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-const*`>`	implementation-defined constant
ptrval	::=		

	|	`nullptr` (τ)		
object_value	::=			C object values
	|	*intval*		integer value
	|	*floatval*		floating-point
	|	*ptrval*		pointer value
	|	`array` $(\,\overline{loaded\_value_i}^{\,i}\,)$		C array value
	|	$($`struct` $tag)\{\,\overline{.member_i : \tau_i = memval_i}^{\,i}\,\}$		C struct value
	|	$($`union` $tag)\{.member = memval\}$		C union value
loaded_value	::=			potentially unsp
	|	`Specified` $(object\_value)$		non-unspecifie
	|	`Unspecified` (τ)		unspecified lo
value	::=			Core values
	|	*object_value*		C object valu
	|	*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 v
	|	`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 lo
	|	`Fvfromint`		cast integer to
	|	`Ivfromfloat`		cast floating t
maybesym_base_type	::=			
	|	$\_ : bTy$	binders = {}	
	|	$ident : bTy$	binders = *ident*	
mu_pattern_aux	::=			

	$\vert$	*maybesym_base_type*	
	$\vert$	$\mathit{ctor}(\,\overline{\mathit{mu\_pattern}_i}^{\,i}\,)$	
mu_pattern	::=		
	$\vert$	*annots mu_pattern_aux*	
mu_pexpr_aux	::=		Core p
	$\vert$	*ident*	
	$\vert$	<*impl-const*>	imp
	$\vert$	*value*	
	$\vert$	`constrained` $(\,\overline{\mathit{Mem\_mem\_iv\_constraint}_i, \mathit{ident}_i}^{\,i}\,)$	cons
	$\vert$	`undef` $\mathit{Loc\_t}(\mathit{ub\text{-}name})$	und
	$\vert$	`error` $(\mathit{string}, \mathit{ident})$	imp
	$\vert$	$\mathit{ctor}(\,\overline{\mathit{ident}_i}^{\,i}\,)$	data
	$\vert$	`array_shift` $(\mathit{ident}_1, \tau, \mathit{ident}_2)$	poi
	$\vert$	`member_shift` $(\mathit{ident}_1, \mathit{ident}_2, \mathit{member})$	poi
	$\vert$	`not` (ident)	boo
	$\vert$	$\mathit{ident}_1\ \mathit{binop}\ \mathit{ident}_2$	
	$\vert$	(`struct` $\mathit{ident})\{\,\overline{.\mathit{member}_i = \mathit{ident}_i}^{\,i}\,\}$	C st
	$\vert$	(`union` $\mathit{ident}_1)\{.\mathit{member} = \mathit{ident}_2\}$	C u
	$\vert$	`memberof` $(\mathit{ident}_1, \mathit{member}, \mathit{ident}_2)$	C st
	$\vert$	$\mathit{name}(\mathit{ident}_1, \,..\,, \mathit{ident}_n)$	pur
	$\vert$	`assert_undef` $(\mathit{ident}, \mathit{ub\text{-}name})$	
	$\vert$	`bool_to_integer` (ident)	
	$\vert$	`conv_int` (τ, ident)	
	$\vert$	`wrapI` (τ, ident)	
e	::=		
	$\vert$	*annots bty mu_pexpr_aux*	
mu_tpexpr	::=		Core t
	$\vert$	`case` *ident* `of` $\overline{\vert \mathit{mu\_pattern}_i \Rightarrow \mathit{mu\_tpexpr}_i}^{\,i}$ `end`	patt
	$\vert$	`let` $\mathit{mu\_pattern} = \mathit{mu\_tpexpr}_1 \in \mathit{mu\_tpexpr}_2$	pur
	$\vert$	`if` *ident* `then` $\mathit{mu\_tpexpr}_1$ `else` $\mathit{mu\_tpexpr}_2$	pur
	$\vert$	`done` *ident*	pur
mu_action_aux	::=		memo
	$\vert$	`create` (e_1, e_2)	
	$\vert$	`create_readonly` (e_1, e_2, e_3)	
	$\vert$	`alloc` (e_1, e_2)	
	$\vert$	`kill` (bool, e)	the
	$\vert$	`store` $(\mathit{bool}, e_1, e_2, e_3, \mathit{memory\text{-}order})$	the
	$\vert$	`load` $(e_1, e_2, \mathit{memory\text{-}order})$	
	$\vert$	`rmw` $(e_1, e_2, e_3, e_4, \mathit{memory\text{-}order}_1, \mathit{memory\text{-}order}_2)$	
	$\vert$	`fence` $(\mathit{memory\text{-}order})$	
	$\vert$	`compare_exchange_strong` $(e_1, e_2, e_3, e_4, \mathit{memory\text{-}order}_1, \mathit{memory\text{-}order}_2)$	
	$\vert$	`compare_exchange_weak` $(e_1, e_2, e_3, e_4, \mathit{memory\text{-}order}_1, \mathit{memory\text{-}order}_2)$	

