64-bit ALU in Verilog HDL

Ananya Sane (2020102007)   L Lakshmanan (2020112024)

Inputs

Operands, Opcode.

Outputs

Result, Overflow flag, Zero flag.

Functions

The ALU has 4 functions that are binary or unary, and that can be used on 64-bit binary numbers. These 4 functions are

- Add → 00
- Subtract → 01
- AND → 10
- XOR → 11

Each of these functions have their own two bit opcodes, which have been specified beside them. These opcodes are what the ALU uses to determine which operation needs to be computed.

Two flags have been initialised in the ALU, one for the overflow and one for indicating if the output is a zero. The overflow flag is 1 when there is an overflow in the performed operation and is zero in all other cases, while the zero flag is 1 when the output is zero. Results of the operations are stored in the `res` register.

Add function

Takes two 64-bit inputs. Returns the sum as a result, along with the appropriate flags. The 64-bit adder is constructed using loops and full adder modules, which have been instantiated using the `generate` and the for loop structures.

Subtract function

Takes two 64-bit inputs. Returns the difference as a result, along with the appropriate flags. The subtraction module has been constructed using the 64-bit

adder as a module, and by complementing the required number using `generate` block, for loops and the `not` primitive.

AND function

Takes two 64-bit inputs. Returns the logical AND value of the two as a result, along with the appropriate flags. Constructed using just the gate primitive, the `generate` block and for loop structure.

XOR function

Takes two 8 bit inputs. Returns the logical XOR value of the two as a result, along with the appropriate flags. Constructed using just the gate primitive and the `generate` block and for loop structure.

Testbenches

Each function's testbench seeds inputs with random numbers and passes them to the ALU, with the final result being stored in the `res` register. For each set of inputs, the value `res - (a\ OP\ b)` is printed. There are two cases: - In most cases, the output is 0; this means the ALU works and both results match. - In the cases where the expression is non zero, the overflow bit is set; this makes sense as `res` stores a value that is not the correct answer, and hence it is not equal to the value of `a\ OP\ b`.

The outputs of the testbenches are written to the .txt files attached.