

# VideoCore IV

## Typed Assembly Language

### Version 0.1

---

El Pin Al

## Index

1. Syntax	1
2. Type system	3
2.1. Propositions	4
2.2. Memory representaion	5
2.3. Typing rules	5
2.4. Auxiliary functions	12
3. Future	13

## 1. Syntax

To define the syntax, there are some primitive terms:

- $i$  denotes an integer.
- $i4$  denotes a 4-bit integer in range  $[0, 15]$ .
- $i4^*$  denotes a 4-bit integer in range  $[1, 16]$ .
- $i7^*$  denotes a 7-bit integer in range  $[1, 128]$ .
- $f$  denotes a floating-point number.

- $l$  denotes a label.
- $nat$  denotes an integer which is greater than or equal to 0.
- In general,  $\varepsilon$  denotes an empty construct.
- $\alpha$  denotes a variable.

The syntax is given below.

$\nu ::= n \mid rr$	operands
$n ::= i \mid f$	numbers
$r ::= rr \mid wr$	registers
$rr ::= rwr \mid \text{uniform} \mid \text{element\_number} \mid \text{vpmr}$	readable registers
$urr ::= a \mid \text{element\_number}$	unconstrained readable registers
$wr ::= rwr \mid \text{uniforms\_address} \mid tmu \mid \text{broadcast} \mid \text{vpmw}$	writable registers
$rwr ::= ga \mid rf$	both readable and writable registers
$a ::= ga \mid sa$	accumulators
$ga ::= r0 \mid r1 \mid r2 \mid r3$	general-purpose accumulators
$sa ::= r4 \mid r5$	special-purpose accumulators
$rf ::= A \mid B$	register files
$A ::= ra0 \mid \dots \mid ra31$	locations in register file A
$B ::= rb0 \mid \dots \mid rb31$	locations in register file B
$tmu ::= tmu0 \mid tmu1$	TMU
$vpmq ::= \varepsilon \mid (Y, i) :: vpmq$	VPM read queues
$Y ::= y_0 \mid \dots \mid y_{63}$	VPM Y

$\iota ::= \text{rotate}(wr, rr, i4) \mid \text{mov}(wr, \nu) \mid \text{setup\_vpm\_read}(Y, i4^*)$ $\mid \text{setup\_vpm\_write}(Y) \mid \text{setup\_dma\_load}(Y, i4^*) \mid \text{start\_dma\_load}(rr)$ $\mid \text{wait\_dma\_load} \mid \text{setup\_dma\_store}(Y, i7^*) \mid \text{start\_dma\_store}(rr)$ $\mid \text{wait\_dma\_store}$	instructions
$cc ::= Z \mid N \mid C$	condition classes
$c ::= \text{set}(cc) \mid \text{clear}(cc)$	conditions
$qc ::= \text{all}(c) \mid \text{any}(c)$	quantified conditions
$s ::= \varepsilon \mid \text{load}\langle tmu \rangle$	signals
$ci ::= (\iota, \varepsilon) \mid (\iota, c)$	conditional instructions
$csi ::= (ci, s)$	conditional instructions with signals
$I ::= \text{jmp}(l) ; csi ; csi ; csi \mid (ci, \text{thread\_end}) ; csi ; csi \mid csi ; I$ $\mid \text{if } qc \text{ jmp}(l) ; csi ; csi ; csi ; I$	instruction sequences
$R ::= \varepsilon \mid R[rr \mapsto arr]$	register contexts
$arr ::= [n, n, n, n, n, n, n, n, n, n, n, n, n, n, n, n]$	arrays
$U ::= \varepsilon \mid n \circ U$	uniforms
$T ::= \varepsilon \mid arr \mid arr :: arr \mid arr :: arr :: arr \mid arr :: arr :: arr :: arr$	TMU
$V ::= \varepsilon \mid V[Y \mapsto at]$	VPM states
$P ::= \varepsilon \mid P[l \mapsto I]$	programs
$M ::= (R, U, T, V, vpmq, P, I)$	machines

