Hardware Modeling [VU] (191.011)
– WS25 –

From Circuits to Code and Back

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

Modified: 2025-12-16, 16:04 (f8a58e9)

Introduction

Lecture Goal

Develop an intuition for how VHDL code maps to hardware and the associated circuit complexity.

Introduction

Lecture Goal

Develop an intuition for how VHDL code maps to hardware and the associated circuit complexity.

- Two examples
  - Circuit diagram → Derive VHDL code
  - VHDL code → Derive circuit diagram

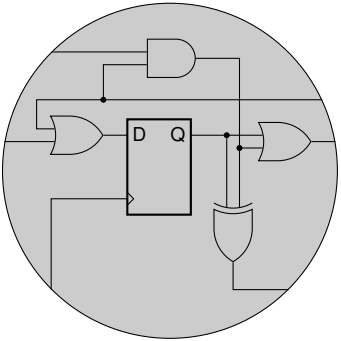
Introduction

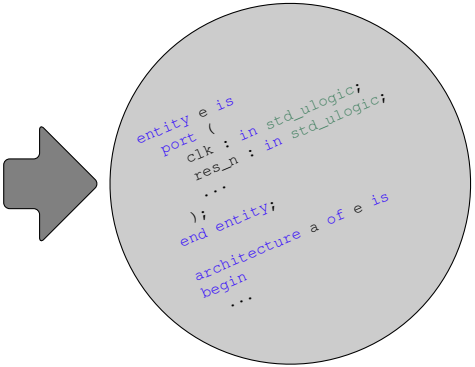
Lecture Goal

Develop an intuition for how VHDL code maps to hardware and the associated circuit complexity.

- Two examples
  - Circuit diagram → Derive VHDL code
  - VHDL code → Derive circuit diagram
- Important VHDL concepts used in the lecture
  - Sequential circuit elements
  - Behavioral modeling
  - Array-types and arithmetic functions

Example 1: Circuit → VHDL Code

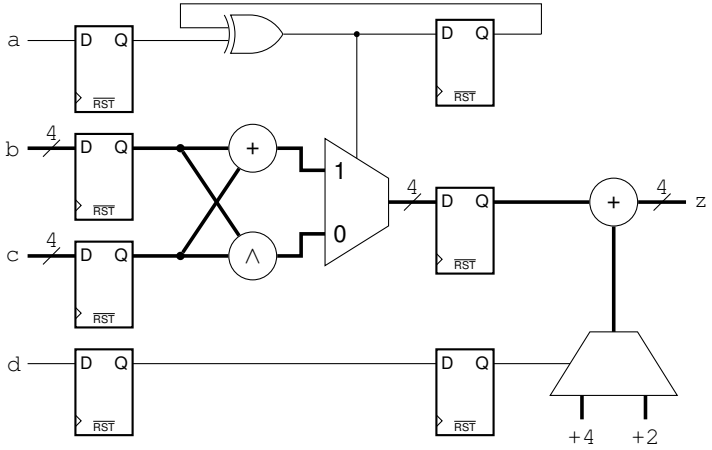




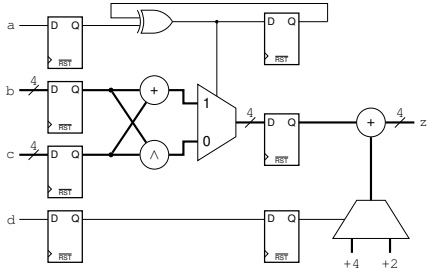
```
entity e is
  port (
    clk : in std_ulogic;
    res_n : in std_ulogic;
    ...
  );
end entity;

architecture a of e is
begin
  ...
```

Example 1: Circuit Diagram

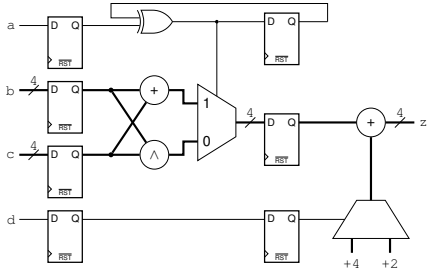


Example 1 - Entity



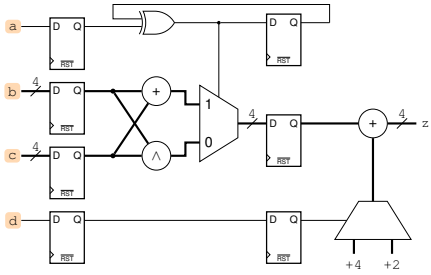

```
1  entity circuit2code is
2    port (
3      clk : in std_ulogic;
4      res_n : in std_ulogic;
5      a : in std_ulogic;
6      b : in std_ulogic_vector(3 downto 0);
7      c : in std_ulogic_vector(3 downto 0);
8      d : in std_ulogic;
9      z : out std_ulogic_vector(3 downto 0)
10   );
11 end entity;
```

