`asm.js`

Dave Herman, Luke Wagner, and Alon Zakai

November 4, 2012

1 Abstract syntax

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

$$
\begin{aligned}
P &::= \texttt{function}(b, e) \ \{ \ \overline{imp_x} \ \overline{fn_f} \ exp \ \} \\
imp_x &::= \texttt{var} \ x \ \texttt{=} \ e.y; \\
&\mid \texttt{var} \ x \ \texttt{=} \ \texttt{new} \ e.y(b); \\
exp &::= \texttt{return} \ f; \\
&\mid \texttt{return} \ \{ \ \overline{x\!:\!f} \ \}; \\
fn_f &::= \texttt{function} \ f(\overline{x}) \ \{ \ \overline{x \ \texttt{=} \ \kappa_x;} \ \texttt{var} \ \overline{y \ \texttt{=} \ v}; \ ss \ \}
\end{aligned}
$$

$$
\begin{aligned}
s \ ::= \ & \{ \ ss \ \} \\
\mid \ & e; \\
\mid \ & \texttt{if} \ (e) \ s \\
\mid \ & \texttt{if} \ (e) \ s \ \texttt{else} \ s \\
\mid \ & \texttt{return} \ v; \\
\mid \ & \texttt{while} \ (e) \ s \\
\mid \ & \texttt{do} \ s \ \texttt{while} \ (e); \\
\mid \ & \texttt{for} \ (e; \ e; \ e) \ s \\
\mid \ & \texttt{switch} \ (e) \ \{ \ \overline{c} \ \} \\
\mid \ & \texttt{switch} \ (e) \ \{ \ \overline{c} \ d \ \} \\
\mid \ & \texttt{break}; \\
\mid \ & \texttt{break} \ lab; \\
\mid \ & \texttt{continue}; \\
\mid \ & \texttt{continue} \ lab; \\
\mid \ & lab\!:\!s
\end{aligned}
$$

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

$$
\begin{aligned}
c &::= \texttt{case} \ e\!:\!ss \\
d &::= \texttt{default}\!:\!ss \\
cd &::= c \mid d
\end{aligned}
$$

$$\kappa_X ::= \text{\texttildelow\texttildelow} X \mid \text{+}X$$

$$
\begin{aligned}
v \quad &::= \quad r \mid n \\[4pt]
e \quad &::= \quad v \\
&\quad\mid \quad lval \\
&\quad\mid \quad lval \text{ = } e \\
&\quad\mid \quad f(\overline{e}) \\
&\quad\mid \quad unop\ e \\
&\quad\mid \quad e\ binop\ e \\
&\quad\mid \quad e\ \text{?}\ e\ \text{:}\ e \\
&\quad\mid \quad (\overline{e}) \\[4pt]
unop \quad &::= \quad \text{+} \mid \text{\texttildelow} \mid \text{!} \\[4pt]
binop \quad &::= \quad \text{+} \mid \text{-} \mid \text{*} \mid \text{/} \mid \text{\%} \\
&\quad\mid \quad \text{|} \mid \text{\&} \mid \text{\textasciicircum} \mid \text{<<} \mid \text{>>} \mid \text{>>>} \\
&\quad\mid \quad \text{<} \mid \text{<=} \mid \text{>} \mid \text{>=} \mid \text{!=} \mid \text{==} \\
lval \quad &::= \quad x \mid x\,\text{[}(e\ \text{\&}\ m)\ \text{>>}\ n\text{]}
\end{aligned}
$$

2 Type rules

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

$$
\begin{aligned}
\ell \quad &::= \quad lab \mid \epsilon \\
L \quad &::= \quad \{\overline{\ell}\} \\
\varepsilon \quad &::= \quad L \mid \mathsf{return}
\end{aligned}
$$