	$\vert$	`linux_fence` $(\mathit{linux\text{-}memory\text{-}order})$		
	$\vert$	`linux_load` $(e_1, e_2, \mathit{linux\text{-}memory\text{-}order})$		
	$\vert$	`linux_store` $(e_1, e_2, e_3, \mathit{linux\text{-}memory\text{-}order})$		
	$\vert$	`linux_rmw` $(e_1, e_2, e_3, \mathit{linux\text{-}memory\text{-}order})$		
$\mathit{mu\_action}$	$::=$			
	$\vert$	$\mathit{Loc\_t}\ \mathit{mu\_action\_aux}$		
$\mathit{mu\_paction}$	$::=$			memory actions with po
	$\vert$	$\mathit{polarity}\ \mathit{mu\_action}$		
	$\vert$	$\mathit{mu\_action}$	M	positive, sequenced b
	$\vert$	$\neg\,(\mathit{mu\_action})$	M	negative, only sequen
memop	$::=$			operations involving the
	$\vert$	$\mathit{pointer\text{-}equality\text{-}operator}$		pointer equality comp
	$\vert$	$\mathit{pointer\text{-}relational\text{-}operator}$		pointer relational con
	$\vert$	`ptrdiff`		pointer subtraction
	$\vert$	`intFromPtr`		cast of pointer value
	$\vert$	`ptrFromInt`		cast of integer value t
	$\vert$	`ptrValidForDeref`		dereferencing validity
	$\vert$	`ptrWellAligned`		
	$\vert$	`ptrArrayShift`		
	$\vert$	`memcpy`		
	$\vert$	`memcmp`		
	$\vert$	`realloc`		TODO: not sure abou
	$\vert$	`va_start`		
	$\vert$	`va_copy`		
	$\vert$	`va_arg`		
	$\vert$	`va_end`		
$\mathit{tyvarsym\_base\_type\_pair}$	$::=$			
	$\vert$	$\mathit{ident} : \mathit{bTy}$		
$\mathit{base\_type\_pexpr\_pair}$	$::=$			
	$\vert$	$\mathit{bTy} := e$		
E	$::=$			(effectful) expression
	$\vert$	`pure` (e)		
	$\vert$	`memop` $(\mathit{memop}, e_1, .., e_n)$		pointer op involving
	$\vert$	$\mathit{mu\_paction}$		memory action
	$\vert$	`case` e `with` $\overline{\vert \mathit{mu\_pattern}_i => E_i}^{\,i}$ `end`		pattern matching
	$\vert$	`let` $\mathit{mu\_pattern} = e \in E$		
	$\vert$	`if` e `then` E_1 `else` E_2		
	$\vert$	`skip`		
	$\vert$	`ccall` $(e_1, e_2, \overline{e_i}^{\,i})$		C function call
	$\vert$	`pcall` $(\mathit{name}, \overline{e_i}^{\,i})$		Core procedure call
	$\vert$	`unseq` $(E_1, .., E_n)$		unsequenced expressi

	$\vert$	$\texttt{let}\,\texttt{weak}\, \mathit{mu\_pattern} = E_1 \,\in\, E_2$	weak seq
	$\vert$	$\texttt{let}\,\texttt{strong}\, \mathit{mu\_pattern} = E_1 \,\in\, E_2$	strong se
	$\vert$	$\texttt{let}\,\texttt{atomic}\, \mathit{tyvarsym\_base\_type\_pair} = \mathit{mu\_action}_1 \,\in\, \mathit{mu\_paction}_2$	atomic se
	$\vert$	$\texttt{indet}\,[n](E)$	indeterm
	$\vert$	$\texttt{bound}\,[n](E)$	...and bo
	$\vert$	$\texttt{nd}\,(E_1, \,..\, , E_n)$	nondeter
	$\vert$	$\texttt{save}\, \mathit{tyvarsym\_base\_type\_pair}(\, \overline{\mathit{ident}_i : \mathit{base\_type\_pexpr\_pair}_i}^{\,i}\,) \,\in\, E$	save labe
	$\vert$	$\texttt{run}\, \mathit{ident}(\, \overline{e_i}^{\,i}\,)$	run from
	$\vert$	$\texttt{par}\,(E_1, \,..\, , E_n)$	cppmem-
	$\vert$	$\texttt{wait}\,(\mathit{thread\text{-}id})$	wait for t
E	::=		
	$\vert$	$\mathit{annots}\, E$	
$\mathit{terminals}$	::=		
	$\vert$	λ	
	$\vert$	$\longrightarrow$	
	$\vert$	$\rightarrow$	
	$\vert$	$\vdash$	
	$\vert$	$\in$	
	$\vert$	Π	
	$\vert$	$\forall$	
	$\vert$	$\multimap$	
	$\vert$	$\supset$	
	$\vert$	Σ	
	$\vert$	$\exists$	
	$\vert$	$\star$	
	$\vert$	$\wedge$	
	$\vert$	$\neg$	
bt	::=		OCaml typ
	$\vert$		
lit	::=		
	$\vert$	ident	
$\mathit{bool\_op}$	::=		
	$\vert$	$\neg\, \mathit{index\_term}$	
$\mathit{index\_term\_aux}$	::=		
	$\vert$	$\mathit{bool\_op}$	
$\mathit{index\_term}$	::=		
	$\vert$	lit	
	$\vert$	$\mathit{index\_term\_aux}\ \mathit{bt}$	
arg	::=		argument ty

