asm.js

Dave Herman, Luke Wagner, and Alon Zakai

November 12, 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

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

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

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

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

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

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

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

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

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

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

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

[T-Program]

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

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

[T-ImportStd]

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

[T-ImportFFI]

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

[T-NewView]

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

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

[T-Function]

$$\frac{\overline{x}, \overline{y} \text{ distinct} \qquad \Delta(f) = (\overline{\sigma}) \to \tau \qquad \overline{\sigma} = \overline{\mathit{type}(\kappa_x)} \qquad \tau \neq \texttt{void} \\ \Delta; \{\overline{x : \sigma}, \overline{y : \mathit{type}(v)}\}; \emptyset \vdash \mathit{ss} : \tau/\mathsf{return}}{\Delta \vdash \texttt{function}\ f\texttt{(}\overline{x}\texttt{)}\ \texttt{\{}\ \overline{x\ \texttt{=}\ \kappa_x\texttt{;}}\ \overline{\texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}}\ \mathit{ss}\ \texttt{\}}\ \mathbf{ok}}$$

[T-VoidFunction]

$$\frac{\overline{x}, \overline{y} \text{ distinct} \qquad \Delta(f) = (\overline{\sigma}) \to \texttt{void} \qquad \overline{\sigma} = \overline{\mathit{type}(\kappa_x)} \\ \Delta; \{\overline{x : \sigma}, \overline{y : \mathit{type}(v)}\}; \emptyset \vdash \mathit{ss} : \texttt{void}/\varepsilon}{\Delta \vdash \texttt{function}\ f\texttt{(}\overline{x}\texttt{)}\ \texttt{\{}\ \overline{x\ \texttt{=}\ \kappa_x\texttt{;}}\ \overline{\texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}}\ \mathit{ss}\ \texttt{\}}\ \mathbf{ok}}$$

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

[T-Singleton]

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

[T-Module]

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

Statement list checking $\boxed{\Delta;\Gamma;L \vdash ss : \tau/\varepsilon}$

$$\text{[T-NoStatements]}\quad \frac{}{\Delta;\Gamma;L \vdash \epsilon : \tau/\emptyset}$$

$$\text{[T-Statements]}\quad \frac{\forall i.\Delta;\Gamma;L \vdash s_i : \tau/\varepsilon_i \qquad n > 0 \qquad \varepsilon = \varepsilon_1\ ;\ \ldots\ ;\ \varepsilon_n}{\Delta;\Gamma;L \vdash \overline{s} : \tau/\varepsilon}$$

Statement checking $\boxed{\Delta;\Gamma;L \vdash s : \tau/\varepsilon}$

$$\text{[T-Block]}\quad \frac{\Delta;\Gamma;\emptyset \vdash ss : \tau/\varepsilon}{\Delta;\Gamma;L \vdash \texttt{\{}\ ss\ \texttt{\}} : \tau/\varepsilon}$$

$$\text{[T-ExprStmt]}\quad \frac{\Delta;\Gamma \vdash e : \sigma}{\Delta;\Gamma;L \vdash e\texttt{;} : \tau/\emptyset}$$

$$\text{[T-If]}\quad \frac{\Delta;\Gamma \vdash e : \texttt{boolish} \qquad \Delta;\Gamma;\emptyset \vdash s : \tau/\varepsilon \qquad \varepsilon' = \varepsilon \cup \emptyset}{\Delta;\Gamma;L \vdash \texttt{if (}e\texttt{)}\ s : \tau/\varepsilon'}$$

$$\text{[T-IfElse]}\quad \frac{\Delta;\Gamma \vdash e : \texttt{boolish} \qquad \Delta;\Gamma;\emptyset \vdash s_1 : \tau/\varepsilon_1 \qquad \Delta;\Gamma;\emptyset \vdash s_2 : \tau/\varepsilon_2 \qquad \varepsilon = \varepsilon_1 \cup \varepsilon_2}{\Delta;\Gamma;L \vdash \texttt{if (}e\texttt{)}\ s_1\ \texttt{else}\ s_2 : \tau/\varepsilon}$$

$$\text{[T-ReturnExpr]}\quad \frac{\mathit{type}(e) <: \tau \qquad \Delta;\Gamma \vdash e : \tau}{\Delta;\Gamma;L \vdash \texttt{return}\ e\texttt{;} : \tau/\mathsf{return}}$$

$$\text{[T-ReturnVoid]}\quad \frac{}{\Delta;\Gamma;L \vdash \texttt{return;} : \mathsf{void}/\mathsf{return}}$$

$$\text{[T-While]}\quad \frac{\Delta;\Gamma \vdash e : \texttt{boolish} \qquad \Delta;\Gamma;L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \qquad \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\})}{\Delta;\Gamma;L \vdash \texttt{while (}e\texttt{)}\ s : \tau/\varepsilon'}$$

$$\text{[T-DoWhile]}\quad \frac{\Delta;\Gamma;L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \qquad \Delta;\Gamma \vdash e : \texttt{boolish} \qquad \varepsilon' = \varepsilon - (L \cup \{\epsilon\})}{\Delta;\Gamma;L \vdash \texttt{do}\ s\ \texttt{while (}e\texttt{);} : \tau/\varepsilon'}$$

$$\text{[T-For]}\quad \frac{[\Delta;\Gamma \vdash e_1 : \sigma_1] \qquad [\Delta;\Gamma \vdash e_2 : \texttt{boolish}] \qquad [\Delta;\Gamma \vdash e_3 : \sigma_3] \qquad \Delta;\Gamma;L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \qquad \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\})}{\Delta;\Gamma;L \vdash \texttt{for (}[e_1]\texttt{; }[e_2]\texttt{; }[e_3]\texttt{)}\ s : \tau/\varepsilon'}$$

