asm.js

Dave Herman, Luke Wagner, and Alon Zakai

November 14, 2012

1 Abstract syntax

$$\begin{array}{rcl} b, e, f, g, x, y, z & \in & \mathit{Identifier} \\ \texttt{arguments}, \texttt{eval} & \notin & \mathit{Identifier} \end{array}$$

$$\begin{array}{rcl} P & ::= & \texttt{function } [g]\texttt{(}[e[,b]]\texttt{) \{ "use asm"; } \overline{\mathit{imp}_x}\ \overline{\mathit{fn}_f}\ \overline{\texttt{var } \overline{y \texttt{ = } v}\texttt{;}}\ \mathit{exp}\ \texttt{\}} \\ \mathit{imp}_x & ::= & \texttt{var } x \texttt{ = } e\texttt{.}y\texttt{;} \\ & | & \texttt{var } x \texttt{ = new } e\texttt{.}y\texttt{(}b\texttt{);} \\ \mathit{exp} & ::= & \texttt{return } f\texttt{;} \\ & | & \texttt{return \{ } \overline{x\texttt{:}f}\texttt{ \};} \\ \mathit{fn}_f & ::= & \texttt{function } f\texttt{(}\overline{x}\texttt{) \{ } \overline{x \texttt{ = } \kappa_x\texttt{;}}\ \overline{\texttt{var } \overline{y \texttt{ = } v}\texttt{;}}\ \mathit{ss}\ \texttt{\}} \end{array}$$

$$\begin{array}{rcl} s & ::= & \texttt{\{ } \mathit{ss} \texttt{ \}} \\ & | & e\texttt{;} \\ & | & \texttt{;} \\ & | & \texttt{if (}e\texttt{) } s \\ & | & \texttt{if (}e\texttt{) } s \texttt{ else } s \\ & | & \texttt{return } e\texttt{;} \\ & | & \texttt{while (}e\texttt{) } s \\ & | & \texttt{do } s \texttt{ while (}e\texttt{);} \\ & | & \texttt{for (}[e]\texttt{; }[e]\texttt{; }[e]\texttt{) } s \\ & | & \texttt{switch (}e\texttt{) \{ } \overline{c} \texttt{ \}} \\ & | & \texttt{switch (}e\texttt{) \{ } \overline{c}\ d \texttt{ \}} \\ & | & \texttt{break;} \\ & | & \texttt{break } \mathit{lab}\texttt{;} \\ & | & \texttt{continue;} \\ & | & \texttt{continue } \mathit{lab}\texttt{;} \\ & | & \mathit{lab}\texttt{: } s \end{array}$$

$$\mathit{ss} ::= \overline{s}$$

$$\begin{array}{rcl} c & ::= & \texttt{case } v\texttt{: } \mathit{ss} \\ d & ::= & \texttt{default: } \mathit{ss} \\ \mathit{cd} & ::= & c \mid d \end{array}$$

$$\kappa_x ::= \texttt{~~}x \mid \texttt{+}x \mid x\texttt{|0} \mid x\texttt{>>>0}$$

$$
\begin{array}{rcl}
v & ::= & r \mid n \\
\\
e & ::= & v \\
& \mid & \mathit{lval} \\
& \mid & \mathit{lval}\ \texttt{=}\ e \\
& \mid & f\texttt{(}\overline{e}\texttt{)} \\
& \mid & \mathit{unop}\ e \\
& \mid & e\ \mathit{binop}\ e \\
& \mid & e\ \texttt{?}\ e\ \texttt{:}\ e \\
& \mid & \texttt{(}\overline{e}\texttt{)} \\
\\
\mathit{unop} & ::= & \texttt{+} \mid \texttt{~} \mid \texttt{!} \\
\\
\mathit{binop} & ::= & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \mid \texttt{\%} \\
& \mid & \texttt{|} \mid \texttt{\&} \mid \texttt{\^{}} \mid \texttt{<<} \mid \texttt{>>} \mid \texttt{>>>} \\
& \mid & \texttt{<} \mid \texttt{<=} \mid \texttt{>} \mid \texttt{>=} \mid \texttt{!=} \mid \texttt{==} \\
\mathit{lval} & ::= & x \mid x\texttt{[(}e\ \texttt{\&}\ m\texttt{)}\ \texttt{>>}\ n\texttt{]}
\end{array}
$$