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

$$
\begin{aligned}
L \cup \mathsf{return} \quad &= \quad L \\
\mathsf{return} \cup L \quad &= \quad L \\
\mathsf{return} \cup \mathsf{return} \quad &= \quad \mathsf{return}
\end{aligned}
$$

$$
\begin{aligned}
type(\verb|~~|X) &= \verb|int| \\
type(\verb|+|X) &= \verb|double| \\
type(n) &= \verb|int| \\
type(r) &= \verb|double|
\end{aligned}
$$

$$
\begin{aligned}
\verb|signed|, \verb|unsigned| &<: \verb|int|, \verb|jsval| \\
\verb|bit|, \verb|int| &<: \verb|boolish| \\
\verb|void|, \verb|double|, \verb|array|_\tau^n, \verb|function|, \verb|unknown| &<: \verb|jsval| \\
\verb|unknown|, \verb|int| &<: \verb|coercible| \\
((\overline{\sigma}) \to \tau) \wedge \ldots \wedge ((\overline{\sigma'}) \to \tau') &<: \verb|function| \\
((\overline{\sigma}_1) \to \tau_1) \wedge \ldots \wedge ((\overline{\sigma}_n) \to \tau_n) &<: ((\overline{\sigma}_1) \to \tau_1) \wedge \ldots \wedge ((\overline{\sigma}_{n-1}) \to \tau_{n-1}) \\
\verb|imul| &<: (\verb|coercible|, \verb|coercible|) \to \verb|signed|
\end{aligned}
$$

$$
\Gamma ::= \{\overline{x : \tau}\} \mid \Gamma, \{\overline{x : \tau}\}
$$

$$
\begin{aligned}
M(\verb|imul|) &: \verb|imul| \\
M(\verb|ceil|), M(\verb|sin|), M(\verb|cos|) &: (\verb|double|) \to \verb|double|
\end{aligned}
$$

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

$$
\begin{aligned}
\verb|+|, \verb|-| \quad &: \quad (\verb|double|, \verb|double|) \to \verb|double| \\
&\wedge (\verb|int|, \verb|int|) \to \verb|coercible| \\
\verb|*| \quad &: \quad (\verb|double|, \verb|double|) \to \verb|double| \\
\verb|/|, \verb|%| \quad &: \quad (\verb|double|, \verb|double|) \to \verb|double| \\
&\wedge (\verb|signed|, \verb|signed|) \to \verb|coercible| \\
&\wedge (\verb|unsigned|, \verb|unsigned|) \to \verb|coercible|
\end{aligned}
$$

$$
\begin{aligned}
\verb:|:, \verb|&|, \verb|^|, \verb|<<|, \verb|>>| \quad &: \quad (\verb|coercible|, \verb|coercible|) \to \verb|signed| \\
\verb|>>>| \quad &: \quad (\verb|coercible|, \verb|coercible|) \to \verb|unsigned|
\end{aligned}
$$

$$
\begin{aligned}
\verb|<|, \verb|<=|, \verb|>|, \verb|>=|, \verb|==|, \verb|!=| \quad &: \quad (\verb|signed|, \verb|signed|) \to \verb|bit| \\
&\wedge (\verb|unsigned|, \verb|unsigned|) \to \verb|bit| \\
&\wedge (\verb|double|, \verb|double|) \to \verb|bit|
\end{aligned}
$$

$$
\begin{aligned}
\verb|+| \quad &: \quad (\verb|coercible|) \to \verb|double| \\
\verb|~| \quad &: \quad (\verb|coercible|) \to \verb|signed| \\
\verb|!| \quad &: \quad (\verb|boolish|) \to \verb|bit|
\end{aligned}
$$

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

[T-Program]
$$\frac{\begin{array}{ccc} \{\overline{x}\} \cap \{\overline{f}\} = \emptyset & \{\overline{x}\} \cap \{b, e\} = \emptyset & \{\overline{f}\} \cap \{b, e\} = \emptyset \\ \forall i.b; e; \Gamma_0 \vdash imp_x \text{ ok} & \forall i.\Gamma_0, \Gamma_1 \vdash fn_f \text{ ok} & \forall i.\Gamma_0, \Gamma_1 \vdash exp \text{ ok} \end{array}}{\vdash \texttt{function}(b, e) \ \{ \ \overline{imp_x} \ \overline{fn_f} \ exp \ \} \text{ ok}}$$

Import checking $\boxed{b; e; \Gamma \vdash imp \text{ ok}}$

[T-ImportStd]
$$\frac{\Gamma(x) = M(y)}{b; e; \Gamma \vdash \texttt{var} \ x \ \texttt{=} \ e.y\texttt{;} \ \text{ok}}$$

[T-ImportFFI]
$$\frac{y \notin dom(M) \qquad \Gamma(x) = \texttt{function}}{b; e; \Gamma \vdash \texttt{var} \ x \ \texttt{=} \ e.y\texttt{;} \ \text{ok}}$$