Statement checking (cont'd) $\boxed{\Delta;\Gamma;L \vdash s : \tau/\varepsilon}$

[T-Break]
$$\frac{}{\Delta;\Gamma;L \vdash \texttt{break;} : \tau/\{\epsilon\}}$$

[T-BreakLabel]
$$\frac{}{\Delta;\Gamma;L \vdash \texttt{break}\ \ lab\texttt{;} : \tau/\{lab\}}$$

[T-Continue]
$$\frac{}{\Delta;\Gamma;L \vdash \texttt{continue;} : \tau/\emptyset}$$

[T-ContinueLabel]
$$\frac{}{\Delta;\Gamma;L \vdash \texttt{continue}\ \ lab\texttt{;} : \tau/\emptyset}$$

[T-Label]
$$\frac{\begin{array}{c}\Delta;\Gamma;L \cup \{lab\} \vdash s : \tau/\varepsilon \\ \varepsilon' = \varepsilon - (L \cup \{lab\})\end{array}}{\Delta;\Gamma;L \vdash lab\texttt{:}\ s : \tau/\varepsilon'}$$

[T-Switch]
$$\frac{\begin{array}{c}\Delta;\Gamma \vdash e : \sigma \qquad \sigma <: \texttt{extern} \\ \forall i.cd_i = \texttt{case}\ \ v_i\texttt{:}\ ss_i \Rightarrow \mathit{type}(v_i) <: \sigma \\ \forall i.\Delta;\Gamma;L \cup \{\epsilon\} \vdash cd_i : \tau/\varepsilon_i \\ \varepsilon = \begin{cases} \mathsf{return} & \text{if } \varepsilon_n = \mathsf{return} \wedge \forall i.\varepsilon_i \cup \emptyset = \emptyset \\ \bigcup \varepsilon_i - (L \cup \{\epsilon\}) & \text{otherwise} \end{cases}\end{array}}{\Delta;\Gamma;L \vdash \texttt{switch (}e\texttt{) \{ }\overline{cd}\texttt{ \}} : \tau/\varepsilon}$$

[T-EmptySwitch]
$$\frac{\Delta;\Gamma \vdash e : \sigma}{\Delta;\Gamma;L \vdash \texttt{switch (}e\texttt{) \{ \}} : \tau/\emptyset}$$

[T-EmptyStatement]
$$\frac{}{\Delta;\Gamma;L \vdash \texttt{;} : \tau/\emptyset}$$

Case checking $\boxed{\Delta;\Gamma;L \vdash cd : \sigma, \tau/\varepsilon}$

[T-Case]
$$\frac{\begin{array}{c}\Delta;\Gamma \vdash e : \sigma \\ \Delta;\Gamma;L \vdash ss : \tau/\varepsilon\end{array}}{\Delta;\Gamma;L \vdash \texttt{case}\ \ e\texttt{:}\ ss : \sigma, \tau/\varepsilon}$$

[T-Default]
$$\frac{\Delta;\Gamma;L \vdash ss : \tau/\varepsilon}{\Delta;\Gamma;L \vdash \texttt{default:}\ ss : \sigma, \tau/\varepsilon}$$

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

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

$$\text{[T-Constant]}\ \frac{-2^{31} \leq n < 2^{32}}{\Delta; \Gamma \vdash n : \texttt{constant}} \qquad \text{[T-Double]}\ \frac{}{\Delta; \Gamma \vdash r : \texttt{double}}$$

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

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

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

$$\text{[T-IMul]}\ \frac{(\Delta \cdot \Gamma)(f) = \texttt{imul} \qquad \forall i. \Delta; \Gamma \vdash e_i : \texttt{intish}}{\Delta; \Gamma \vdash f\texttt{(}e_1, e_2\texttt{)} : \texttt{signed}} \qquad \text{[T-FunCall]}\ \frac{(\Delta \cdot \Gamma)(f) = (\overline{\sigma}) \to \tau \qquad \forall i. \Delta; \Gamma \vdash e_i : \sigma_i}{\Delta; \Gamma \vdash f\texttt{(}\overline{e}\texttt{)} : \tau}$$

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

$$\text{[T-Conditional]}\ \frac{\Delta; \Gamma \vdash e_1 : \texttt{boolish} \qquad \Delta; \Gamma \vdash e_2 : \tau \qquad \Delta; \Gamma \vdash e_3 : \tau}{\Delta; \Gamma \vdash e_1 \texttt{ ? } e_2 \texttt{ : } e_3 : \tau} \qquad \text{[T-Paren]}\ \frac{\forall i \leq n. \Delta; \Gamma \vdash e_i : \tau_i}{\Delta; \Gamma \vdash \texttt{(}\overline{e}\texttt{)} : \tau_n}$$

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

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

$$\text{[T-Sub]}\ \frac{\Delta; \Gamma \vdash e : \sigma \qquad \sigma <: \tau}{\Delta; \Gamma \vdash e : \tau} \qquad \text{[T-Cast]}\ \frac{\Delta; \Gamma \vdash e : \texttt{double}}{\Delta; \Gamma \vdash \texttt{\~{}\~{}}e : \texttt{signed}}$$