```
Library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED;
use work.custom_pack.all;
use IEEE.NUMERIC_STD.ALL;
entity CU is
Generic(d_w: integer := 4);
Port (
opr1: in std_logic_vector(d_w-1 downto 0);
opr2: in std_logic_vector(d_w-1 downto 0);
Sel: in std_logic_vector(4 downto 0);
res: out std_logic_vector(d_w-1 downto 0));
end CU;
architecture Behavioral of CU is
begin
CU_proc : process(opr1,opr2,Sel)
variable T:std_logic_vector(2*d_w-1 downto 0);
begin
case Sel is
 when "00001"=> res <= opr1 and opr2;
 when "00010"=> res <= opr1 or opr2;
 when "00011"=> res <= opr1 nand opr2;
 when "00100"=> res <= opr1 nor opr2;
 when "00101"=> res <= opr1 xor opr2;
 when "00110"=> res <= opr1 xnor opr2;
 when "00111"=> res <= std_logic_vector(unsigned(opr1) +
unsigned(opr2));
 when "01000"=> res <= std_logic_vector(unsigned(opr1) -
unsigned(opr2));
 when "01001"=>
 T := std_logic_vector(unsigned(opr1) * unsigned(opr2));
 res<=T(d_w-1 downto 0);
 when "01010"=>
 if (opr1>opr2) then
 res <=(others => '1');
 else
 res <= (others => '0');
```

```
 end if;
 When "01011"=>
 if (opr1<opr2) then
 res <=(others => '1');
 else
 res <=(others => '0');
 end if;
 When "01100"=>
 if (opr1=opr2) then
 res <=(others => '1');
 else
 res <=(others => '0');
 end if;
 When "01101"=>
 if (opr1>=opr2) then
 res <=(others => '1');
 else
 res <=(others => '0');
 end if;
 When "01110"=>
 if (opr1<=opr2) then
 res <=(others => '1');
 else
 res <=(others => '0');
 end if;
 When "01111"=>
 if (opr1/=opr2) then
 res <=(others => '1');
 else
 res <=(others => '0');
 end if;
 when "10000"=>
 res <= to_stdlogicvector(to_bitvector(opr1) sla
to_integer(unsigned(opr2)));
 when "10001"=>
 res <= to_stdlogicvector(to_bitvector(opr1) sra
to_integer(unsigned(opr2)));
```

```
 when "10010"=>
 res <= to_stdlogicvector(to_bitvector(opr1) rol
to_integer(unsigned(opr2)));
 when "10011"=>
 res <= to_stdlogicvector(to_bitvector(opr1) ror
to_integer(unsigned(opr2)));
 when "10100"=>
 res <= to_stdlogicvector(to_bitvector(opr1) sll
to_integer(unsigned(opr2)));
 when "10101"=>
 res <= to_stdlogicvector(to_bitvector(opr1) srl
to_integer(unsigned(opr2)));
 when others =>
 res <=(others => 'Z');
 end case;
end process;
end Behavioral;
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use work.custom_pack.all;
entity storageUnit is
generic (d_w: integer:= 4);
Port (
  D: in std_logic_vector(d_w - 1 downto 0);
  w_en, clk: in std_logic;
  Q: out std_logic_vector(d_w - 1 downto 0));
end storageUnit;
architecture Behavioral of storageUnit is
signal stored_data: std_logic_vector(d_w - 1 downto 0);
begin
store: process(D, w_en, clk)
begin
```

```
if(rising_edge(clk)) then
    if(w_en = '1') then
        stored_data <= D;
    elsif(w_en = '0') then
        Q <= stored_data;
    end if;
end if;
end process store;
end Behavioral;
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use work.custom_pack.all;
use IEEE.NUMERIC_STD.ALL;
entity MUX_4x1 is
Generic(d_w: integer := 4);
Port ( a1,a2,a3,a4: in std_logic_vector(d_w-1 downto 0);
S: in std_logic_vector(1 downto 0);
cu_Output: inout std_logic_vector(d_w-1 downto 0));
end MUX_4x1;
architecture Behavioral of MUX_4x1 is
begin
Mux_4x1_proc: process(a1,a2,a3,a4,S)
 begin
 case S is
 when "00"=> cu_Output <=a1;
 when "01"=> cu_Output <=a2;
 when "10"=> cu_Output <=a3;
 when "11"=> cu_Output <=a4;
 when others=> cu_Output <="ZZZZ";
 end case;
 end process;
```

```
end Behavioral;




library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use work.custom_pack.all;
