}\,(E_1, \ldots, E_n)$	unsequenced expressi

	$\mid$	$\texttt{let weak}\ mu\_pattern = E_1\ \in\ E_2$
	$\mid$	$\texttt{let strong}\ mu\_pattern = E_1\ \in\ E_2$
	$\mid$	$\texttt{let atomic}\ tyvarsym\_base\_type\_pair = mu\_action_1\ \in\ mu\_paction_2$
	$\mid$	$\texttt{indet}\,[n](E)$
	$\mid$	$\texttt{bound}\,[n](E)$
	$\mid$	$\texttt{nd}\,(E_1, .., E_n)$
	$\mid$	$\texttt{save}\ tyvarsym\_base\_type\_pair(\ \overline{ident_i : base\_type\_pexpr\_pair_i}^{\,i}\ )\ \in\ E$
	$\mid$	$\texttt{run}\ ident(\ \overline{e_i}^{\,i}\ )$
	$\mid$	$\texttt{par}\,(E_1, .., E_n)$
	$\mid$	$\texttt{wait}\,(thread\text{-}id)$

E	::=	
	$\mid$	$annots\ E$

$terminals$	::=	
	$\mid$	λ
	$\mid$	$\longrightarrow$
	$\mid$	$\rightarrow$
	$\mid$	$\vdash$
	$\mid$	$\in$
	$\mid$	Π
	$\mid$	$\forall$
	$\mid$	$\multimap$
	$\mid$	$\supset$
	$\mid$	Σ
	$\mid$	$\exists$
	$\mid$	$\star$
	$\mid$	$\wedge$
	$\mid$	$\neg$
	$\mid$	$=$

bt	::=	
	$\mid$	

lit	::=	
	$\mid$	$ident$
	$\mid$	$\texttt{Unit}$

$bool\_op$	::=	
	$\mid$	$\neg\ index\_term$
	$\mid$	$index\_term_1 = index\_term_2$

$index\_term\_aux$	::=	
	$\mid$	$bool\_op$

$index\_term$	::=	
	$\mid$	lit

	|	*index_term_aux bt*	
	|	(*index_term*)	S parentheses
arg	::=		argument types
	|	Π *ident* : *bTy.arg*	
	|	$\forall$ *ident* : `logSort` .*arg*	
	|	`resource` $\multimap$ *arg*	
	|	*index_term* $\supset$ *arg*	
	|	`I`	
ret	::=		return types
	|	Σ *ident* : *bTy.ret*	
	|	$\exists$ *ident* : `logSort` .*ret*	
	|	`resource` $\star$ *ret*	
	|	*index_term* $\wedge$ *ret*	
	|	`I`	
Γ	::=		computational var env
	|	`empty`	
	|	$\Gamma, x : bTy$	
Λ	::=		logical var env
	|	`empty`	
	|	Λ, x	
Ξ	::=		constraints env
	|	`empty`	
	|	Ξ, `phi`	
formula	::=		
	|	*judgement*	
	|	`not` (*formula*)	
	|	*ident* : *bTy* $\in \Gamma$	
Jtype	::=		
	|	$\Gamma; \Lambda; \Xi \vdash$ *mu_pexpr_aux* : *ret*	
judgement	::=		
	|	*Jtype*	
user_syntax	::=		
	|	*ident*	
	|	*tag*	
	|	*n*	
	|	<*impl-const*>	
	|	*x*	
	|	*intval*	

| $floatval$
| $memval$
| $member$
| τ
| bty
| $annots$
| $Mem\_mem\_iv\_constraint$
| $ub\text{-}name$
| $string$
|
| n
| $bool$
| Loc_t
|
| $memory\text{-}order$
| $linux\text{-}memory\text{-}order$
| $thread\text{-}id$
| oTy
| bTy
| $coreTy$
| $binop$
| $polarity$
| $name$
| $ptrval$
| $object\_value$
| $loaded\_value$
| $value$
| $ctor$
| $maybesym\_base\_type$
| $mu\_pattern\_aux$
| $mu\_pattern$
| mu_pexpr_aux
| e
| mu_tpexpr
| mu_action_aux
| mu_action
| $mu\_paction$
| $memop$
| $tyvarsym\_base\_type\_pair$
| $base\_type\_pexpr\_pair$
| E
| E
| $terminals$
| bt
| lit
| $bool\_op$

| *index_term_aux*
| *index_term*
| *arg*
| *ret*
| Γ
| Λ
| Ξ
| *formula*

$\boxed{\Gamma; \Lambda; \Xi \vdash \mathit{mu\_pexpr\_aux} : \mathit{ret}}$

$$\frac{x : \mathit{bTy} \in \Gamma}{\Gamma; \Lambda; \Xi \vdash x : \Sigma\, y : \mathit{bTy}.\mathtt{I}} \quad \textsc{GtT\_var}$$

$$\frac{x : \mathit{bool} \in \Gamma}{\Gamma; \Lambda; \Xi \vdash \mathtt{not}\,(x) : \Sigma\, y : \mathit{bool}.y = (\neg\, x\,) \wedge \mathtt{I}} \quad \textsc{GtT\_not}$$

$$\frac{}{\Gamma; \Lambda; \Xi \vdash \mathtt{Unit} : \Sigma\, y : \mathtt{unit}.y = \mathtt{Unit} \wedge \mathtt{I}} \quad \textsc{GtT\_value\_unit}$$

```
Definition rules:        3 good    0 bad
Definition rule clauses: 5 good    0 bad
```