2 Type rules

$$
\begin{array}{rcl}
\sigma, \tau & ::= & \texttt{bit} \mid \texttt{double} \mid \texttt{int} \mid \texttt{signed} \mid \texttt{unsigned} \mid \texttt{boolish} \mid \texttt{intish} \mid \texttt{void} \mid \texttt{unknown} \\
\rho & ::= & \tau \mid \texttt{array}^n_\tau \mid \texttt{imul} \mid \texttt{function} \mid (\overline{\sigma}) \rightarrow \tau \\
\omega & ::= & ((\overline{\sigma}) \rightarrow \tau) \wedge \ldots \wedge ((\overline{\sigma'}) \rightarrow \tau')
\end{array}
$$

$$
\begin{array}{rcl}
\ell & ::= & \mathit{lab} \mid \epsilon \\
L & ::= & \{\overline{\ell}\} \\
\varepsilon & ::= & L \mid \textsf{return}
\end{array}
$$

$$
\begin{array}{rcl}
L \mathbin{;} L' & = & L \cup L' \\
\emptyset \mathbin{;} \textsf{return} & = & \textsf{return} \\
\{\ell, \overline{\ell'}\} \mathbin{;} \textsf{return} & = & \{\ell, \overline{\ell'}\} \\
\textsf{return} \mathbin{;} L & = & \textsf{return}
\end{array}
$$

$$
\begin{array}{rcl}
L \cup \textsf{return} & = & L \\
\textsf{return} \cup L & = & L \\
\textsf{return} \cup \textsf{return} & = & \textsf{return}
\end{array}
$$

$$
\begin{array}{rcl}
\mathit{type}(\texttt{\~{}\~{}}X) & = & \texttt{int} \\
\mathit{type}(\texttt{+}X) & = & \texttt{double} \\
\mathit{type}(n) & = & \texttt{int} \\
\mathit{type}(r) & = & \texttt{double} \\
\mathit{type}(X\,\texttt{|0}) & = & \texttt{signed} \\
\mathit{type}(X\texttt{>>>0}) & = & \texttt{unsigned}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{constant} & <: & \texttt{signed}, \texttt{unsigned} \\
\texttt{signed}, \texttt{unsigned} & <: & \texttt{int}, \texttt{extern} \\
\texttt{bit}, \texttt{int} & <: & \texttt{boolish} \\
\texttt{double} & <: & \texttt{extern} \\
\texttt{unknown}, \texttt{int} & <: & \texttt{intish}
\end{array}
$$

$$
\begin{array}{rcl}
M(\texttt{imul}) & : & \texttt{imul} \\
M(\texttt{ceil}), M(\texttt{sin}), M(\texttt{cos}) & : & (\texttt{double}) \to \texttt{double}
\end{array}
$$