entity storageUnit_Operation is
generic (d_w: integer:= 5);
Port (
  D: in std_logic_vector(d_w - 1 downto 0);
  w_en, clk: in std_logic;
  Q: out std_logic_vector(d_w - 1 downto 0));
end storageUnit_Operation;
architecture Behavioral of storageUnit_Operation is
signal stored_data: std_logic_vector(d_w - 1 downto 0);
begin
store: process(D, w_en, clk)
begin
if(rising_edge(clk)) then
    if(w_en = '1') then
        stored_data <= D;
    elsif(w_en = '0') then
        Q <= stored_data;
    end if;
end if;
end process store;
end Behavioral;




library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use work.custom_pack.all;
entity final is
```

```
generic(d_w: integer:= 4);
Port (input1, input2: in arr_1d(0 to 3)(d_w - 1 downto 0);
cu_Output: out arr_2d(0 to 3)(0 to 3)(d_w - 1 downto 0);
final_output: out arr_1d(0 to 3)(d_w - 1 downto 0);
Sel: in arr_2d(0 to 3)(0 to 3)(4 downto 0);
mux_sel : in arr_2d(0 to 3)(0 to 7)(1 downto 0);
clk: in std_logic; w_en_ina,w_en_inb,w_en_in, w_en_out: in arr_2d_b
(0 to 3)(0 to 3));
end final;
architecture Behavioral of final is
signal mux_output: arr_2d(0 to 3)(0 to 7)(d_w -1 downto 0);
signal A_output,B_output,Y_output,R_output: arr_2d(0 to 3)(0 to
3)(d_w -1 downto 0);
signal operation: arr_2d(0 to 3)(0 to 3)(4 downto 0);
signal gnd: std_logic_vector(d_w - 1 downto 0):= (others => '0');
begin
--CU(0,0)
Mu1: entity work.Mux_4x1(Behavioral)
port map(
 a1 => input1(0),
 a2 => Y_output(3)(0),
 a3 => gnd,
 a4 => gnd,
 S => mux_sel(0)(0),
 cu_Output => mux_output(0)(0));
Mu2: entity work.Mux_4x1(Behavioral)
port map(
a1 => input2(0),
a2 => Y_output(3)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(0)(1),
cu_Output => mux_output(0)(1));
Au1: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(0) ,
w_en => w_en_ina(0)(0),
```

```
clk=>clk,
Q => A_output(0)(0));
Bu1: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(1) ,
w_en => w_en_inb(0)(0),
clk=>clk,
Q => B_output(0)(0));
store_Op1: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(0)(0) ,
w_en => w_en_in(0)(0),
clk=>clk,
Q => operation(0)(0));
CU1: entity work.CU(Behavioral)
port map(
opr1 =>A_output(0)(0),
opr2 =>B_output(0)(0),
Sel => operation(0)(0),
res => R_output(0)(0));
Yu1: entity work.storageUnit(Behavioral)
port map(
D => R_output(0)(0) ,
w_en => w_en_out(0)(0),
clk=>clk,
Q => Y_output(0)(0));

--CU(0,1)
Mu3: entity work.Mux_4x1(Behavioral)
port map(
a1 => input1(1),
a2 => Y_output(0)(0),
a3 => Y_output(3)(1),
a4 => gnd,
S => mux_sel(0)(2),
cu_Output => mux_output(0)(2));
Mu4: entity work.Mux_4x1(Behavioral)
```

```
port map(
a1 => input2(1),
a2 => Y_output(0)(0),
a3 => Y_output(3)(1),
a4 => gnd,
S => mux_sel(0)(3),
cu_Output => mux_output(0)(3));
Au2: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(2) ,
w_en => w_en_ina (0)(1),
clk=>clk,
Q => A_output(0)(1));
Bu2: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(3) ,
w_en => w_en_inb (0)(1),
clk=>clk,
Q => B_output(0)(1));
store_Op2: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(0)(1) ,
w_en => w_en_in (0)(1),
clk=>clk,
Q => operation(0)(1));
CU2: entity work.CU(Behavioral)
port map(
opr1 =>A_output(0)(1),
opr2 =>B_output(0)(1),
Sel => operation(0)(1),
res => R_output(0)(1));
Yu2: entity work.storageUnit(Behavioral)
port map(
D => R_output(0)(1) ,
w_en => w_en_out(0)(1),
clk=>clk,
Q => Y_output(0)(1));
```

```
--CU(0,2)
Mu5: entity work.Mux_4x1(Behavioral)
port map(
a1 => input1(2),
a2 => Y_output(0)(0),
a3 => Y_output(3)(2),
