```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_misc.or_reduce;
ENTITY alu32 IS
    PORT( A, B: in std_logic_vector (31 downto 0);
        ALUOp: in std_logic_vector (3 downto 0);
        RESULT: out std_logic_vector (31 downto 0);
        Z, V, C: out std_logic );
END alu32;

architecture big_alu of alu32 is
    signal intRes, carry: std_logic_vector(31 downto 0);
    signal set : std_logic;
    component bit_slice port(
        a,b,Less,Ainvert,carryIn        : in std_logic;
        Op                              : in std_logic_vector (2 downto 0);
        result,carryOut                 : out std_logic
     ); end component;
     component MSB_slice port(
        a,b,Less,Ainvert,carryIn        : in std_logic;
        Op                              : in std_logic_vector (2 downto 0);
        result,carryOut,set,overflow    : out std_logic
     ); end component;
begin
    LSB: bit_slice PORT MAP(
        a=>A(0), b=>B(0), Less=>set, Ainvert=>ALUOp(3), carryIn=>ALUOp(2),
        Op=>ALUOp(2 downto 0),
        result=>intRes(0), carryOut=>carry(0) );
    bitSliceSetup: for x in 1 to 30 generate
        BITx: bit_slice PORT MAP(
            a=>A(x), b=>B(x), Less=>'0', Ainvert=>ALUOp(3), carryIn=>carry(x-1),
            Op=>ALUOp(2 downto 0),
            result=>intRes(x), carryOut=>carry(x) );
    end generate bitSliceSetup;
    MSB: MSB_slice PORT MAP(
            a=>A(31), b=>B(31), Less=>'0', Ainvert=>ALUOp(3), carryIn=>carry(30),
            Op=>ALUOp(2 downto 0),
            result=>intRes(31), carryOut=>C,set=>set, overflow=>V );

    --Handle zero flag
    Z <= not or_reduce(intRes);
    --connect result output to signal
    result <= intRes;

end big_alu;
```