# ADL: Architecture Description Language

Toshihiro YOSHINO

$$
\begin{array}{llll}
\text{Integer} & n & ::= & \cdots, -1, 0, 1, 2, \cdots \\
\text{Pointer} & m & ::= & \ell + n \\
\text{Value} & v & ::= & \texttt{junk} \mid n \mid m \qquad :: \mathsf{Value}
\end{array}
$$

図 1: ADL values

$$
\begin{array}{llll}
\text{Byte Value} & b & \in & [0, 255] \\
\text{Atomic Value} & a & ::= & \texttt{junk} \mid b \mid m[n] \qquad :: \mathsf{Atom} \\
\text{Data} & d & ::= & \langle a_0, a_1, \cdots, a_{n-1} \rangle \quad :: \mathsf{Data} \\
\text{Memory Block} & k & ::= & \langle c_0; c_1; \cdots; c_{n-1} \rangle \qquad \text{(Code Block)} \\
 & & \mid & d \qquad \text{(Data Block)} \\
 & & & \\
\text{Register File} & R & = & \{\texttt{r}_1 \mapsto d_1, \texttt{r}_2 \mapsto d_2, \cdots, \texttt{r}_\texttt{N} \mapsto d_N\} \\
\text{Memory} & M & = & \{\ell_1 \mapsto k_1, \ell_2 \mapsto k_2, \cdots\} \\
\text{Temporary Variable} & V & = & \{A \mapsto v_1, B \mapsto v_2, \cdots\} \\
\text{Machine State} & S & = & (R, M, V, m)
\end{array}
$$

図 2: Storages in the abstract machine

$$
\begin{array}{llcll}
\text{Left Value} & l & ::= & \mathtt{r_i}[n, n] & \text{(Register)} \\
 & & \mid & *[n]e_v & \text{(Memory Reference)} \\
\text{Expression} & e_v & ::= & v & \text{(Literal)} \\
 & & \mid & l & \text{(Left value)} \\
 & & \mid & x & \text{(Variable)} \\
 & & \mid & e_v\ op_b\ e_v \mid op_u e_v & \\
 & & & & \text{(Arithmetic)} \\
 & op_b & ::= & + \mid - \mid * \mid / \mid \% \mid \& \mid | \mid \hat{} & \\
 & op_u & ::= & - \mid \tilde{} & \\
 & & & & \\
\text{Boolean Expr.} & e_b & ::= & e_v\ cmp\ e_v & \text{(Comparison)} \\
 & & \mid & e_b \wedge e_b \mid e_b \vee e_b & \text{(Logical Operator)} \\
 & & \mid & !e_b & \text{(Negation)} \\
 & cmp & ::= & == \mid != \mid \cdots & \\
 & & & & \\
\text{Command} & c & ::= & \mathtt{nop} & \text{(No Operation)} \\
 & & \mid & \mathtt{error} & \text{(Runtime Error)} \\
 & & \mid & l = e_v & \text{(Assignment)} \\
 & & \mid & x = e_v & \text{(Variable Definition)} \\
 & & \mid & \mathtt{goto}\, e_v & \text{(Jump)} \\
 & & \mid & \mathtt{if}\, e_b\, \mathtt{then}\, c\, \mathtt{else}\, c & \\
 & & & & \text{(Conditional)} \\
 & & \mid & \mathtt{if}\, e_v : kind\, \mathtt{then}\, c\, \mathtt{else}\, c & \\
 & & & & \text{(Conditional by Kind)} \\
 & kind & ::= & \mathtt{junk} \mid \mathtt{int} \mid \mathtt{pointer} & 
\end{array}
$$

図 3: ADL syntax