a4 => gnd, S => mux_sel(0)(4),
cu_Output => mux_output(0)(4));
Mu6: entity work.Mux_4x1(Behavioral)
port map(
a1 => input2(2),
a2 => Y_output(0)(0),
a3 => Y_output(3)(2),
a4 => gnd,
S => mux_sel(0)(5),
cu_Output => mux_output(0)(5));
Au3: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(4) ,
w_en => w_en_ina (0)(2),
clk=>clk,
Q => A_output(0)(2));
Bu3: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(5) ,
w_en => w_en_inb(0)(2),
clk=>clk,
Q => B_output(0)(2));
store_Op3: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(0)(2) ,
w_en => w_en_in (0)(2),
clk=>clk,
Q => operation(0)(2));
CU3: entity work.CU(Behavioral)
port map(
```

```
opr1 =>A_output(0)(2),
opr2 =>B_output(0)(2),
Sel => operation(0)(2),
res => R_output(0)(2));
Yu3: entity work.storageUnit(Behavioral)
port map(
D => R_output(0)(2) ,
w_en => w_en_out(0)(2),
clk=>clk,
Q => Y_output(0)(2));

--CU(0,3)
Mu7: entity work.Mux_4x1(Behavioral)
port map(
a1 => input1(3),
a2 => Y_output(0)(0),
a3 => Y_output(3)(3),
a4 => gnd,
S => mux_sel(0)(6),
cu_Output => mux_output(0)(6));
Mu8: entity work.Mux_4x1(Behavioral)
port map(
a1 => input2(3),
a2 => Y_output(0)(0),
a3 => Y_output(3)(3),
a4 => gnd,
S => mux_sel(0)(7),
cu_Output => mux_output(0)(7));
Au4: entity work.storageUnit(Behavioral)
port map(D => mux_output(0)(6) ,
w_en => w_en_ina (0)(3),
clk=>clk,
Q => A_output(0)(3));
Bu4: entity work.storageUnit(Behavioral)
port map(
D => mux_output(0)(7) ,
w_en => w_en_inb (0)(3),
```

```
clk=>clk,
Q => B_output(0)(3));
store_Op4: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(0)(3) ,
w_en => w_en_in (0)(3),
clk=>clk,
Q => operation(0)(3));
CU4: entity work.CU(Behavioral)
port map(
opr1 =>A_output(0)(3),
opr2 =>B_output(0)(3),
Sel => operation(0)(3),
res => R_output(0)(3));
Yu4: entity work.storageUnit(Behavioral)
port map(
D => R_output(0)(3) ,
w_en => w_en_out (0)(3),
clk=>clk,
Q => Y_output(0)(3));

--CU(1,0)
Mu9: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(0),
a2 => gnd,
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(0),
cu_Output => mux_output(1)(0));
Mu10: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(0),
a2 => gnd,
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(1),
```

```
cu_Output => mux_output(1)(1));
Au5: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(0) ,
w_en => w_en_ina (1)(0),
clk=>clk,
Q => A_output(1)(0));
Bu5: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(1) ,
w_en => w_en_inb (1)(0),
clk=>clk,
Q => B_output(1)(0));
store_Op5: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(1)(0) ,
w_en => w_en_in (1)(0),
clk=>clk,
Q => operation(1)(0));
CU5: entity work.CU(Behavioral)
port map(
opr1 =>A_output(1)(0),
opr2 =>B_output(1)(0),
Sel => operation(1)(0),
res => R_output(1)(0));
Yu5: entity work.storageUnit(Behavioral)
port map(
D => R_output(1)(0) ,
w_en => w_en_out (1)(0),
clk=>clk,
Q => Y_output(1)(0));

--CU(1,1)
Mu11: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(1),
a2 => Y_output(1)(0),
```

```
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(2),
cu_Output => mux_output(1)(2));
Mu12: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(1),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(3),
cu_Output => mux_output(1)(3));
Au6: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(2) ,
w_en => w_en_ina (1)(1),
clk=>clk,
Q => A_output(1)(1));
Bu6: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(3) ,
w_en => w_en_inb (1)(1),
clk=>clk,
Q => B_output(1)(1));