$$
\begin{array}{rcl}
A(\texttt{Uint8Array}), A(\texttt{Int8Array}) & = & \texttt{array}^{8}_{\texttt{int}} \\
A(\texttt{Uint16Array}), A(\texttt{Int16Array}) & = & \texttt{array}^{16}_{\texttt{int}} \\
A(\texttt{Uint32Array}), A(\texttt{Int32Array}) & = & \texttt{array}^{32}_{\texttt{int}} \\
A(\texttt{Float32Array}) & = & \texttt{array}^{32}_{\texttt{double}} \\
A(\texttt{Float64Array}) & = & \texttt{array}^{64}_{\texttt{double}}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{+}, \texttt{-} & : & \quad (\texttt{double}, \texttt{double}) \to \texttt{double} \\
 & & \wedge\ (\texttt{int}, \texttt{int}) \to \texttt{intish} \\
\texttt{*} & : & \quad (\texttt{double}, \texttt{double}) \to \texttt{double} \\
\texttt{/}, \texttt{\%} & : & \quad (\texttt{double}, \texttt{double}) \to \texttt{double} \\
 & & \wedge\ (\texttt{signed}, \texttt{signed}) \to \texttt{intish} \\
 & & \wedge\ (\texttt{unsigned}, \texttt{unsigned}) \to \texttt{intish} \\
 & & \\
\texttt{|}, \texttt{\&}, \texttt{\^{}}, \texttt{<<}, \texttt{>>} & : & \quad (\texttt{intish}, \texttt{intish}) \to \texttt{signed} \\
\texttt{>>>} & : & \quad (\texttt{intish}, \texttt{intish}) \to \texttt{unsigned} \\
 & & \\
\texttt{<}, \texttt{<=}, \texttt{>}, \texttt{>=}, \texttt{==}, \texttt{!=} & : & \quad (\texttt{signed}, \texttt{signed}) \to \texttt{bit} \\
 & & \wedge\ (\texttt{unsigned}, \texttt{unsigned}) \to \texttt{bit} \\
 & & \wedge\ (\texttt{double}, \texttt{double}) \to \texttt{bit} \\
 & & \\
\texttt{+} & : & \quad (\texttt{intish}) \to \texttt{double} \\
\texttt{\~{}} & : & \quad (\texttt{intish}) \to \texttt{signed} \\
\texttt{!} & : & \quad (\texttt{boolish}) \to \texttt{bit}
\end{array}
$$

$$
\begin{array}{rcl}
\Delta & ::= & \{\overline{x : \rho}\} \\
\Gamma & ::= & \{\overline{x : \tau}\}
\end{array}
$$

Program checking $\boxed{\vdash P\ \textbf{ok}}$

[T-Program]

$$\dfrac{\begin{gathered}\overline{x}, \overline{y}, \overline{f}, [g], [e], [b] \text{ distinct} \qquad \forall y.\Delta(y) = \mathit{type}(v) \\ \forall i.[e];[b];\Delta \vdash \mathit{imp}_x\ \textbf{ok} \qquad \forall i.\Delta \vdash \mathit{fn}_f\ \textbf{ok} \qquad \forall i.\Delta \vdash \mathit{exp}\ \textbf{ok}\end{gathered}}{\vdash \texttt{function}\ [g]\texttt{(}[e[,b]]\texttt{)}\ \texttt{\{}\ \texttt{"use asm";}\ \overline{\mathit{imp}_x}\ \overline{\mathit{fn}_f}\ \overline{\texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}}\ \mathit{exp}\ \texttt{\}}\ \textbf{ok}}$$

Import checking $\boxed{[e];[b];\Delta \vdash \mathit{imp}\ \textbf{ok}}$

[T-ImportStd]

$$\dfrac{\Delta(x) = M(y)}{e;[b];\Delta \vdash \texttt{var}\ x\ \texttt{=}\ e\texttt{.}y\texttt{;}\ \textbf{ok}}$$

[T-ImportFFI]

$$\dfrac{y \notin \mathit{dom}(M), \mathit{dom}(A) \qquad \Delta(x) = \texttt{function}}{e;[b];\Delta \vdash \texttt{var}\ x\ \texttt{=}\ e\texttt{.}y\texttt{;}\ \textbf{ok}}$$

[T-View]

$$\dfrac{\Delta(x) = \texttt{array}^n_{A(y)}}{e;b;\Delta \vdash \texttt{var}\ x\ \texttt{=}\ \texttt{new}\ e\texttt{.}y(b)\texttt{;}\ \textbf{ok}}$$

