

# New Theory for Verilog Size Checking

## Path

A path is represented as a list of natural numbers that encodes navigation through the abstract syntax tree: starting from the root, each number indicates which child to visit next (with 0 denoting the first child, 1 the second, and so forth). For example, the path  $[1, 0]$  refers to the first child of the second child of the root expression. The empty path  $[]$  refers to the root expression itself.

## 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:

- $\Gamma$  maps atoms to their size,
- $\Phi$  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

$$\begin{aligned} \text{Unary}(f, t) &::= \begin{cases} [] & \mapsto t \\ 0 :: p & \mapsto f(p) \end{cases} & \text{Binary}(t, f, g) &::= \begin{cases} [] & \mapsto t \\ 0 :: p & \mapsto f(p) \\ 1 :: p & \mapsto g(p) \end{cases} \\ \text{Ternary}(t, f, g, h) &::= \begin{cases} [] & \mapsto t \\ 0 :: p & \mapsto f(p) \\ 1 :: p & \mapsto g(p) \\ 2 :: p & \mapsto h(p) \end{cases} & \text{Nary}(t, f_1, \dots, f_k) &::= \begin{cases} [] & \mapsto t \\ i :: p & \mapsto f_i(p) \end{cases} \end{aligned}$$

## Resizing

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

H

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

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

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

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

## Synthesize

$$\frac{\Gamma(e) = s \quad e \in \mathcal{O}}{e \Rightarrow s \dashv \{\square \mapsto s\}} \text{Operand} \Rightarrow$$

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

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

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

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

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

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

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

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

$$\frac{\phi(l) = t \quad e \Leftarrow t \dashv f}{(l = e) \Rightarrow t \dashv \text{Unary}(t, f)} \text{LAssign} \Rightarrow$$

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

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

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

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

$$\frac{e_1 \Rightarrow t_1 \dashv f_1 \quad \dots \quad e_k \Rightarrow t_k \dashv f_k \quad t = t_1 + \dots + t_k}{\{e_1, \dots, e_k\} \Rightarrow t \dashv \text{Nary}(t, f_1, \dots, f_k)} \text{Concat} \Rightarrow$$