store_Op6: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(1)(1) ,
w_en => w_en_in (1)(1),
clk=>clk,
Q => operation(1)(1));
CU6: entity work.CU(Behavioral)
port map(
opr1 =>A_output(1)(1),
opr2 =>B_output(1)(1),
Sel => operation(1)(1),
res => R_output(1)(1));
Yu6: entity work.storageUnit(Behavioral)
```

```
port map(
D => R_output(1)(1) ,
w_en => w_en_out (1)(1),
clk=>clk,
Q => Y_output(1)(1));

--CU(1,2)
Mu13: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(2),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(4),
cu_Output => mux_output(1)(4));
Mu14: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(2),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(5),
cu_Output => mux_output(1)(5));
Au7: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(4) ,
w_en => w_en_ina (1)(2),
clk=>clk,
Q => A_output(1)(2));
Bu7: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(5) ,
w_en => w_en_inb (1)(2),
clk=>clk,
Q => B_output(1)(2));
store_Op7: entity work.storageUnit_Operation(Behavioral)
port map(
```

```
D => Sel(1)(2) ,
w_en => w_en_in (1)(2),
clk=>clk,
Q => operation(1)(2));
CU7: entity work.CU(Behavioral)
port map(
opr1 =>A_output(1)(2),
opr2 =>B_output(1)(2),
Sel => operation(1)(2),
res => R_output(1)(2));
Yu7: entity work.storageUnit(Behavioral)
port map(
D => R_output(1)(2) ,
w_en => w_en_out (1)(2),
clk=>clk,
Q => Y_output(1)(2));

--CU(1,3)
Mu15: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(3),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(6),
cu_Output => mux_output(1)(6));
Mu16: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(3),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(1)(7),
cu_Output => mux_output(1)(7));
Au8: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(6) ,
```

```
w_en => w_en_ina (1)(3),
clk=>clk,
Q => A_output(1)(3));
Bu8: entity work.storageUnit(Behavioral)
port map(
D => mux_output(1)(7) ,
w_en => w_en_inb (1)(3),
clk=>clk,
Q => B_output(1)(3));
store_Op8: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(1)(3) ,
w_en => w_en_in (1)(3),
clk=>clk,
Q => operation(1)(3));
CU8: entity work.CU(Behavioral)
port map(
opr1 =>A_output(1)(3),
opr2 =>B_output(1)(3),
Sel => operation(1)(3),
res => R_output(1)(3));
Yu8: entity work.storageUnit(Behavioral)
port map(
D => R_output(1)(3) ,
w_en => w_en_out (1)(3),
clk=>clk,
Q => Y_output(1)(3));

--CU(2,0)
Mu17: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(0),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(2)(0),
cu_Output => mux_output(2)(0));
```

```
Mu18: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(0),
a2 => Y_output(1)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(2)(1),
cu_Output => mux_output(2)(1));
Au9: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(0) ,
w_en => w_en_ina (2)(0),
clk=>clk,
Q => A_output(2)(0));
Bu9: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(1) ,
w_en => w_en_inb (2)(0),
clk=>clk,
Q => B_output(2)(0));
store_Op9: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(2)(0) ,
w_en => w_en_in (2)(0),
clk=>clk,
Q => operation(2)(0));
CU9: entity work.CU(Behavioral)
port map(
opr1 =>A_output(2)(0),
opr2 =>B_output(2)(0),
Sel => operation(2)(0),
res => R_output(2)(0));
Yu9: entity work.storageUnit(Behavioral)
port map(
D => R_output(2)(0) ,
w_en => w_en_out (2)(0),
clk=>clk,
```

```
Q => Y_output(2)(0));

--CU(2,1)
Mu19: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(1),
a2 => Y_output(1)(1),
a3 => Y_output(2)(0),
a4 => gnd,
S => mux_sel(2)(2),
cu_Output => mux_output(2)(2));