Function checking $\boxed{\Delta \vdash \mathit{fn}\ \textbf{ok}}$

[T-Function]

$$\dfrac{\begin{gathered}\overline{x}, \overline{y} \text{ distinct} \qquad \Delta(f) = (\overline{\sigma}) \to \tau \qquad \overline{\sigma} = \overline{\mathit{type}(\kappa_x)} \\ \tau \neq \texttt{void} \Rightarrow \vdash \mathit{ss} \hookrightarrow \mathsf{return} \\ \Delta; \{\overline{x : \sigma}, \overline{y : \mathit{type}(v)}\}; f \vdash \mathit{ss}\ \textbf{ok}\end{gathered}}{\Delta \vdash \texttt{function}\ f\texttt{(}\overline{x}\texttt{)}\ \texttt{\{}\ \overline{x\ \texttt{=}\ \kappa_x\texttt{;}}\ \overline{\texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}}\ \mathit{ss}\ \texttt{\}}\ \textbf{ok}}$$

Export checking $\boxed{\Delta \vdash \mathit{exp}\ \textbf{ok}}$

[T-Singleton]

$$\dfrac{\Delta(f) = (\overline{\sigma}) \to \tau \qquad \tau <: \texttt{extern}}{\Delta \vdash \texttt{return}\ f\texttt{;}\ \textbf{ok}}$$

[T-Module]

$$\dfrac{\forall f.\Delta(f) = (\overline{\sigma}) \to \tau \wedge \tau <: \texttt{extern}}{\Delta \vdash \texttt{return}\ \texttt{\{}\ \overline{x\texttt{:}f}\ \texttt{\};}\ \textbf{ok}}$$

Statement list control flow analysis $\boxed{\vdash ss \hookrightarrow \varepsilon}$

$$\text{[A-NoStatements]}\quad \frac{}{\vdash \epsilon \hookrightarrow \emptyset} \qquad \text{[A-Statements]}\quad \frac{\forall i. \vdash s_i \hookrightarrow \varepsilon_i \qquad n > 0 \qquad \varepsilon = \varepsilon_1; \ldots; \varepsilon_n}{\vdash \overline{s} \hookrightarrow \varepsilon}$$

Statement control flow analysis $\boxed{\vdash s \hookrightarrow \varepsilon}$

$$\text{[A-Block]}\quad \frac{\vdash ss \hookrightarrow \varepsilon}{\vdash \texttt{\{}\ ss\ \texttt{\}} \hookrightarrow \varepsilon} \qquad \text{[A-ExprStmt]}\quad \frac{}{\vdash e\texttt{;} \hookrightarrow \emptyset} \qquad \text{[A-Return]}\quad \frac{}{\vdash \texttt{return}\ [e]\texttt{;} \hookrightarrow \mathsf{return}}$$

$$\text{[A-If]}\quad \frac{\vdash s \hookrightarrow \varepsilon \quad \varepsilon' = \varepsilon \cup \emptyset}{\vdash \texttt{if (}e\texttt{)}\ s \hookrightarrow \varepsilon'} \qquad \text{[A-IfElse]}\quad \frac{\vdash s_1 \hookrightarrow \varepsilon_1 \qquad \vdash s_2 \hookrightarrow \varepsilon_2 \quad \varepsilon = \varepsilon_1 \cup \varepsilon_2}{\vdash \texttt{if (}e\texttt{)}\ s_1\ \texttt{else}\ s_2 \hookrightarrow \varepsilon}$$

$$\text{[A-While]}\quad \frac{\vdash s \hookrightarrow \varepsilon \quad \varepsilon' = \emptyset \cup \varepsilon - \{\epsilon\}}{\vdash \texttt{while (}e\texttt{)}\ s \hookrightarrow \varepsilon'} \qquad \text{[A-DoWhile]}\quad \frac{\vdash s \hookrightarrow \varepsilon \varepsilon' = \varepsilon - \{\epsilon\}}{\vdash \texttt{do}\ s\ \texttt{while (}e\texttt{);} \hookrightarrow \varepsilon'}$$

$$\text{[A-For]}\quad \frac{\vdash s \hookrightarrow \varepsilon \quad \varepsilon' = \emptyset \cup \varepsilon - \{\epsilon\}}{\vdash \texttt{for (}[e_1]\texttt{; }[e_2]\texttt{; }[e_3]\texttt{)}\ s \hookrightarrow \varepsilon'}$$

