11. SHIFT AND ROTATE OPERATORS

**LRM REFERENCES: 7.2**

11.1. BACKGROUND

VHDL'87 did not define any shift or rotate operators. The main problem here was reaching a consensus on a minimum set of operators. This has now been done, and VHDL'92 predefines four shift and two rotate operators.

11.2. DESCRIPTION

The six shift and rotate operators are binary operators called **sll**, **srl**, **sla**, **sra**, **rol**, and **ror**. They are only defined for one-dimensional array types (vectors) whose element type is either BIT or BOOLEAN. The left operand must be of that type. The right operand is of type INTEGER:

- if this operand equals zero, no operation at all is performed: (A sll 0) is a null operation whose result is A.
- if the right operand is positive, then the shift or rotate operation is repeated this number of times: (A sll 5) performs a one-bit-shift five times.
• if the right operand is negative, then the opposite shift is performed a number of times corresponding to the absolute value of the right operand: \( (A \text{ rol} -6) \) is equivalent to \( (A \text{ ror} 6) \).

Of course, the main value of having integer as the right operand is the possibility of using an expression \( (A \text{ ror} N*P/I) \) that is dynamically evaluated. In this case, the operator must be considered as a "generic" rotate operator \( \text{ror} \) or \( \text{rol} \) depending on the sign of the expression.

The semantics of the basic operation (one-bit rotate or shift) is illustrated in figure 11.1. Some of these operations (\( \text{sll} \), \( \text{srl} \), \( \text{sla} \), \( \text{sra} \)) imply losing the value of one element. In some cases (\( \text{sll} \) and \( \text{srl} \)), a fill value is used. It is defined as the value of the attribute 'LEFT' on an element type of the array: '0' is the fill value for BIT_VECTOR operations and FALSE is the fill value for boolean vectors. The only possibility of changing this fill value is to overload the operators.

![Shift and Rotate Operator Semantics](image.png)