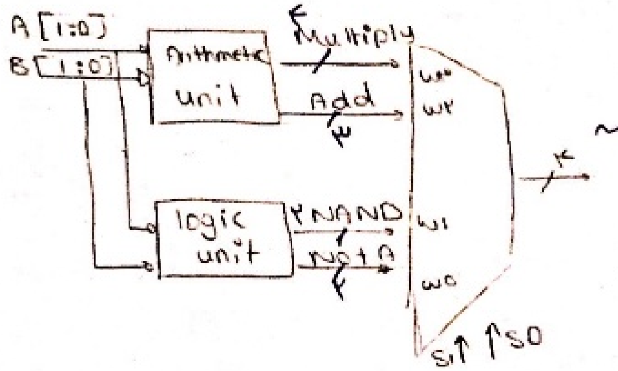
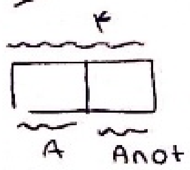
ALU → محاسبه sel=1 ~> (جمع / ضرب / تفریق)
→ منطق sel=2 ~> (NOR / Not / Or و and)
→ کنترل sel=3 ~> تعیین واحد عملیاتی



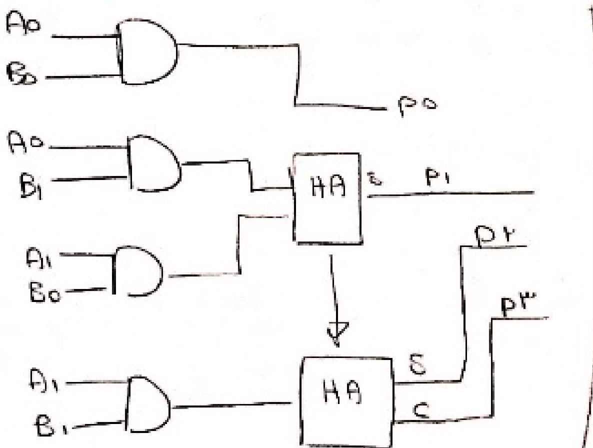
باید همه ورودی های Multiplexer ۴ بیتی باشند بنابراین
۲ بیت به هر دو تای منطقی به بیت‌ها اضافه می‌کنیم



→ assign {A, ~A} (۲ بیت)

→ assign {1'b0, A+B}

| 00 | Nand | → assign {1'b0, 1'b0, ~(A+B)}



مدار اول 8

```
module multiplexer(
    input [3:0] w3,
    input [3:0] w2,
    input [3:0] w1,
    input [3:0] w0,
    input [1:0] sel,
    output [3:0] y
);
assign y = sel[1] ? (sel[0] ? w3 : w2) : (sel[0] ? w1 : w[illegible]
```

```
module alu (
   input [1:0] A,
   input [1:0] B,
   input [1:0] sel,
   output [3:0] y
);
  wire [3:0] Add;
  wire [3:0] Not;
  wire [3:0] Nand;
  wire [3:0] Multiply;
  assign Add = {1'b0, {1'b0, A} + {1'b0, B}};
  assign Not = {A, ~A};
  assign Nand = {1'b0, 1'b0, ~(A&B)};
  assign Multiply = {{1'b0, 1'b0, A} * {1'b0, 1'b0, B}};
  multiplexer mux (Multiply, Add, Nand, Not, sel, y);
endmodule
```

```
module alu (
input [1:0] A,
input [1:0] B,
input [1:0] sel,
output [3:0] y
);

wire [2:0] w;
wire [3:0] Multiple;
wire c;

// not
wire [3:0] Not;
wire [1:0] anot;
assign anot = ~A;
assign Not = {A, anot};

# nand
wire [3:0] Nand;
wire [1:0] w3;
assign w3 = ~(A & B);
assign Nand = {1'b0, 1'b0, w3};

# add
wire [3:0] Add;
wire [2:0] w2;
assign w2 = (A + B);
assign Add = {1'b0, w2};

# Multiple
wire [2:0] w;
wire [3:0] Multiple;
wire c;
and
  g1(Multiple[0], A[0], B[0]),
  g2 (w[0], A[1], B[0])
  g3 (w[1], A[0], B[1])
  g4 (w[2], A[1], B[1]);
halfAdder
    HA1(w[0], w[1], Multiple[1], c),
    HA2 (w[2], c, Multiple[2], Multiple[3]);
multiplexer mux (Multiple, Add, Nand, Not, Sel, y);
endmodule
```

1 Half Adder) طراحی ماژول

```
module halfAdder(
    input A,
    input B,
    output sum,
    output c
);

Xor g1(sum, A, B);
and g2 (C, A, B);
endmodule
```

```
module multiplexer(
input [3:0] w4,
input [3:0] w2,
input [3:0] w1,
input [3:0] w0,
input [1:0] sel,
output [3:0] y
);

assign y = sel[1] ? (sel[0] ? w4 : w2) : (sel[0] ? w1 : w0);
```