Mu20: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(1),
a2 => Y_output(1)(1),
a3 => Y_output(2)(0),
a4 => gnd,
S => mux_sel(2)(3),
cu_Output => mux_output(2)(3));
Au10: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(2) ,
w_en => w_en_ina (2)(1),
clk=>clk,
Q => A_output(2)(1));
Bu10: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(3) ,
w_en => w_en_inb (2)(1),
clk=>clk,
Q => B_output(2)(1));
store_Op10: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(2)(1) ,
w_en => w_en_in (2)(1),
clk=>clk,
Q => operation(2)(1));
```

```
CU10: entity work.CU(Behavioral)
port map(
opr1 =>A_output(2)(1),
opr2 =>B_output(2)(1),
Sel => operation(2)(1),
res => R_output(2)(1));
Yu10: entity work.storageUnit(Behavioral)
port map(
D => R_output(2)(1) ,
w_en => w_en_out (2)(1),
clk=>clk,
Q => Y_output(2)(1));

--CU(2,2)
Mu21: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(2),
a2 => Y_output(1)(2),
a3 => Y_output(2)(0),
a4 => gnd,
S => mux_sel(2)(4),
cu_Output => mux_output(2)(4));
Mu22: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(2),
a2 => Y_output(1)(2),
a3 => Y_output(2)(0),
a4 => gnd,
S => mux_sel(2)(5),
cu_Output => mux_output(2)(5));
Au11: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(4) ,
w_en => w_en_ina (2)(2),
clk=>clk,
Q => A_output(2)(2));
Bu11: entity work.storageUnit(Behavioral)
```

```
port map(
D => mux_output(2)(5) ,
w_en => w_en_inb (2)(2),
clk=>clk,
Q => B_output(2)(2));
store_Op11: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(2)(2) ,
w_en => w_en_in (2)(2),
clk=>clk,
Q => operation(2)(2));
CU11: entity work.CU(Behavioral)
port map(
opr1 =>A_output(2)(2),
opr2 =>B_output(2)(2),
Sel => operation(2)(2),
res => R_output(2)(2));
Yu11: entity work.storageUnit(Behavioral)
port map(
D => R_output(2)(2) ,
w_en => w_en_out (2)(2),
clk=>clk,
Q => Y_output(2)(2));

--CU(2,3)
Mu23: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(3),
a2 => Y_output(1)(3),
a3 => Y_output(2)(0),
a4 => gnd, S => mux_sel(2)(6),
cu_Output => mux_output(2)(6));
Mu24: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(0)(3),
a2 => Y_output(1)(3),
a3 => Y_output(2)(0),
```

```
a4 => gnd,
S => mux_sel(2)(7),
cu_Output => mux_output(2)(7));
Au12: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(6) ,
w_en => w_en_ina (2)(3),
clk=>clk,
Q => A_output(2)(3));
Bu12: entity work.storageUnit(Behavioral)
port map(
D => mux_output(2)(7) ,
w_en => w_en_inb (2)(3),
clk=>clk,
Q => B_output(2)(3));
store_Op12: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(2)(3) ,
w_en => w_en_in (2)(3),
clk=>clk,
Q => operation(2)(3));
CU12: entity work.CU(Behavioral)
port map(
opr1 =>A_output(2)(3),
opr2 =>B_output(2)(3),
Sel => operation(2)(3),
res => R_output(2)(3));
Yu12: entity work.storageUnit(Behavioral)
port map(
D => R_output(2)(3) ,
w_en => w_en_out (2)(3),
clk=>clk,
Q => Y_output(2)(3));

--CU(3,0)
Mu25: entity work.Mux_4x1(Behavioral)
port map(
```

```
a1 => Y_output(1)(0),
a2 => Y_output(2)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(3)(0),
cu_Output => mux_output(3)(0));
Mu26: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(0),
a2 => Y_output(2)(0),
a3 => gnd,
a4 => gnd,
S => mux_sel(3)(1),
cu_Output => mux_output(3)(1));
Au13: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(0) ,
w_en => w_en_ina (3)(0),
clk=>clk,
Q => A_output(3)(0));
Bu13: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(1) ,
w_en => w_en_inb (3)(0),
clk=>clk,
Q => B_output(3)(0));
store_Op13: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(3)(0) ,