## 2. Type system

The type syntax is defined as follows:

$\tau ::= b \mid at \mid vt \mid \Psi$	types
--	-------

$p ::= nat \mid \alpha \mid p + p \mid nat \times p$	pointers
$b ::= int(i) \mid int(?) \mid float \mid ptr(p) \mid code(\Theta)$	basic types
$at ::= [b, b, b, b, b, b, b, b, b, b, b, b, b, b]$	array types
$vt ::= vec(b, \alpha)$	vector types
$\Gamma ::= \varepsilon \mid \Gamma[rr \mapsto at]$	register context types
$\Psi ::= \varepsilon \mid b \circ \Psi$	uniforms types
$\Sigma ::= \varepsilon \mid at \mid at :: at \mid at :: at :: at \mid at :: at :: at :: at$	TMU types
$\Pi ::= \varepsilon \mid \Pi[Y \mapsto at]$	VPM state types
$vpmw\_addr ::= \varepsilon \mid Y$	VPM write addresses
$C ::= (vpmq, \Pi, vpmw\_addr)$	VPM compound types
$u ::= u0 \mid u1 \mid u2$	uniform-access countdowns
$\Theta ::= (\Gamma, \Psi, u, \Sigma, C, \Omega, wr)$	state types
$\Phi ::= \varepsilon \mid \Phi[l \mapsto \Theta]$	program types
$DLS ::= \varepsilon \mid (Y, i4^*)$	DMA load setups
$DSS ::= \varepsilon \mid (Y, i7^*)$	DMA store setups
$DL ::= \varepsilon \mid (Y, i4^*, \tau)$	DMA loads
$DS ::= \varepsilon \mid (Y, i7^*, \tau) \mid \text{type\_preserving}$	DMA stores

## 2.1. Propositions

$p_1 .. p_2$  represents a range  $[p_1, p_2)$ .

$\varphi ::= p .. p \mid \varphi \vee \varphi$	propositions
--	--------------

## 2.2. Memory representaion

$\Omega ::= \{\Xi \mid \varphi\}$  memory subset types

$\Xi ::= \varepsilon \mid \Xi[p \mapsto \tau]$  memory types

The evaluation rules of memory subset types and memory types are given below. The equality rules for  $p$  are not defined here. There are the abuses of notations of a form  $x[y \mapsto z]$ .

$$\frac{p \text{ is in } \varphi}{\{\Xi \mid \varphi\}(p) \rightarrow_{\Omega} \Xi(p)}$$

$$\frac{p = p_1}{\Xi[p_1 \mapsto \tau](p) \rightarrow_{\Xi} \tau}$$

$$\frac{p \neq p_1}{\Xi[p_1 \mapsto \tau](p) \rightarrow_{\Xi} \Xi(p)}$$

## 2.3. Typing rules

Typing rules are defined as follows. Note that  $\text{dom}(\Gamma)$  represents the domain of a context  $\Gamma$ , a map from read / write registers to array types.

numbers:

$$\vdash i : \text{int}(i) \quad \vdash f : \text{float}$$

subtype relations:

$$\text{int}(i) <: \text{int}(?) \quad \text{int}(?) <: \text{int}(?) \quad \frac{\forall i. \text{at}_1[i] <: \text{at}_2[i]}{\text{at}_1 <: \text{at}_2}$$

$$\frac{m \leq n}{\text{vec}(b, m) <: \text{vec}(b, n)}$$

$$\Gamma <: \Gamma \quad \frac{\Gamma_1 <: \Gamma_2}{\Gamma_1[rr \mapsto at] <: \Gamma_2}$$

pointers:

$$\frac{p \in \text{dom}(\Xi)}{\Xi \vdash p : \Xi(p)}$$

well-formed memory subsets:

$$\frac{\forall p \in \text{dom}(\Xi) . p \dots (p + \text{size\_of}(\Xi(p))) \text{ is in } \varphi}{\vdash \{\Xi \mid \varphi\}}$$

