asm.js

Dave Herman

October 23, 2012

1 Abstract syntax

$$
\begin{array}{rcl}
\rho & ::= & \textit{runtime lexical environment} \\
\Delta & ::= & \{\overline{f : (\overline{\sigma}) \to \tau}\} \\
\Gamma & ::= & \{\overline{x : \tau}\}
\end{array}
$$

$$
\begin{array}{rcl}
P & ::= & \overline{fn_f}\ \texttt{return}\ f\texttt{;} \\
 & | & \overline{fn_f}\ \texttt{return \{}\ \overline{x\texttt{:}f}\ \texttt{\};} \\
fn_f & ::= & \texttt{function}\ f\texttt{(}\overline{x}\texttt{) \{}\ \overline{x\ \texttt{=}\ \kappa_x\texttt{;}}\ \texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}\ ss\ \texttt{\}}
\end{array}
$$

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

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

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

$$
\begin{array}{rcl}
\kappa_X & ::= & X \;\texttt{|}\; \texttt{0} \\
& | & X \;\texttt{>>>}\; \texttt{0} \\
& | & \texttt{+}X \\
& | & \texttt{[}X\texttt{[0] >>> 0}, X\texttt{[1] | 0]} \\
& | & \texttt{[}X\texttt{[0] >>> 0}, X\texttt{[1] >>> 0]}
\end{array}
$$

$$
\begin{array}{rcl}
v & ::= & \kappa_{num} \\
& | & \texttt{[}\kappa_{num}, \kappa_{num}\texttt{]}
\end{array}
$$

$$
\begin{array}{rcl}
e & ::= & \kappa_{num} \\
& | & \kappa_{e} \\
& | & \kappa_{x\texttt{[}e\texttt{]}} \\
& | & x \\
& | & lval \;\texttt{=}\; e \\
& | & f\texttt{(}\overline{e}\texttt{)} \\
& | & unop\; e \\
& | & e\; binop\; e \\
& | & e \;\texttt{?}\; e \;\texttt{:}\; e \\
& | & \texttt{(}\overline{e}\texttt{)} \\
& | & \texttt{[}e, e\texttt{]} \\
& | & e\texttt{[0]} \\
& | & e\texttt{[1]} \\
\\
unop & ::= & \texttt{\~{}} \mid \texttt{-} \mid \texttt{!} \\
\\
binop & ::= & \texttt{<} \mid \texttt{<=} \mid \texttt{>} \mid \texttt{>=} \mid \texttt{!=} \mid \texttt{==} \\
& | & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \mid \texttt{\%} \\
& | & \texttt{|} \mid \texttt{\&} \mid \texttt{\^{}} \\
& | & \texttt{<<} \mid \texttt{>>} \mid \texttt{>>>} \\
\\
lval & ::= & x \\
& | & x\texttt{[}e\texttt{]}
\end{array}
$$

2 Type rules

$$
\begin{array}{rcl}
\sigma, \tau & ::= & \texttt{bits1} \mid \texttt{bits32} \mid \texttt{boolish} \\
& | & \texttt{int32} \mid \texttt{uint32} \\
& | & \texttt{int64} \mid \texttt{uint64} \\
& | & \texttt{float64} \\
& | & \texttt{array}_{\tau} \\
& | & \texttt{function} \\
& | & \texttt{jsval}
\end{array}
$$

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

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

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

$$
\begin{array}{rcl}
\mathit{type}(X\ \texttt{|}\ \texttt{0}) & = & \texttt{int32} \\
\mathit{type}(X\ \texttt{>>>}\ \texttt{0}) & = & \texttt{uint32} \\
\mathit{type}(\texttt{+}X) & = & \texttt{float64} \\
\mathit{type}(\texttt{[}X\texttt{[0]}\ \texttt{>>>}\ \texttt{0}, X\texttt{[1]}\ \texttt{|}\ \texttt{0]}) & = & \texttt{int64} \\
\mathit{type}(\texttt{[}X\texttt{[0]}\ \texttt{>>>}\ \texttt{0}, X\texttt{[1]}\ \texttt{>>>}\ \texttt{0]}) & = & \texttt{uint64}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{int32}, \texttt{uint32} & <: & \texttt{bits32} \\
\texttt{bits1} & <: & \texttt{boolish} \\
\texttt{bits32} & <: & \texttt{boolish} \\
\texttt{bits32} & <: & \texttt{float64} \\
\texttt{float64} & <: & \texttt{jsval} \\
\texttt{function} & <: & \texttt{jsval} \\
\texttt{array}_\tau & <: & \texttt{jsval}
\end{array}
$$

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

[T-Singleton]

$$
\frac{\forall f.\rho; \Delta \vdash \mathit{fn}_f\ \mathbf{ok} \qquad f \in \{\overline{f}\} \qquad \Delta(f) = (\overline{\sigma}) \to \tau \wedge \tau <: \texttt{jsval}}{\rho \vdash \overline{\mathit{fn}_f}\ \texttt{return}\ f\texttt{;}\ \mathbf{ok}}
$$

[T-Module]

$$
\frac{\forall f.\rho; \Delta \vdash \mathit{fn}_f\ \mathbf{ok} \qquad \{\overline{g}\} \subseteq \{\overline{f}\} \qquad \forall g.\Delta(g) = (\overline{\sigma}) \to \tau \wedge \tau <: \texttt{jsval}}{\rho \vdash \overline{\mathit{fn}_f}\ \texttt{return}\ \texttt{\{}\ \overline{x\texttt{:}g}\ \texttt{\};}\ \mathbf{ok}}
$$