w_en => w_en_in (3)(0),
clk=>clk,
Q => operation(3)(0));
CU13: entity work.CU(Behavioral)
port map(
opr1 =>A_output(3)(0),
opr2 =>B_output(3)(0),
Sel => operation(3)(0),
```

```
res => R_output(3)(0));
Yu13: entity work.storageUnit(Behavioral)
port map(
D => R_output(3)(0) ,
w_en => w_en_out (3)(0),
clk=>clk,
Q => Y_output(3)(0));

--CU(3,1)
Mu27: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(1),
a2 => Y_output(2)(1),
a3 => Y_output(3)(0),
a4 => gnd,
S => mux_sel(3)(2),
cu_Output => mux_output(3)(2));
Mu28: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(1),
a2 => Y_output(2)(1),
a3 => Y_output(3)(0),
a4 => gnd,
S => mux_sel(3)(3),
cu_Output => mux_output(3)(3));
Au14: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(2) ,
w_en => w_en_ina (3)(1),
clk=>clk,
Q => A_output(3)(1));
Bu14: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(3) ,
w_en => w_en_inb (3)(1),
clk=>clk,
Q => B_output(3)(1));
```

```
store_Op14: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(3)(1) ,
w_en => w_en_in (3)(1),
clk=>clk,
Q => operation(3)(1));
CU14: entity work.CU(Behavioral)
port map(
opr1 =>A_output(3)(1),
opr2 =>B_output(3)(1),
Sel => operation(3)(1),
res => R_output(3)(1));
Yu14: entity work.storageUnit(Behavioral)
port map(
D => R_output(3)(1) ,
w_en => w_en_out (3)(1),
clk=>clk,
Q => Y_output(3)(1));

--CU(3,2)
Mu29: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(2),
a2 => Y_output(2)(2),
a3 => Y_output(3)(0),
a4 => gnd,
S => mux_sel(3)(4),
cu_Output => mux_output(3)(4));
Mu30: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(2),
a2 => Y_output(2)(2),
a3 => Y_output(3)(0),
a4 => gnd,
S => mux_sel(3)(5),
cu_Output => mux_output(3)(5));
Au15: entity work.storageUnit(Behavioral)
```

```
port map(
D => mux_output(3)(4) ,
w_en => w_en_ina (3)(2),
clk=>clk,
Q => A_output(3)(2));
Bu15: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(5) ,
w_en => w_en_inb (3)(2),
clk=>clk,
Q => B_output(3)(2));
store_Op15: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(3)(2) ,
w_en => w_en_in (3)(2),
clk=>clk,
Q => operation(3)(2));
CU15: entity work.CU(Behavioral)
port map(
opr1 =>A_output(3)(2),
opr2 =>B_output(3)(2),
Sel => operation(3)(2),
res => R_output(3)(2));
Yu15: entity work.storageUnit(Behavioral)
port map(
D => R_output(3)(2) ,
w_en => w_en_out (3)(2),
clk=>clk,
Q => Y_output(3)(2));

--CU(3,3)
Mu31: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(3),
a2 => Y_output(2)(3),
a3 => Y_output(3)(0),
a4 => gnd,
```

```
S => mux_sel(3)(6),
cu_Output => mux_output(3)(6));
Mu32: entity work.Mux_4x1(Behavioral)
port map(
a1 => Y_output(1)(3),
a2 => Y_output(2)(3),
a3 => Y_output(3)(0),
a4 => gnd,
S => mux_sel(3)(7),
cu_Output => mux_output(3)(7));
Au16: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(6) ,
w_en => w_en_ina (3)(3),
clk=>clk,
Q => A_output(3)(3));
Bu16: entity work.storageUnit(Behavioral)
port map(
D => mux_output(3)(7) ,
w_en => w_en_inb (3)(3),
clk=>clk,
Q => B_output(3)(3));
store_Op16: entity work.storageUnit_Operation(Behavioral)
port map(
D => Sel(3)(3) ,
w_en => w_en_in (3)(3),
clk=>clk,
Q => operation(3)(3));
CU16: entity work.CU(Behavioral)
port map(
opr1 =>A_output(3)(3),
opr2 =>B_output(3)(3),
Sel => operation(3)(3),
res => R_output(3)(3));
Yu16: entity work.storageUnit(Behavioral)
port map(
D => R_output(3)(3) ,
```

```
w_en => w_en_out (3)(3),
clk=>clk,
Q => Y_output(3)(3));

cu_Output <= Y_output;
final_output <= Y_output(1);
end Behavioral;
```