registers:

$$\frac{rr \in \text{dom}(\Gamma)}{\Gamma \vdash rr : \Gamma(rr)} \quad \frac{\Psi_1 \equiv b \circ \Psi_2}{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2}$$

$$\vdash \text{element\_number} : [\text{int}(0), \text{int}(1), \dots, \text{int}(15)]$$

$$\frac{vpmq_1 \equiv (vpmq_2 :: (Y, i)) \quad i \geq 2 \quad \Pi \vdash Y : at}{\Pi \vdash \text{vpmr} : at \mid vpmq_1 \rightarrow vpmq_2 :: (\text{inc}(Y), i - 1)}$$

$$\frac{vpmq_1 \equiv (vpmq_2 :: (Y, 1)) \quad \Pi \vdash Y : at}{\Pi \vdash \text{vpmr} : at \mid vpmq_1 \rightarrow vpmq_2}$$

VPM:

$$\frac{Y \in \text{dom}(\Pi)}{\Pi \vdash Y : \Pi(Y)}$$

$$\frac{\Pi \vdash Y : at \quad at <: vt}{\Pi \vdash (Y, 1) : vt}$$

$$\frac{\Pi \vdash Y : at \quad at <: vt_1 \quad n \leq 62 \quad i7^* \geq 2 \quad \Pi \vdash (y_{n+1}, i7^* - 1) : vt_2}{\Pi \vdash (y_n, i7^*) : \text{concat}(vt_1, vt_2)}$$

instructions:

$$\frac{rr \neq wr_{\text{before}} \quad rr \in \text{dom}(\Gamma) \quad at \equiv \text{rotate}(\Gamma(rr), i4)}{wr_{\text{before}} \vdash \text{rotate}(rwr, rr, i4) : \Gamma \rightarrow \Gamma[rwr \mapsto at] ; rwr}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2 \quad at \equiv \text{array}(b)}{u0 \vdash \text{rotate}(rwr, \text{uniform}, i4) : \Gamma \rightarrow \Gamma[rwr \mapsto at] \mid \Psi_1 \rightarrow \Psi_2 ; rwr}$$

$$\frac{\vdash \text{element\_number} : at}{\vdash \text{rotate}(rwr, \text{element\_number}, i4) : \Gamma \rightarrow \Gamma[rwr \mapsto \text{rotate}(at, i4)] ; rwr}$$

$$\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2}{\Pi \vdash \text{rotate}(rwr, \text{vpmr}, i4) : \text{vpmq}_1 \rightarrow \text{vpmq}_2 \mid \Gamma \rightarrow \Gamma[rwr \mapsto \text{rotate}(at, i4)] ; rwr}$$

$$\frac{rr \not\equiv wr_{before} \quad \Gamma \vdash rr : at \quad fst(\text{rotate}(at, i4)) \equiv b}{\Gamma ; wr_{before} \vdash \text{rotate}(\text{broadcast}, rr, i4) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)]}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2}{\vdash \text{rotate}(\text{broadcast}, \text{uniform}, i4) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)] \mid \Psi_1 \rightarrow \Psi_2}$$

$$\frac{\vdash \text{element\_number} : at \quad b \equiv fst(\text{rotate}(at, i4))}{\vdash \text{rotate}(\text{broadcast}, \text{element\_number}, i4) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)]}$$

$$\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \quad fst(\text{rotate}(at, i4)) \equiv b}{\Pi \vdash \text{rotate}(\text{broadcast}, \text{vpmr}, i4) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)] \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2}$$

$$\frac{rr \not\equiv wr_{before} \quad \Gamma \vdash rr : at \quad fst(\text{rotate}(at, i4)) \equiv \text{ptr}(p) \quad \Xi \vdash p : \Psi_2}{\Gamma ; \Xi ; wr_{before} \vdash \text{rotate}(\text{uniforms\_address}, rr, i4) : \Psi_1 \rightarrow \Psi_2 \mid u \rightarrow u2}$$