$$\text{[A-Break]}\quad \frac{}{\vdash \texttt{break;} \hookrightarrow \{\epsilon\}} \qquad \text{[A-BreakLabel]}\quad \frac{}{\vdash \texttt{break}\ lab\texttt{;} \hookrightarrow \{lab\}} \qquad \text{[A-Continue]}\quad \frac{}{\vdash \texttt{continue}\ [lab]\texttt{;} \hookrightarrow \emptyset}$$

$$\text{[A-Label]}\quad \frac{\vdash s \hookrightarrow \varepsilon \quad \varepsilon' = \varepsilon - \{lab\}}{\vdash lab\texttt{:}\ s \hookrightarrow \varepsilon'} \qquad \text{[A-Switch]}\quad \frac{\forall i. \vdash cd_i \hookrightarrow \varepsilon_i \quad \varepsilon = \begin{cases} \mathsf{return} & \text{if } \varepsilon_n = \mathsf{return} \land \forall i. \varepsilon_i \cup \emptyset = \emptyset \\ \bigcup \varepsilon_i - \{\epsilon\} & \text{otherwise} \end{cases}}{\vdash \texttt{switch (}e\texttt{) \{}\ \overline{cd}\ \texttt{\}} \hookrightarrow \varepsilon}$$

$$\text{[A-EmptySwitch]}\quad \frac{}{\vdash \texttt{switch (}e\texttt{) \{ \}} \hookrightarrow \emptyset} \qquad \text{[A-EmptyStatement]}\quad \frac{}{\vdash \texttt{;} \hookrightarrow \emptyset}$$

Case control flow analysis $\boxed{\vdash cd \hookrightarrow \varepsilon}$

$$\text{[A-Case]}\quad \frac{\vdash ss \hookrightarrow \varepsilon}{\vdash \texttt{case}\ v\texttt{:}\ ss \hookrightarrow \varepsilon} \qquad \text{[A-Default]}\quad \frac{\vdash ss \hookrightarrow \varepsilon}{\vdash \texttt{default:}\ ss \hookrightarrow \varepsilon}$$

Statement list checking $\boxed{\Delta; \Gamma; f \vdash ss\ \mathbf{ok}}$

$$\frac{\forall i. \Delta; \Gamma; f \vdash s_i\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \overline{s}\ \mathbf{ok}}\ \text{[T-Statements]}$$

Statement checking $\boxed{\Delta; \Gamma; f \vdash s\ \mathbf{ok}}$

$$\frac{\Delta; \Gamma; f \vdash ss\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{\{}\ ss\ \texttt{\}}\ \mathbf{ok}}\ \text{[T-Block]} \qquad \frac{\Delta; \Gamma \vdash e : \sigma}{\Delta; \Gamma; f \vdash e\texttt{;}\ \mathbf{ok}}\ \text{[T-ExprStmt]} \qquad \frac{}{\Delta; \Gamma; f \vdash \texttt{;}\ \mathbf{ok}}\ \text{[T-EmptyStatement]}$$

$$\frac{\Delta; \Gamma \vdash e : \texttt{boolish} \quad \Delta; \Gamma; f \vdash s\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{if (}e\texttt{)}\ s\ \mathbf{ok}}\ \text{[T-If]} \qquad \frac{\Delta; \Gamma \vdash e : \texttt{boolish} \quad \Delta; \Gamma; f \vdash s_1\ \mathbf{ok} \quad \Delta; \Gamma; f \vdash s_2\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{if (}e\texttt{)}\ s_1\ \texttt{else}\ s_2\ \mathbf{ok}}\ \text{[T-IfElse]}$$

$$\frac{\Delta(f) = (\overline{\sigma}) \to \tau \quad \Delta; \Gamma \vdash x : \tau}{\Delta; \Gamma; f \vdash \texttt{return}\ x\texttt{;}\ \mathbf{ok}}\ \text{[T-ReturnVar]} \qquad \frac{\Delta(f) = (\overline{\sigma}) \to \tau \quad \Delta; \Gamma \vdash e : \tau \quad \mathit{type}(e) <: \tau}{\Delta; \Gamma; f \vdash \texttt{return}\ e\texttt{;}\ \mathbf{ok}}\ \text{[T-ReturnExpr]}$$