	$\mid$	$\Pi\, \mathit{ident} : \mathit{bTy}.\mathit{arg}$	
	$\mid$	$\forall\, \mathit{ident} : \texttt{logSort}\,.\mathit{arg}$	
	$\mid$	$\texttt{resource} \multimap \mathit{arg}$	
	$\mid$	$\mathit{index\_term} \supset \mathit{arg}$	
	$\mid$	$\texttt{I}$	
ret	$::=$		return types
	$\mid$	$\Sigma\, \mathit{ident} : \mathit{bTy}.\mathit{ret}$	
	$\mid$	$\exists\, \mathit{ident} : \texttt{logSort}\,.\mathit{ret}$	
	$\mid$	$\texttt{resource} \star \mathit{ret}$	
	$\mid$	$\mathit{index\_term} \wedge \mathit{ret}$	
	$\mid$	$\texttt{I}$	
Γ	$::=$		computational var env
	$\mid$	$\texttt{empty}$	
	$\mid$	$\Gamma, x : \mathit{bTy}$	
Λ	$::=$		logical var env
	$\mid$	$\texttt{empty}$	
	$\mid$	Λ, x	
Ξ	$::=$		constraints env
	$\mid$	$\texttt{empty}$	
	$\mid$	$\Xi, \texttt{phi}$	
$\mathit{formula}$	$::=$		
	$\mid$	$\mathit{judgement}$	
	$\mid$	$\texttt{not}\,(\mathit{formula})$	
	$\mid$	$\mathit{ident} : \mathit{bTy} \in \Gamma$	
Jtype	$::=$		
	$\mid$	$\Gamma; \Lambda; \Xi \vdash \mathit{mu\_pexpr\_aux} : \mathit{ret}$	
$\mathit{judgement}$	$::=$		
	$\mid$	Jtype	
$\mathit{user\_syntax}$	$::=$		
	$\mid$	ident	
	$\mid$	tag	
	$\mid$	n	
	$\mid$	<*impl-const*>	
	$\mid$	x	
	$\mid$	intval	
	$\mid$	$floatval$	
	$\mid$	memval	
	$\mid$	member	
	$\mid$	τ	

$$
\begin{array}{ll}
| & bty \\
| & annots \\
| & Mem\_mem\_iv\_constraint \\
| & ub\text{-}name \\
| & string \\
| & \\
| & n \\
| & bool \\
| & Loc\_t \\
| & \\
| & memory\text{-}order \\
| & linux\text{-}memory\text{-}order \\
| & thread\text{-}id \\
| & oTy \\
| & bTy \\
| & coreTy \\
| & binop \\
| & polarity \\
| & name \\
| & ptrval \\
| & object\_value \\
| & loaded\_value \\
| & value \\
| & ctor \\
| & maybesym\_base\_type \\
| & mu\_pattern\_aux \\
| & mu\_pattern \\
| & mu\_pexpr\_aux \\
| & e \\
| & mu\_tpexpr \\
| & mu\_action\_aux \\
| & mu\_action \\
| & mu\_paction \\
| & memop \\
| & tyvarsym\_base\_type\_pair \\
| & base\_type\_pexpr\_pair \\
| & E \\
| & E \\
| & terminals \\
| & bt \\
| & lit \\
| & bool\_op \\
| & index\_term\_aux \\
| & index\_term \\
| & arg \\
| & ret
\end{array}
$$

$|\quad \Gamma$
$|\quad \Lambda$
$|\quad \Xi$
$|\quad \mathit{formula}$

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

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

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

```
Definition rules:        2 good    0 bad
Definition rule clauses: 4 good    0 bad
```