$$\frac{\vdash \text{uniform} : \text{ptr}(p) \mid \Psi_1 \rightarrow \Psi_2 \quad \Xi \vdash p : \Psi_3}{\Xi \vdash \text{rotate}(\text{uniforms\_address}, \text{uniform}, i4) : \Psi_1 \rightarrow \Psi_3 \mid u0 \rightarrow u2}$$

$$\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \quad fst(\text{rotate}(at, i4)) \equiv \text{ptr}(p) \quad \Xi \vdash p : \Psi_2}{\Pi ; \Xi \vdash \text{rotate}(\text{uniforms\_address}, \text{vpmr}, i4) : \Psi_1 \rightarrow \Psi_2 \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \mid u \rightarrow u2}$$

$$\frac{rr \not\equiv wr_{before} \quad \Gamma \vdash rr : at \quad notfull(\Sigma^{tmu})}{\Gamma ; \Xi ; wr_{before} \vdash \text{rotate}(tmu, rr, i4) : \Sigma^{tmu} \rightarrow \text{rotate}(\text{map}(\text{unwrap}_{\Xi}, at), i4) :: \Sigma^{tmu}}$$

$$\frac{\vdash \text{uniform} : \text{ptr}(p) \mid \Psi_1 \rightarrow \Psi_2 \quad \Xi \vdash p : b \quad notfull(\Sigma^{tmu})}{u0 ; \Xi \vdash \text{rotate}(tmu, \text{uniform}, i4) : \Sigma^{tmu} \rightarrow \text{array}(b) :: \Sigma^{tmu} \mid \Psi_1 \rightarrow \Psi_2}$$

$$\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \quad \text{not\_full}(\Sigma^{tmu})}{\Pi; \Xi \vdash \text{rotate}(tmu, \text{vpmr}, i4) : \Sigma^{tmu} \rightarrow \text{rotate}(\text{map}(\text{unwrap}_{\Xi}, at), i4) :: \Sigma^{tmu}}$$

$$\frac{rr \not\equiv wr_{\text{before}} \quad rr \in \text{dom}(\Gamma) \quad at \equiv \text{rotate}(\Gamma(rr), i4)}{wr_{\text{before}} \vdash \text{rotate}(\text{vpmw}, rr, i4) : \Pi \rightarrow \Pi[Y \mapsto at] \mid Y \rightarrow \text{inc}(Y)}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2 \quad at \equiv \text{array}(b)}{u0 \vdash \text{rotate}(\text{vpmw}, \text{uniform}, i4) : \Pi \rightarrow \Pi[Y \mapsto at] \mid \Psi_1 \rightarrow \Psi_2 \mid Y \rightarrow \text{inc}(Y)}$$

$$\frac{\vdash \text{element\_number} : at}{\vdash \text{rotate}(\text{vpmw}, \text{element\_number}, i4) : \Pi \rightarrow \Pi[Y \mapsto \text{rotate}(at, i4)] \mid Y \rightarrow \text{inc}(Y)}$$

$$\frac{\vdash n : b}{\vdash \text{mov}(rwr, n) : \Gamma \rightarrow \Gamma[rwr \mapsto \text{array}(b)] ; rwr}$$

$$\frac{\Gamma \vdash urr : at}{\vdash \text{mov}(rwr, urr) : \Gamma \rightarrow \Gamma[rwr \mapsto at] ; rwr}$$

$$\frac{\Gamma \vdash rf : at \quad rf \not\equiv wr_{\text{before}}}{wr_{\text{before}} \vdash \text{mov}(rwr, rf) : \Gamma \rightarrow \Gamma[rwr \mapsto at] ; rwr}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2}{u0 \vdash \text{mov}(rwr, \text{uniform}) : \Gamma \rightarrow \Gamma[rwr \mapsto \text{array}(b)] \mid \Psi_1 \rightarrow \Psi_2 ; rwr}$$