[T-View]
$$\frac{\Gamma(x) = \texttt{array}^n_{A(y)}}{b; e; \Gamma \vdash \texttt{var} \ x \ \texttt{=} \ e.y(b)\texttt{;} \ \text{ok}}$$

[T-NewView]
$$\frac{\Gamma(x) = \texttt{array}^n_{A(y)}}{b; e; \Gamma \vdash \texttt{var} \ x \ \texttt{=} \ \texttt{new} \ e.y(b)\texttt{;} \ \text{ok}}$$

Function checking $\boxed{\Gamma \vdash fn \text{ ok}}$

[T-Function]
$$\frac{\begin{array}{ccc} \{\overline{x}\} \cap \{\overline{y}\} = \emptyset & \Gamma(f) = (\overline{\sigma}) \to \tau & \overline{\sigma} = \overline{type(\kappa_x)} \qquad \tau \neq \texttt{void} \\ \multicolumn{3}{c}{\Gamma, \{\overline{x : \sigma}, \overline{y : type(v)}\}; \emptyset \vdash ss : \tau/\texttt{return}} \end{array}}{\Gamma \vdash \texttt{function} \ f(\overline{x}) \ \{ \ \overline{x \ \texttt{=} \ \kappa_x\texttt{;}} \ \texttt{var} \ \overline{y \ \texttt{=} \ v}\texttt{;} \ ss \ \} \text{ ok}}$$

[T-VoidFunction]
$$\frac{\begin{array}{ccc} \{\overline{x}\} \cap \{\overline{y}\} = \emptyset & \Gamma(f) = (\overline{\sigma}) \to \texttt{void} & \overline{\sigma} = \overline{type(\kappa_x)} \\ \multicolumn{3}{c}{\Gamma, \{\overline{x : \sigma}, \overline{y : type(v)}\}; \emptyset \vdash ss : \texttt{void}/\varepsilon} \end{array}}{\Gamma \vdash \texttt{function} \ f(\overline{x}) \ \{ \ \overline{x \ \texttt{=} \ \kappa_x\texttt{;}} \ \texttt{var} \ \overline{y \ \texttt{=} \ v}\texttt{;} \ ss \ \} \text{ ok}}$$

Export checking $\boxed{\Gamma \vdash exp \text{ ok}}$

[T-Singleton]
$$\frac{\Gamma(f) = (\overline{\sigma}) \to \tau \qquad \tau <: \texttt{jsval}}{\Gamma \vdash \texttt{return} \ f\texttt{;} \ \text{ok}}$$

[T-Module]
$$\frac{\forall f.\Gamma(f) <: (\overline{\sigma}) \to \tau \wedge \tau <: \texttt{jsval}}{\Gamma \vdash \texttt{return} \ \{ \ \overline{x\texttt{:}f} \ \}\texttt{;} \ \text{ok}}$$

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

[T-Statements]

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

$$\frac{\forall i. \Gamma; L \vdash s_i : \tau/\varepsilon_i \qquad n > 0 \qquad \varepsilon = \varepsilon_1 ; \ldots ; \varepsilon_n}{\Gamma; L \vdash \overline{s} : \tau/\varepsilon}$$

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

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

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

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

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

[T-ReturnExpr]
$$\frac{\Gamma \vdash e : \tau}{\Gamma; L \vdash \texttt{return } e; : \tau/\texttt{return}}$$

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

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

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

[T-For]
$$\frac{\forall i. \Gamma \vdash e_i : \sigma_i \qquad \Gamma; L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \qquad \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\})}{\Gamma; L \vdash \texttt{for } (e_1;\ e_2;\ e_3)\ s : \tau/\varepsilon'}$$

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

[T-Break]

$$\overline{\Gamma; L \vdash \texttt{break}; \; : \tau/\{\epsilon\}}$$

[T-BreakLabel]

$$\overline{\Gamma; L \vdash \texttt{break} \; lab; \; : \tau/\{lab\}}$$

[T-Continue]

$$\overline{\Gamma; L \vdash \texttt{continue}; \; : \tau/\emptyset}$$

[T-ContinueLabel]

$$\overline{\Gamma; L \vdash \texttt{continue} \; lab; \; : \tau/\emptyset}$$

[T-Label]

$$\frac{\Gamma; L \cup \{lab\} \vdash s : \tau/\varepsilon \quad \varepsilon' = \varepsilon - (L \cup \{lab\})}{\Gamma; L \vdash lab \colon s : \tau/\varepsilon'}$$

