ident	Core identifier
tag	struct/union tag
n, i	
<*impl-const*>	
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
	|	`boolean`	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 valu
	|	*intval*	integer valu
	|	*floatval*	floating-poi
	|	*ptrval*	pointer valu
	|	$\texttt{array}\,(\overline{loaded\_value_i}^{\,i})$	C array val
	|	$(\texttt{struct}\ tag)\{\,\overline{.member_i : \tau_i = memval_i}^{\,i}\,\}$	C struct va
	|	$(\texttt{union}\ tag)\{.member = memval\}$	C union val
loaded_value	::=		potentially un
	|	$\texttt{Specified}\,(object\_value)$	non-unspec
	|	$\texttt{Unspecified}\,(\tau)$	unspecified
value	::=		Core values
	|	*object_value*	C object va
	|	*loaded_value*	loaded C ob
	|	`Unit`	
	|	`True`	
	|	`False`	
	|	$'\tau'$	C type as v
	|	$[generic\_value1, .., \texttt{generic\_valuei}]$	
	|	$(value_1, .., value_i)$	tuple
ctor	::=		data construc
	|	`Nil` *bty*	empty list
	|	`Cons`	list cons
	|	`Tuple`	tuple
	|	`Array`	C array
	|	`Ivmax`	max integer
	|	`Ivmin`	min integer
	|	`Ivsizeof`	sizeof value
	|	`Ivalignof`	alignof valu
	|	`IvCOMPL`	bitwise com
	|	`IvAND`	bitwise AN
	|	`IvOR`	bitwise OR
	|	`IvXOR`	bitwise XO
	|	`Specified`	non-unspec
	|	`Unspecified`	unspecified
	|	`Fvfromint`	cast integer
	|	`Ivfromfloat`	cast floatin
$'sym$: `core_base_type`	::=		
	|	_ : *bTy*	binders = {}
	|	*ident* : *bTy*	binders = *ident*
generic_pattern_aux	::=		

| | $'sym$: `core_base_type`
| | $ctor(\overline{generic\_pattern_i}^{\,i})$

generic_pattern ::=
| | *annots generic_pattern_aux*

generic_pexpr_aux ::=
| | *ident*
| | <*impl-const*>
| | *value*
| | `constrained` $(\overline{Mem\_mem\_iv\_constraint_i, ident_i}^{\,i})$
| | `undef` $Loc\_t(ub\text{-}name)$
| | `error` $(string, ident)$
| | $ctor(\overline{ident_i}^{\,i})$
| | `array_shift` $(ident_1, \tau, ident_2)$
| | `member_shift` $(ident_1, ident_2, member)$
| | `not` $(ident)$
| | $ident_1\ binop\ ident_2$
| | $(\texttt{struct}\ ident)\{\overline{.member_i = ident_i}^{\,i}\}$
| | $(\texttt{union}\ ident_1)\{.member = ident_2\}$
| | `memberof` $(ident_1, member, ident_2)$
| | $name(ident_1, .., ident_n)$
| | `assert_undef` $(ident, ub\text{-}name)$
| | `bool_to_integer` $(ident)$
| | `conv_int` $(\tau, ident)$
| | `wrapI` $(\tau, ident)$

e ::=
| | *annots bty generic_pexpr_aux*

generic_tpexpr ::=
| | `case` *ident* `of` $\overline{|generic\_pattern_i => generic\_tpexpr_i}^{\,i}$ `end`
| | `let` *generic_pattern* $=$ *generic_tpexpr*$_1$ $\in$ *generic_tpexpr*$_2$
| | `if` *ident* `then` *generic_tpexpr*$_1$ `else` *generic_tpexpr*$_2$
| | `done` *ident*

generic_action_aux ::=
| | `create` (e_1, e_2)
| | `create_readonly` (e_1, e_2, e_3)
| | `alloc` (e_1, e_2)
| | `kill` $(bool, e)$
| | `store` $(bool, e_1, e_2, e_3, memory\text{-}order)$
| | `load` $(e_1, e_2, memory\text{-}order)$
| | `rmw` $(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$
| | `fence` $(memory\text{-}order)$
| | `compare_exchange_strong` $(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$
| | `compare_exchange_weak` $(e_1, e_2, e_3, e_4, memory\text{-}order_1, memory\text{-}order_2)$