$$\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2}{\Pi \vdash \text{mov}(rwr, \text{vpmr}) : \Gamma \rightarrow \Gamma[rwr \mapsto at] \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 ; rwr}$$

$$\frac{\vdash n : b}{\vdash \text{mov}(\text{broadcast}, n) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)]}$$

$$\frac{\Gamma \vdash urr : at \quad \text{fst}(at) \equiv b}{\Gamma \vdash \text{mov}(\text{broadcast}, urr) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)]}$$

$$\frac{\Gamma \vdash rf : at \quad \text{fst}(at) \equiv b \quad rf \not\equiv wr_{\text{before}}}{\Gamma; wr_{\text{before}} \vdash \text{mov}(\text{broadcast}, rf) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)]}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2}{u0 \vdash \text{mov}(\text{broadcast}, \text{uniform}) : \Gamma \rightarrow \Gamma[r5 \mapsto \text{array}(b)] \mid \Psi_1 \rightarrow \Psi_2}$$



$$\begin{array}{c}
\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2}{\Pi \vdash \text{mov}(\text{broadcast}, \text{vpmr}) : \Gamma \rightarrow \Gamma[r5 \mapsto at] \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2} \\
\\
\frac{\Gamma \vdash a : at \quad fst(at) \equiv \text{ptr}(p) \quad \Xi \vdash p : \Psi_2}{\Gamma; \Xi \vdash \text{mov}(\text{uniforms\_address}, a) : \Psi_1 \rightarrow \Psi_2 \mid u \rightarrow u2} \\
\\
\frac{\Gamma \vdash rf : at \quad fst(at) \equiv \text{ptr}(p) \quad \Xi \vdash p : \Psi_2 \quad rf \not\equiv wr_{before}}{\Gamma; \Xi; wr_{before} \vdash \text{mov}(\text{uniforms\_address}, rf) : \Psi_1 \rightarrow \Psi_2 \mid u \rightarrow u2} \\
\\
\frac{\vdash \text{uniform} : \text{ptr}(p) \mid \Psi_1 \rightarrow \Psi_2 \quad \Xi \vdash p : \Psi_3}{\Xi \vdash \text{mov}(\text{uniforms\_address}, \text{uniform}) : \Psi_1 \rightarrow \Psi_3 \mid u0 \rightarrow u2} \\
\\
\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \quad fst(at) \equiv \text{ptr}(p) \quad \Xi \vdash p : \Psi_2}{\Pi; \Xi \vdash \text{mov}(\text{uniforms\_address}, \text{vpmr}) : \text{vpmq}_1 \rightarrow \text{vpmq}_2 \mid \Psi_1 \rightarrow \Psi_2 \mid u \rightarrow u2} \\
\\
\frac{\Gamma \vdash a : at \quad notfull(\Sigma^{tmu})}{\Gamma; \Xi \vdash \text{mov}(tmu, a) : \Sigma^{tmu} \rightarrow map(unwrap_{\Xi}, at) :: \Sigma^{tmu}} \\
\\
\frac{\Gamma \vdash rf : at \quad notfull(\Sigma^{tmu}) \quad rf \not\equiv wr_{before}}{\Gamma; \Xi; wr_{before} \vdash \text{mov}(tmu, rf) : \Sigma^{tmu} \rightarrow map(unwrap_{\Xi}, at) :: \Sigma^{tmu}} \\
\\
\frac{\vdash \text{uniform} : \text{ptr}(p) \mid \Psi_1 \rightarrow \Psi_2 \quad \Xi \vdash p : b \quad notfull(\Sigma^{tmu})}{u0; \Xi \vdash \text{mov}(tmu, \text{uniform}) : \Sigma^{tmu} \rightarrow array(b) :: \Sigma^{tmu} \mid \Psi_1 \rightarrow \Psi_2} \\
\\
\frac{\Pi \vdash \text{vpmr} : at \mid \text{vpmq}_1 \rightarrow \text{vpmq}_2 \quad notfull(\Sigma^{tmu})}{\Pi; \Xi \vdash \text{mov}(tmu, \text{vpmr}) : \text{vpmq}_1 \rightarrow \text{vpmq}_2 \mid \Sigma^{tmu} \rightarrow map(unwrap_{\Xi}, at) :: \Sigma^{tmu}} \\
\\
\frac{\vdash n : b}{\vdash \text{mov}(\text{vpmw}, n) : \Pi \rightarrow \Pi[Y \mapsto array(b)] \mid Y \rightarrow inc(Y)} \\
\\
\frac{\Gamma \vdash urr : at}{\vdash \text{mov}(\text{vpmw}, urr) : \Pi \rightarrow \Pi[Y \mapsto at] \mid Y \rightarrow inc(Y)} \\
\\
\frac{\Gamma \vdash rf : at \quad rf \not\equiv wr_{before}}{wr_{before} \vdash \text{mov}(\text{vpmw}, rf) : \Pi \rightarrow \Pi[Y \mapsto at] \mid Y \rightarrow inc(Y)}
\end{array}$$