$$\frac{\Delta(f) = (\overline{\sigma}) \to \texttt{void}}{\Delta; \Gamma; f \vdash \texttt{return;}\ \mathbf{ok}}\ \text{[T-ReturnVoid]}$$

$$\frac{\Delta; \Gamma \vdash e : \texttt{boolish} \quad \Delta; \Gamma; f \vdash s\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{while (}e\texttt{)}\ s\ \mathbf{ok}}\ \text{[T-While]} \qquad \frac{\Delta; \Gamma; f \vdash s\ \mathbf{ok} \quad \Delta; \Gamma \vdash e : \texttt{boolish}}{\Delta; \Gamma; f \vdash \texttt{do}\ s\ \texttt{while (}e\texttt{);}\ \mathbf{ok}}\ \text{[T-DoWhile]}$$

$$\frac{[\Delta; \Gamma \vdash e_1 : \sigma_1] \quad [\Delta; \Gamma \vdash e_2 : \texttt{boolish}] \quad [\Delta; \Gamma \vdash e_3 : \sigma_3] \quad \Delta; \Gamma; f \vdash s\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{for (}[e_1]\texttt{;}\ [e_2]\texttt{;}\ [e_3]\texttt{)}\ s\ \mathbf{ok}}\ \text{[T-For]}$$

$$\frac{}{\Delta; \Gamma; f \vdash \texttt{break}\ [\mathit{lab}]\texttt{;}\ \mathbf{ok}}\ \text{[T-Break]} \qquad \frac{}{\Delta; \Gamma; f \vdash \texttt{continue}\ [\mathit{lab}]\texttt{;}\ \mathbf{ok}}\ \text{[T-Continue]}$$

$$\frac{\Delta; \Gamma; f \vdash s\ \mathbf{ok}}{\Delta; \Gamma; f \vdash \mathit{lab}\texttt{:}\ s\ \mathbf{ok}}\ \text{[T-Label]} \qquad \frac{\Delta; \Gamma \vdash e : \sigma \quad \sigma <: \texttt{extern} \quad \forall i. \mathit{type}(v_i) <: \sigma \quad \forall i. \Delta; \Gamma; f \vdash ss_i\ \mathbf{ok} \quad [\Delta; \Gamma; f \vdash ss\ \mathbf{ok}]}{\Delta; \Gamma; f \vdash \texttt{switch (}e\texttt{) \{}\ \overline{\texttt{case}\ v_i\texttt{:}\ ss_i}\ [\texttt{default:}\ ss]\ \texttt{\}}\ \mathbf{ok}}\ \text{[T-Switch]}$$

Case checking $\boxed{\Delta; \Gamma; f \vdash cd \ \mathbf{ok}}$

[T-CASE]
$$\frac{\Delta; \Gamma; f \vdash ss \ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{case} \ \ v\texttt{:} \ ss \ \mathbf{ok}}$$

[T-DEFAULT]
$$\frac{\Delta; \Gamma; f \vdash ss \ \mathbf{ok}}{\Delta; \Gamma; f \vdash \texttt{default:} \ ss \ \mathbf{ok}}$$

$$(\Delta \cdot \Gamma)(x) = \begin{cases} \Gamma(x) & \text{if } x \in \mathit{dom}(\Gamma) \\ \Delta(x) & \text{otherwise} \end{cases}$$

Expression checking $\boxed{\Delta; \Gamma \vdash e : \tau}$

[T-CONSTANT]
$$\frac{-2^{31} \leq n < 2^{32}}{\Delta; \Gamma \vdash n : \texttt{constant}}$$

