New Theory for Verilog Size Checking

Path

Definition of paths and their relation to sub-expressions.

Expressions

- $\mathcal{A}$ is the set of atoms in an expression. In the AST, they correspond to leaves. In System-Verilog, they refer to what the standard calls an "operand" (variables, integers, function calls, slices of a variable, etc.).
- $\mathcal{R}$ is the set of resizable expressions. It corresponds to the expression whose top level is one of: atom, comparisons, logic operation, reduction, assignments, shift assignments, concatenation, replication and inside operation.
- $\mathcal{E}$ is the set of System-Verilog expressions. We have $\mathcal{A} \subset \mathcal{E}$. An expression either contains an expression or is an atom.

Rules

We use the following notations:

- Γ maps atoms to their size,
- Φ maps lvalues to their size,
- The statement $e \Rightarrow t \dashv f$ means "e has size t, with f mapping each sub-expressions of e to their size",
- The statement $e \Leftarrow t \dashv f$ means "e can be resized to t, with f mapping each sub-expressions of e to their size".

Function combinator

$$\text{Binary}(t, f, g) : \begin{cases} [] & \mapsto & t \\ 0 :: p & \mapsto & f(p) \\ 1 :: p & \mapsto & g(p) \end{cases} \qquad \text{Unary}(f, t) : \begin{cases} [] & \mapsto & t \\ 0 :: p & \mapsto & f(p) \end{cases}$$

$$\text{Narry}(t, f_1, \ldots, f_k) : \begin{cases} [] & \mapsto & t \\ i :: p & \mapsto & f_i(p) \end{cases} \qquad \text{Ternary}(t, f, g, h) : \begin{cases} [] & \mapsto & t \\ 0 :: p & \mapsto & f(p) \\ 1 :: p & \mapsto & g(p) \\ 2 :: p & \mapsto & h(p) \end{cases}$$

Base case

$$\frac{\Gamma(e) = s \qquad e \in \mathcal{A}}{e \Rightarrow s \dashv \{[] \mapsto s\}}\ \text{Atom}\Rightarrow$$

Resize case

$$\frac{e \Rightarrow s \dashv f \qquad s \leqslant t \qquad e \in \mathcal{R}}{e \Leftarrow t \dashv f\big[[] \mapsto t\big]}\ \text{Resize}\Leftarrow$$

Operators

- $\oplus \in \{$`+`, `-`, `*`, `/`, `%`, `&`, `|`, `^`, `^~`, `~^`$\}$:

$$\frac{a \Rightarrow t \dashv f_a \qquad b \Leftarrow t \dashv f_b}{a \oplus b \Rightarrow t \dashv \text{Binary}(t, f_a, f_b)}\ \text{LBinOp}\Rightarrow \qquad \frac{a \Leftarrow t \dashv f_a \qquad b \Rightarrow t \dashv f_b}{a \oplus b \Rightarrow t \dashv \text{Binary}(t, f_a, f_b)}\ \text{RBinOp}\Rightarrow$$

$$\frac{a \Leftarrow t \dashv f_a \qquad b \Leftarrow t \dashv f_b}{a \oplus b \Leftarrow t \dashv \text{Binary}(t, f_a, f_b)}\ \text{BinOp}\Leftarrow$$

- $\oplus \in \{$`+`, `-`, `++`, `--`, `~`$\}$:

$$\frac{e \Rightarrow t \dashv f}{\oplus e \Rightarrow t \dashv \text{Unary}(t, f)}\ \text{UnOp}\Rightarrow \qquad \frac{e \Leftarrow t \dashv f}{\oplus e \Leftarrow t \dashv \text{Unary}(t, f)}\ \text{UnOp}\Leftarrow$$