[T-Switch]

$$\frac{\Gamma \vdash e : \sigma \quad \forall i.\Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \quad \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\varepsilon \quad \varepsilon \neq \texttt{return} \vee \exists i.\varepsilon_i \cup \emptyset \neq \emptyset \quad \varepsilon' = (\varepsilon \cup \bigcup_i \varepsilon_i) - (L \cup \{\epsilon\})}{\Gamma; L \vdash \texttt{switch (} e \texttt{) \{ } \overline{c} \; cd \texttt{ \} } : \tau/\varepsilon'}$$

[T-SwitchReturn]

$$\frac{\Gamma \vdash e : \sigma \quad \forall i.\Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \quad \forall i.\varepsilon_i \cup \emptyset = \emptyset \quad \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\texttt{return}}{\Gamma; L \vdash \texttt{switch (} e \texttt{) \{ } \overline{c} \; cd \texttt{ \} } : \tau/\texttt{return}}$$

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

[T-Case]

$$\frac{\Gamma \vdash e : \sigma \quad \Gamma; L \vdash ss : \tau/\varepsilon}{\Gamma; L \vdash \texttt{case} \; e \colon ss : \sigma, \tau/\varepsilon}$$

[T-Default]

$$\frac{\Gamma; L \vdash ss : \tau/\varepsilon}{\Gamma; L \vdash \texttt{default} \colon ss : \sigma, \tau/\varepsilon}$$

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

[T-Int]
$$\frac{}{\Gamma \vdash n : \texttt{int}}$$

[T-Double]
$$\frac{}{\Gamma \vdash r : \texttt{double}}$$

[T-VarRef]
$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}$$

[T-Assign]
$$\frac{= \Gamma \vdash e : \Gamma(x)}{\Gamma \vdash x \ \texttt{=} \ e : \tau}$$

[T-Load]
$$\frac{\begin{array}{c} m = 2^k - 1 \\ \Gamma(x) = \texttt{array}^n_\tau \qquad \Gamma \vdash e : \texttt{int} \end{array}}{\Gamma \vdash x\texttt{[(}e \ \texttt{\&} \ m\texttt{)} \ \texttt{>>} \ n/8\texttt{]} : \tau}$$

[T-Store]
$$\frac{\begin{array}{c} m = 2^k - 1 \qquad \Gamma(x) = \texttt{array}^n_\tau \\ \Gamma \vdash e_1 : \texttt{int} \qquad \Gamma \vdash e_2 : \tau \end{array}}{\Gamma \vdash x\texttt{[(}e_1 \ \texttt{\&} \ m\texttt{)} \ \texttt{>>} \ n/8\texttt{]} \ \texttt{=} \ e_2 : \tau}$$

[T-FunCall]
$$\frac{\begin{array}{c} \Gamma(f) = (\overline{\sigma}) \to \tau \\ \forall i.\Gamma \vdash e_i : \sigma_i \end{array}}{\Gamma \vdash f(\overline{e}) : \tau}$$

[T-FFI]
$$\frac{\begin{array}{c} \Gamma(f) = \texttt{function} \\ \forall i.\Gamma \vdash e_i : \texttt{jsval} \end{array}}{\Gamma \vdash f(\overline{e}) : \texttt{unknown}}$$

[T-Conditional]
$$\frac{\begin{array}{c} \Gamma \vdash e_1 : \texttt{boolish} \\ \Gamma \vdash e_2 : \tau \qquad \Gamma \vdash e_3 : \tau \end{array}}{\Gamma \vdash e_1 \ \texttt{?} \ e_2 \ \texttt{:} \ e_3 : \tau}$$

[T-Paren]
$$\frac{\forall i \le n.\Gamma \vdash e_i : \tau_i}{\Gamma \vdash (\overline{e}) : \tau_n}$$

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

[T-Binop]
$$\frac{\begin{array}{c} binop : {}_{\_} \wedge (\sigma_1, \sigma_2) \to \tau \wedge {}_{\_} \\ \Gamma \vdash e_1 : \sigma_1 \qquad \Gamma \vdash e_2 : \sigma_2 \end{array}}{\Gamma \vdash e_1 \ binop \ e_2 : \tau}$$

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