[T-DOUBLE]
$$\frac{}{\Delta; \Gamma \vdash r : \texttt{double}}$$

[T-VARREF]
$$\frac{(\Delta \cdot \Gamma)(x) = \tau}{\Delta; \Gamma \vdash x : \tau}$$

[T-ASSIGN]
$$\frac{\Delta; \Gamma \vdash e : \tau \qquad \tau <: (\Delta \cdot \Gamma)(x)}{\Delta; \Gamma \vdash x \ \texttt{=} \ e : \tau}$$

[T-LOAD]
$$\frac{m = 2^k - 1 \qquad (\Delta \cdot \Gamma)(x) = \texttt{array}^n_\tau \qquad \Delta; \Gamma \vdash e : \texttt{intish}}{\Delta; \Gamma \vdash x\texttt{[(}e \ \texttt{\&} \ m\texttt{)} \ \texttt{>>} \ n/8\texttt{]} : \tau}$$

[T-STORE]
$$\frac{m = 2^k - 1 \qquad (\Delta \cdot \Gamma)(x) = \texttt{array}^n_\tau \qquad \Delta; \Gamma \vdash e_1 : \texttt{intish} \qquad \Delta; \Gamma \vdash e_2 : \tau}{\Delta; \Gamma \vdash x\texttt{[(}e_1 \ \texttt{\&} \ m\texttt{)} \ \texttt{>>} \ n/8\texttt{]} \ \texttt{=} \ e_2 : \tau}$$

[T-IMUL]
$$\frac{(\Delta \cdot \Gamma)(f) = \texttt{imul} \qquad \forall i.\Delta; \Gamma \vdash e_i : \texttt{intish}}{\Delta; \Gamma \vdash f\texttt{(}e_1, e_2\texttt{)} : \texttt{signed}}$$

[T-FUNCALL]
$$\frac{(\Delta \cdot \Gamma)(f) = (\overline{\sigma}) \to \tau \qquad \forall i.\Delta; \Gamma \vdash e_i : \sigma_i}{\Delta; \Gamma \vdash f\texttt{(}\overline{e}\texttt{)} : \tau}$$

[T-FFI]
$$\frac{(\Delta \cdot \Gamma)(f) = \texttt{function} \qquad \forall i.\Delta; \Gamma \vdash e_i : \texttt{extern}}{\Delta; \Gamma \vdash f\texttt{(}\overline{e}\texttt{)} : \texttt{unknown}}$$

[T-CONDITIONAL]
$$\frac{\Delta; \Gamma \vdash e_1 : \texttt{boolish} \qquad \Delta; \Gamma \vdash e_2 : \tau \qquad \Delta; \Gamma \vdash e_3 : \tau}{\Delta; \Gamma \vdash e_1 \ \texttt{?} \ e_2 \ \texttt{:} \ e_3 : \tau}$$

[T-PAREN]
$$\frac{\forall i \leq n.\Delta; \Gamma \vdash e_i : \tau_i}{\Delta; \Gamma \vdash \texttt{(}\overline{e}\texttt{)} : \tau_n}$$

[T-UNOP]
$$\frac{\mathit{unop} : \_ \wedge (\sigma) \to \tau \wedge \_ \qquad \Delta; \Gamma \vdash e : \sigma}{\Delta; \Gamma \vdash \mathit{unop} \ e : \tau}$$

[T-BINOP]
$$\frac{\mathit{binop} : \_ \wedge (\sigma_1, \sigma_2) \to \tau \wedge \_ \qquad \Delta; \Gamma \vdash e_1 : \sigma_1 \qquad \Delta; \Gamma \vdash e_2 : \sigma_2}{\Delta; \Gamma \vdash e_1 \ \mathit{binop} \ e_2 : \tau}$$

[T-SUB]
$$\frac{\Delta; \Gamma \vdash e : \sigma \qquad \sigma <: \tau}{\Delta; \Gamma \vdash e : \tau}$$

[T-CAST]
$$\frac{\Delta; \Gamma \vdash e : \texttt{double}}{\Delta; \Gamma \vdash \texttt{\~{}\~{}}e : \texttt{signed}}$$