- $\oplus \in \{$`===`, `!==`, `==?`, `!=?`, `==`, `!=`, `>`, `>=`, `<`, `<=`$\}$:

$$\frac{a \Rightarrow t \dashv f_a \qquad b \Leftarrow t \dashv f_b}{a \oplus b \Rightarrow 1 \dashv \text{Binary}(1, f_a, f_b)}\ \text{LCmp}{\Rightarrow} \qquad \frac{a \Leftarrow t \dashv f_a \qquad b \Rightarrow t \dashv f_b}{a \oplus b \Rightarrow 1 \dashv \text{Binary}(1, f_a, f_b)}\ \text{RCmp}{\Rightarrow}$$

- $\oplus \in \{$`&&`, `||`, `->`, `<->`$\}$:

$$\frac{a \Rightarrow t_a \dashv f_a \qquad b \Rightarrow t_b \dashv f_b}{a \oplus b \Rightarrow 1 \dashv \text{Binary}(1, f_a, f_b)}\ \text{Logic}{\Rightarrow}$$

- $\oplus \in \{$`&`, `~&`, `|`, `~|`, `^`, `~^`, `^~`, `!`$\}$:

$$\frac{e \Rightarrow t \dashv f}{\oplus e \Rightarrow 1 \dashv \text{Unary}(1, f)}\ \text{Red}{\Rightarrow}$$

- $\oplus \in \{$`>>`, `<<`, `**`, `>>>`, `<<<`$\}$:

$$\frac{a \Rightarrow t \dashv f_a \qquad b \Rightarrow t_b \dashv f_b}{a \oplus b \Rightarrow t \dashv \text{Binary}(t, f_a, f_b)}\ \text{Shift}{\Rightarrow} \qquad \frac{a \Leftarrow t \dashv f_a \qquad b \Rightarrow t_b \dashv f_b}{a \oplus b \Leftarrow t \dashv \text{Binary}(t, f_a, f_b)}\ \text{Shift}{\Leftarrow}$$

- Assigment:

$$\frac{\phi(l) = t \qquad e \Leftarrow t \dashv f}{(l \texttt{=} e) \Rightarrow t \dashv \text{Unary}(t, f)}\ \text{LAssign}{\Rightarrow} \qquad \frac{\phi(l) = t \qquad e \Rightarrow t_e \dashv f \qquad t < t_e}{(l \texttt{=} e) \Rightarrow t \dashv \text{Unary}(t, f)}\ \text{RAssign}{\Rightarrow}$$

- If expression:

$$\frac{e \Rightarrow t_e \dashv f_e \qquad a \Rightarrow t \dashv f_a \qquad b \Leftarrow t \dashv f_b}{e \texttt{?} a \texttt{:} b \Rightarrow t \dashv \text{Ternary}(t, f_e, f_a, f_b)}\ \text{LCond}{\Rightarrow} \qquad \frac{e \Rightarrow t_e \dashv f_e \qquad a \Leftarrow t \dashv f_a \qquad b \Rightarrow t \dashv f_b}{e \texttt{?} a \texttt{:} b \Rightarrow t \dashv \text{Ternary}(t, f_e, f_a, f_b)}\ \text{RCond}{\Rightarrow}$$

$$\frac{e \Rightarrow t_e \dashv f_e \qquad a \Leftarrow t \dashv f_a \qquad b \Leftarrow t \dashv f_b}{e \texttt{?} a \texttt{:} b \Leftarrow t \dashv \text{Ternary}(t, f_e, f_a, f_b)}\ \text{Cond}{\Leftarrow}$$

- Concatenation:

$$\frac{e_1 \Rightarrow t_1 \dashv f_i \qquad \ldots \qquad e_k \Rightarrow t_k \dashv f_k \qquad t = t_1 + \cdots + t_k}{\{e_1, \ldots, e_k\} \Rightarrow t \dashv \text{Narry}(t, f_1, \ldots, f_k)}\ \text{Concat}{\Rightarrow}$$

- Replication:

$$\frac{i \in \mathbb{N} \qquad e \Rightarrow t_e \dashv f \qquad t = i \times t_e}{\{i\ e\} \Rightarrow t \dashv \text{Unary}(t, f)}\ \text{Repl}{\Rightarrow}$$