$$\frac{\vdash \text{uniform} : b \mid \Psi_1 \rightarrow \Psi_2}{\text{u0} \vdash \text{mov}(\text{vpwmw}, \text{uniform}) : \Pi \rightarrow \Pi[Y \mapsto \text{array}(b)] \mid \Psi_1 \rightarrow \Psi_2 \mid Y \rightarrow \text{inc}(Y)}$$

$$\frac{\Pi \vdash Y : at}{\Pi \vdash \text{setup\_vpm\_read}(Y, i4^*) : \text{vpmq} \rightarrow (Y, i4^*) :: \text{vpmq}}$$

$$\vdash \text{setup\_vpm\_write}(Y) : \text{vpwmw\_addr} \rightarrow Y$$

$$\frac{i + i4^* \leq 64}{\vdash \text{setup\_dma\_load}(y_i, i4^*) : DLS \rightarrow (y_i, i4^*)}$$

$$\frac{\Gamma \vdash rr : at \quad \text{fst}(at) \equiv \text{ptr}(p) \quad \Xi \vdash p : \tau \quad i4^* \times 16 \times 4 \leq \text{size\_of}(\tau)}{(Y, i4^*); \Xi \vdash \text{start\_dma\_load}(rr) : \varepsilon \rightarrow (Y, i4^*, \tau)}$$

$$\vdash \text{wait\_dma\_load} : (Y, i4^*, \tau) \rightarrow \varepsilon \mid \Pi \rightarrow \Pi \text{ dma\_load}(Y, i4^*, \tau)$$

$$\frac{i + i7^* \leq 64}{\vdash \text{setup\_dma\_store}(y_i, i7^*) : DSS \rightarrow (y_i, i7^*)}$$

$$\frac{\Gamma \vdash rr : at_1 \quad \text{fst}(at_1) \equiv \text{ptr}(p) \quad \Omega \vdash p : \tau \quad \Pi \vdash (Y, i7^*) : \tau_1 \quad \tau_1 <: \tau}{(Y, i7^*); \Omega \vdash \text{start\_dma\_store}(rr) : \varepsilon \rightarrow \text{type\_preserving}}$$

$$\frac{\Gamma \vdash rr : at_1 \quad \text{fst}(at_1) \equiv \text{ptr}(p) \quad \Pi \vdash (Y, i7^*) : \tau \quad p \notin \text{dom}(\Omega) \quad \vdash \Omega[p \mapsto \tau]}{(Y, i7^*); \Omega \vdash \text{start\_dma\_store}(rr) : \varepsilon \rightarrow (p, \tau)}$$

$$\vdash \text{wait\_dma\_store} : \text{type\_preserving} \rightarrow \varepsilon$$

$$\vdash \text{wait\_dma\_store} : (p, \tau) \rightarrow \varepsilon \mid \Omega \rightarrow \Omega[p \mapsto \tau]$$

conditional instructions:

$$\frac{\begin{array}{c} \vdash \iota : \Gamma_1 \rightarrow \Gamma_2 \mid u_1 \rightarrow u_2 \mid \Psi_1 \rightarrow \Psi_2 \mid \Sigma_1^{tmu} \rightarrow \Sigma_2^{tmu} \mid C_1 \rightarrow C_2 \mid wr_1 \rightarrow wr_2 \\ [rr \mapsto at_1] \in \Gamma_1 \quad \Gamma_2 \equiv \Gamma_{21}[rr \mapsto at_2] \\ at_1 <: at_2 \end{array}}{\vdash (\iota, c) : \Gamma_1 \rightarrow \Gamma_2 \mid u_1 \rightarrow u_2 \mid \Psi_1 \rightarrow \Psi_2 \mid \Sigma_1^{tmu} \rightarrow \Sigma_2^{tmu} \mid C_1 \rightarrow C_2 \mid wr_1 \rightarrow wr_2}$$

$$\frac{\vdash \iota : \Theta_1 \rightarrow \Theta_2}{\vdash (\iota, \varepsilon) : \Theta_1 \rightarrow \Theta_2}$$

signals:

$$\vdash \text{load}\langle tmu \rangle : \Sigma^{tmu} :: at \rightarrow \Sigma^{tmu} \mid \Gamma \rightarrow \Gamma[r4 \mapsto at]$$

conditional instructions with signals:

$$\frac{\vdash ci : \Theta_1 \rightarrow \Theta_2}{\vdash (ci, \varepsilon) : \Theta_1 \rightarrow \Theta_2}$$

$$\frac{\vdash ci : \Gamma \rightarrow \Gamma[rr_1 \mapsto at_1] \mid u_1 \rightarrow u_2 \mid \Psi_1 \rightarrow \Psi_2 \mid C_1 \rightarrow C_2 \mid \Omega_1 \rightarrow \Omega_2 \mid wr_1 \rightarrow wr_2 \quad \vdash s : \Gamma \rightarrow \Gamma[r4 \mapsto at_2] \mid \Sigma_1^{tmu} \rightarrow \Sigma_2^{tmu}}{\vdash (ci, s) : \Gamma \rightarrow \Gamma[rr_1 \mapsto at_1][r4 \mapsto at_2] \mid u_1 \rightarrow u_2 \mid \Psi_1 \rightarrow \Psi_2 \mid \Sigma_1^{tmu} \rightarrow \Sigma_2^{tmu} \mid C_1 \rightarrow C_2 \mid \Omega_1 \rightarrow \Omega_2 \mid wr_1 \rightarrow wr_2}$$

labels:

$$\frac{l \in \text{dom}(\Phi)}{\Phi \vdash l : \Phi(l)}$$

instruction sequences:

$$\frac{\vdash csi : \Theta_1 \rightarrow \Theta_2 \quad \Phi \vdash I : \Theta_2}{\Phi \vdash csi ; I : \Theta_1}$$

$$\frac{\begin{array}{l} \vdash csi_1 : \Theta_1 \rightarrow \Theta_2 \\ \vdash csi_2 : \Theta_2 \rightarrow \Theta_3 \\ \vdash csi_3 : \Theta_3 \rightarrow \Theta_4 \\ \Phi \vdash l : \text{code}(\Theta_5) \\ \Theta_4 <: \Theta_5 \end{array}}{\Phi \vdash \text{jmp}(l) ; csi_1 ; csi_2 ; csi_3 : \Theta_1}$$