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

[T-Function]

$$
\frac{\overline{\forall i.\mathit{type}(\kappa_{x_i}) = \sigma_i} \qquad \rho; \Delta; \{\overline{x : \sigma}, \overline{y : \mathit{type}(v)}\}; \emptyset \vdash ss : \Delta(f)/\mathsf{return}}{\rho; \Delta \vdash \texttt{function}\ f(\overline{x})\ \texttt{\{}\ \overline{x\ \texttt{=}\ \kappa_x\texttt{;}}\ \texttt{var}\ \overline{y\ \texttt{=}\ v}\texttt{;}\ ss\ \texttt{\}}\ \mathbf{ok}}
$$

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

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

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

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

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

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

$$\text{[T-Return]}\quad \frac{}{\rho;\Delta;\Gamma;L \vdash \texttt{return}\ \ v\texttt{;} : \mathit{type}(v)/\mathsf{return}}$$

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

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

$$\text{[T-For]}\quad \frac{\forall i \in \{1,2,3\}.\rho;\Delta;\Gamma \vdash e_i : \sigma_i \quad \rho;\Delta;\Gamma;L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \quad \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\})}{\rho;\Delta;\Gamma;L \vdash \texttt{for (}e_1\texttt{;}\ \ e_2\texttt{;}\ \ e_3\texttt{)}\ \ s : \tau/\varepsilon'}$$

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

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

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

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

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

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

[T-Switch]
$$\frac{\begin{array}{c}\rho; \Delta; \Gamma \vdash e : \sigma \\ \forall i.\rho; \Delta; \Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \\ \rho; \Delta; \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\varepsilon \\ \varepsilon \neq \mathsf{return} \vee \exists i.\varepsilon_i \cup \emptyset \neq \emptyset \\ \varepsilon' = (\varepsilon \cup \bigcup_i \varepsilon_i) - (L \cup \{\epsilon\})\end{array}}{\rho; \Delta; \Gamma; L \vdash \texttt{switch (}e\texttt{) \{}\ \overline{c}\ cd\ \texttt{\}} : \tau/\varepsilon'}$$

[T-SwitchReturn]
$$\frac{\begin{array}{c}\rho; \Delta; \Gamma \vdash e : \sigma \\ \forall i.\rho; \Delta; \Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \\ \forall i.\varepsilon_i \cup \emptyset = \emptyset \\ \rho; \Delta; \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\mathsf{return}\end{array}}{\rho; \Delta; \Gamma; L \vdash \texttt{switch (}e\texttt{) \{}\ \overline{c}\ cd\ \texttt{\}} : \tau/\mathsf{return}}$$

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

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

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

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

$$\text{[T-NUMBER]}\ \frac{}{\rho; \Delta; \Gamma \vdash \kappa_{num} : \mathit{type}(\kappa_{num})} \qquad \text{[T-CAST]}\ \frac{\rho; \Delta; \Gamma \vdash e : \tau \qquad e = f\texttt{(}\overline{e}\texttt{)} \Rightarrow f \in \mathit{dom}(\Delta)}{\rho; \Delta; \Gamma \vdash \kappa_e : \mathit{type}(\kappa_e)}$$

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

$$\text{[T-LOAD]}\ \frac{\mathit{type}(\rho(x)) = \texttt{array}_\tau \qquad \rho; \Delta; \Gamma \vdash e : \sigma \qquad \sigma <: \texttt{bits32} \qquad \mathit{type}(\kappa_{x\texttt{[}e\texttt{]}}) = \tau}{\rho; \Delta; \Gamma \vdash \kappa_{x\texttt{[}e\texttt{]}} : \tau} \qquad \text{[T-STORE]}\ \frac{\mathit{type}(\rho(x)) = \texttt{array}_\tau \qquad \rho; \Delta; \Gamma \vdash e_1 : \sigma \qquad \sigma <: \texttt{bits32} \qquad \rho; \Delta; \Gamma \vdash e_2 : \tau}{\rho; \Delta; \Gamma \vdash x\texttt{[}e_1\texttt{] = } e_2 : \tau}$$

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

$$\text{[T-STDLIB]}\ \frac{e = f\texttt{(}\overline{e}\texttt{)} \qquad f \notin \mathit{dom}(\Delta) \qquad f \notin \mathit{dom}(\Gamma) \qquad \rho(f) = (\overline{\sigma}) \to \tau \qquad \forall i. \rho; \Delta; \Gamma \vdash e_i : \sigma_i \qquad \mathit{type}(\kappa_e) = \tau}{\rho; \Delta; \Gamma \vdash \kappa_e : \tau}$$

$$\text{[T-FFI]}\ \frac{f \notin \mathit{dom}(\Delta) \qquad f \notin \mathit{dom}(\Gamma) \qquad \rho(f) = \texttt{function} \qquad \forall i. \rho; \Delta; \Gamma \vdash e_i : \sigma_i}{\rho; \Delta; \Gamma \vdash f\texttt{(}\overline{e}\texttt{)} : \texttt{jsval}}$$

$$\text{[T-CONDITIONAL]}\ \frac{\rho; \Delta; \Gamma \vdash e_1 : \texttt{boolish} \qquad \forall i \in \{2,3\}. \rho; \Delta; \Gamma \vdash e_i : \tau}{\rho; \Delta; \Gamma \vdash e_1 \texttt{ ? } e_2 \texttt{ : } e_3 : \tau} \qquad \text{[T-PAREN]}\ \frac{\forall i \leq n. \rho; \Delta; \Gamma \vdash e_i : \tau_i}{\rho; \Delta; \Gamma \vdash \texttt{(}\overline{e}\texttt{)} : \tau_n} \qquad \text{[T-SUB]}\ \frac{\rho; \Delta; \Gamma \vdash e : \sigma \qquad \sigma <: \tau}{\rho; \Delta; \Gamma \vdash e : \tau}$$