	|	`linux_fence` $(\mathit{linux\text{-}memory\text{-}order})$	
	|	`linux_load` $(e_1, e_2, \mathit{linux\text{-}memory\text{-}order})$	
	|	`linux_store` $(e_1, e_2, e_3, \mathit{linux\text{-}memory\text{-}order})$	
	|	`linux_rmw` $(e_1, e_2, e_3, \mathit{linux\text{-}memory\text{-}order})$	
generic_action	::=		
	|	*Loc_t generic_action_aux*	
generic_paction	::=		memory actions with p
	|	*polarity generic_action*	
	|	*generic_action*	M positive, sequenced
	|	`neg` $(\mathit{generic\_action})$	M negative, only seque
memop	::=		operations involving th
	|	*pointer-equality-operator*	pointer equality con
	|	*pointer-relational-operator*	pointer relational co
	|	`ptrdiff`	pointer subtraction
	|	`intFromPtr`	cast of pointer value
	|	`ptrFromInt`	cast of integer value
	|	`ptrValidForDeref`	dereferencing validit
	|	`ptrWellAligned`	
	|	`ptrArrayShift`	
	|	`memcpy`	
	|	`memcmp`	
	|	`realloc`	TODO: not sure ab
	|	`va_start`	
	|	`va_copy`	
	|	`va_arg`	
	|	`va_end`	
tyvarsym_base_type_pair	::=		
	|	$\mathit{ident} : \mathit{bTy}$	
core_base_type_pexpr_pair	::=		
	|	$\mathit{bTy} := e$	
E	::=		(effectful) expression
	|	`pure` (e)	
	|	`memop` $(\mathit{memop}, e_1, .., e_n)$	pointer op involving
	|	*generic_paction*	memory action
	|	`case` e `with` $\overline{\mid \mathit{generic\_pattern}_i \Rightarrow E_i}^{\,i}$ `end`	pattern matching
	|	`let` $\mathit{generic\_pattern} = e \in E$	
	|	`if` e `then` E_1 `else` E_2	
	|	`skip`	
	|	`ccall` $(e_1, e_2, \overline{e_i}^{\,i})$	C function call
	|	`pcall` $(\mathit{name}, \overline{e_i}^{\,i})$	Core procedure call
	|	`unseq` $(E_1, .., E_n)$	unsequenced expres

	$\mid$	$\texttt{let weak}\ \mathit{generic\_pattern} = E_1 \in E_2$
	$\mid$	$\texttt{let strong}\ \mathit{generic\_pattern} = E_1 \in E_2$
	$\mid$	$\texttt{let atomic}\ \mathit{tyvarsym\_base\_type\_pair} = \mathit{generic\_action}_1 \in \mathit{generic\_paction}_2$
	$\mid$	$\texttt{indet}\,[n](E)$
	$\mid$	$\texttt{bound}\,[n](E)$
	$\mid$	$\texttt{nd}\,(E_1, .., E_n)$
	$\mid$	$\texttt{save}\ \mathit{tyvarsym\_base\_type\_pair}(\ \overline{\mathit{ident}_i : \mathit{core\_base\_type\_pexpr\_pair}_i}^{\,i}\ ) \in E$
	$\mid$	$\texttt{run}\ \mathit{ident}(\ \overline{e_i}^{\,i}\ )$
	$\mid$	$\texttt{par}\,(E_1, .., E_n)$
	$\mid$	$\texttt{wait}\,(\mathit{thread\text{-}id})$
E	::=	
	$\mid$	$\mathit{annots}\ E$
Γ	::=	
	$\mid$	$\texttt{empty}$
	$\mid$	$\Gamma, \texttt{x} : \mathit{bTy}$
terminals	::=	
	$\mid$	λ
	$\mid$	$\longrightarrow$
	$\mid$	$\rightarrow$
	$\mid$	$\vdash$
	$\mid$	$\in$
formula	::=	
	$\mid$	*judgement*
	$\mid$	$\texttt{not}\,(\mathit{formula})$
	$\mid$	$\mathit{ident} : \mathit{bTy} \in \Gamma$
Jtype	::=	
	$\mid$	$\Gamma \vdash \mathit{generic\_pexpr\_aux} : \mathit{bTy}$
judgement	::=	
	$\mid$	*Jtype*
user_syntax	::=	
	$\mid$	*ident*
	$\mid$	*tag*
	$\mid$	n
	$\mid$	<*impl-const*>
	$\mid$	*ident*
	$\mid$	*intval*
	$\mid$	*floatval*
	$\mid$	*memval*
	$\mid$	*member*
	$\mid$	τ

| *bty*
| *annots*
| *Mem_mem_iv_constraint*
| *ub-name*
| *string*
|
| *n*
| *bool*
| *Loc_t*
|
| *memory-order*
| *linux-memory-order*
| *thread-id*
| *oTy*
| *bTy*
| *coreTy*
| *binop*
| *polarity*
| *name*
| *ptrval*
| *object_value*
| *loaded_value*
| *value*
| *ctor*
| $'sym$: `core_base_type`
| *generic_pattern_aux*
| *generic_pattern*
| *generic_pexpr_aux*
| *e*
| *generic_tpexpr*
| *generic_action_aux*
| *generic_action*
| *generic_paction*
| *memop*
| *tyvarsym_base_type_pair*
| *core_base_type_pexpr_pair*
| *E*
| *E*
| Γ
| *terminals*
| *formula*

$\boxed{\Gamma \vdash \mathit{generic\_pexpr\_aux} : \mathit{bTy}}$

$$\frac{\mathit{ident} : \texttt{boolean} \in \Gamma}{\Gamma \vdash \texttt{not}\,(\mathit{ident}) : \texttt{boolean}}\ \textsc{GtT\_value\_name}$$

```
Definition rules:        1 good    0 bad
Definition rule clauses: 2 good    0 bad
```