$$\begin{array}{c}
\vdash ci : \Theta_{11} \rightarrow \Theta_2 \quad \vdash csi_1 : \Theta_2 \rightarrow \Theta_3 \\
\vdash csi_2 : \Theta_3 \rightarrow \Theta_4 \\
regctx(\Theta_2) \equiv regctx(\Theta_{11})[rwr \mapsto at] \\
rwr \neq ran \\
rwr \neq rbn \quad \Theta_1 \equiv (\Gamma_1, \Psi_1, u_1, \Sigma_1^{tmu}, C_1, \Omega_1, wr_1) \\
\Theta_{11} \equiv (\Gamma_1 \setminus \{ra14, rb14\}, \Psi_1, u_1, \Sigma_1^{tmu}, C_1, \Omega_1, wr_1) \\
\{ra14, rb14\} \notin regctx(\Theta_2) \\
\{ra14, rb14\} \notin regctx(\Theta_3) \\
\{ra14, rb14\} \notin regctx(\Theta_4) \\
\hline
\vdash (ci, thread\_end) ; csi_1 ; csi_2 : \Theta_1
\end{array}$$

$$\begin{array}{c}
\vdash csi_1 : \Theta_1 \rightarrow \Theta_2 \quad \vdash csi_2 : \Theta_2 \rightarrow \Theta_3 \\
\vdash csi_3 : \Theta_3 \rightarrow \Theta_4 \\
\Phi \vdash l : code(\Theta_4) \quad \Phi \vdash I : \Theta_5 \\
\Theta_4 <: \Theta_5 \\
\hline
\Phi \vdash \text{if } qc \text{ jmp}(l) ; csi_1 ; csi_2 ; csi_3 ; I : \Theta_1
\end{array}$$

programs:

$$\frac{\forall l \in dom(P) . \Phi \vdash P(l) : \Theta_l}{\Phi \vdash P}$$

It is defined that when the elements of  $at$  all have the same basic type  $b$ , it is convertible with  $vec(b, 16)$ .

## 2.4. Auxiliary functions

Note that all free meta-variables are assumed to be fresh.

$$notfull(\Sigma^{tmu}) \stackrel{\text{def}}{=} (\Sigma^{tmu} \neq at_1 :: at_2 :: at_3 :: at_4)$$

$$unwrap_{\Xi}(\text{ptr}(p)) \stackrel{\text{def}}{=} \Xi(p) \text{ if } p \in dom(\Xi)$$

$$fst([b_0, b_1, \dots, b_{15}]) \stackrel{\text{def}}{=} b_0$$

$$map(f, [b_0, b_1, \dots, b_{15}]) \stackrel{\text{def}}{=} [f(b_0), f(b_1), \dots, f(b_{15})]$$

When an array type has the same 16 basic type, written  $array(b)$ :

$$array(b) \stackrel{\text{def}}{=} [b, b, \dots, b]$$

$$inc(y_{63}) \stackrel{\text{def}}{=} y_0$$

$$inc(y_n) \stackrel{\text{def}}{=} y_{n+1} \text{ if } 0 \leq n \leq 62$$

$$regctx((\Gamma, \Psi, u, \Sigma, C, \Omega, wr)) \stackrel{\text{def}}{=} \Gamma$$

$size\_of(\tau)$  represents the size of a value of  $\tau$  in bytes.

$$size\_of(b) \stackrel{\text{def}}{=} 4$$

$$size\_of(at) \stackrel{\text{def}}{=} 16 \times 4$$

$$size\_of(b \circ \Psi) \stackrel{\text{def}}{=} 4 + size\_of(\Psi)$$

$$size\_of(\varepsilon) \stackrel{\text{def}}{=} 0$$

$$size\_of(\text{vec}(b, n)) \stackrel{\text{def}}{=} size\_of(b) \times n$$

$$dma\_load(Y, 1, \tau) \stackrel{\text{def}}{=} [Y \mapsto \text{truncate}(\tau)]$$

$$dma\_load(Y, i4^*, at) \stackrel{\text{def}}{=} [Y \mapsto at] \text{ if } i4^* \geq 2$$

$$\text{truncate}(at) \stackrel{\text{def}}{=} at$$

$$\text{concat}(\text{vec}(b, m), \text{vec}(b, n)) \stackrel{\text{def}}{=} \text{vec}(b, m + n)$$

### 3. Future

- Any properties are not proved.
- There are many implicitness.
- The current definition is so conservative that it cannot serve practical use.
- The current definition may be incorrect or inconsistent.