

```

Apr 21, 11 16:14      alu.sv      Page 1/1

module alu(
    input      [7:0] in_a    , /* input a */
    input      [7:0] in_b    , /* input b */
    input      [3:0] opcode  , /* opcode input */
    output reg [7:0] alu_out , /* alu output */
    output reg   alu_zero  , /* logic '1' when alu_output [7:0] is all zero */
    output reg   alu_carry /* indicates a carry out from ALU */
);

parameter c_add      = 4'h1; /* in_a + in */
parameter c_sub      = 4'h2; /* in_a - in */
parameter c_inc      = 4'h3; /* in_a + 1 */
parameter c_dec      = 4'h4; /* in_a - 1 */
parameter c_or       = 4'h5; /* in_a OR in_b */
parameter c_and      = 4'h6; /* in_a AND in_b */
parameter c_xor      = 4'h7; /* in_a XOR in_b */
parameter c_shr      = 4'h8; /* in_a is shifted one place right, zero shifted in */
parameter c_shl      = 4'h9; /* in_a is shifted one place left, zero shifted in */
parameter c_onescomp = 4'hA; /* in_a gets "ones complemented" */
parameter c_twoscomp = 4'hB; /* in_a gets "twos complemented" */

always_comb
begin
    alu_carry = 0;
    case (opcode)
        c_add      : {alu_carry, alu_out} = in_a + in_b;
        c_sub      : {alu_carry, alu_out} = in_a - in_b;
        c_inc      : {alu_carry, alu_out} = in_a + 8'h01;
        c_dec      : {alu_carry, alu_out} = in_a - 8'h01;
        c_or       : {alu_carry, alu_out} = in_a | in_b;
        c_and      : {alu_carry, alu_out} = in_a & in_b;
        c_xor      : {alu_carry, alu_out} = in_a ^ in_b;
        c_shr      : {alu_out, alu_carry} = in_a >> 1;
        c_shl      : {alu_carry, alu_out} = in_a << 1;
        c_onescomp : {alu_carry, alu_out} = (in_a ^ 8'hFF);
        c_twoscomp : {alu_carry, alu_out} = (in_a ^ 8'hFF) + 8'h01;
        default    : {alu_carry, alu_out} = 9'bxxxxxxxx;
    endcase
    alu_zero = ~|alu_out;
end
endmodule

```

```

Apr 21, 11 17:15      writeup.txt      Page 1/1

1. The total area used by the alu is 1664.132701.

2. There were 13 different types of gates used by my design:
AND2X1      saed90nm_typ
AO221X1     saed90nm_typ
INVX0       saed90nm_typ
MUX21X1     saed90nm_typ
MUX41X1     saed90nm_typ
NAND2X0     saed90nm_typ
NAND3X0     saed90nm_typ
NOR2X0      saed90nm_typ
OAI21X1     saed90nm_typ
OR4X1       saed90nm_typ
XNOR2X1     saed90nm_typ
XOR2X1      saed90nm_typ
alu_DW01_addsub_0
    FADDX1   saed90nm_typ
    XOR2X1   saed90nm_typ

3. The number of cells used in the design is 120.
The number of gates is 1664.132701 / 5.5296 = 300.9 ~= 301.

4. The hierarchical block introduced to my design was the module alu_DW01_addsub_0. This is most likely the ripple adder introduced when I compiled my design with the statement "in_a + in_b". It is implemented as a chain of full adders.

5. The maximum delay path is 2.53 ps. The beginning point was input external delay and the endpoint was data arrival time.

```

Apr 21, 11 17:01

alu_original.list

Page 1/1

ps	opcode	A	B	c	alu_out	z
9000	0001	11111111	11111111	1	11111110	0
19000	0001	00000001	11111111	1	00000000	1
29000	0010	10010010	01010000	0	01000010	0
39000	0010	11111111	01011000	0	10100111	0
49000	0011	00000000	xxxxxxx	0	00000001	0
59000	0100	00000000	xxxxxxx	1	11111111	0
69000	0101	11110000	10101010	0	11111010	0
79000	0110	11110000	10101010	0	10100000	0
89000	0111	11110000	10101010	0	01011010	0
99000	1000	11111111	xxxxxxx	1	00111111	0
109000	1001	11111111	xxxxxxx	1	11111110	0
119000	1010	10101010	xxxxxxx	0	01010101	0
129000	1011	10101010	xxxxxxx	0	01010110	0

Apr 21, 11 17:02

alu_optimized.list

Page 1/1

ps	opcode	A	B	c	alu_out	z
9000	0001	11111111	11111111	1	11111110	0
19000	0001	00000001	11111111	1	00000000	1
29000	0010	10010010	01010000	0	01000010	0
39000	0010	11111111	01011000	0	10100111	0
49000	0011	00000000	xxxxxxx	0	00000001	0
59000	0100	00000000	xxxxxxx	1	11111111	0
69000	0101	11110000	10101010	0	11111010	0
79000	0110	11110000	10101010	0	10100000	0
89000	0111	11110000	10101010	0	01011010	0
99000	1000	11111111	xxxxxxx	1	00111111	0
109000	1001	11111111	xxxxxxx	1	11111110	0
119000	1010	10101010	xxxxxxx	0	01010101	0
129000	1011	10101010	xxxxxxx	0	01010110	0