Example 1 - Entity



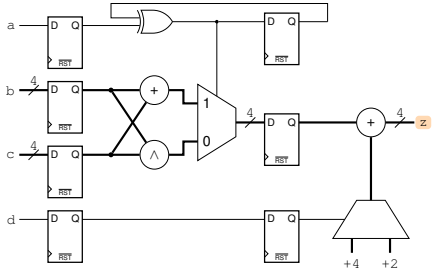

```
1  entity circuit2code is
2    port (
3      clk : in std_ulogic;
4      res_n : in std_ulogic;
5      a : in std_ulogic;
6      b : in std_ulogic_vector(3 downto 0);
7      c : in std_ulogic_vector(3 downto 0);
8      d : in std_ulogic;
9      z : out std_ulogic_vector(3 downto 0)
10   );
11 end entity;
```

Example 1 - Entity



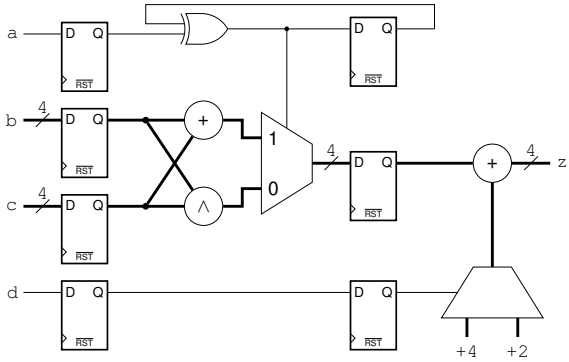

```
1  entity circuit2code is
2    port (
3      clk : in std_ulogic;
4      res_n : in std_ulogic;
5      a : in std_ulogic;
6      b : in std_ulogic_vector(3 downto 0);
7      c : in std_ulogic_vector(3 downto 0);
8      d : in std_ulogic;
9      z : out std_ulogic_vector(3 downto 0)
10   );
11 end entity;
```

Example 1 - Entity

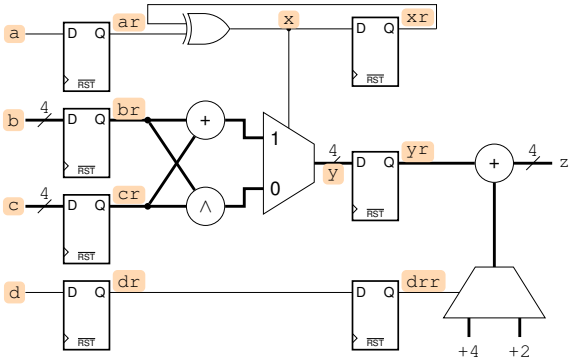



```
1  entity circuit2code is
2    port (
3      clk : in std_ulogic;
4      res_n : in std_ulogic;
5      a : in std_ulogic;
6      b : in std_ulogic_vector(3 downto 0);
7      c : in std_ulogic_vector(3 downto 0);
8      d : in std_ulogic;
9      z : out std_ulogic_vector(3 downto 0)
10   );
11 end entity;
```

Example 1 - Architecture



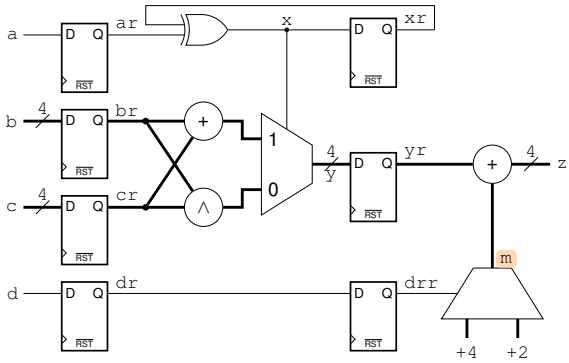
Example 1 - Architecture



Circuit Preparation

Label **all** register inputs and outputs and other important signals!

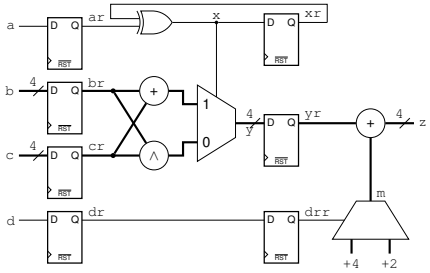
Example 1 - Architecture



Circuit Preparation

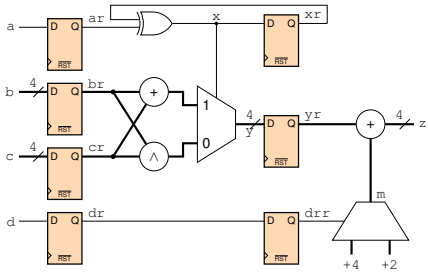
Label **all** register inputs and outputs and other important signals!

Architecture - Signal Declarations



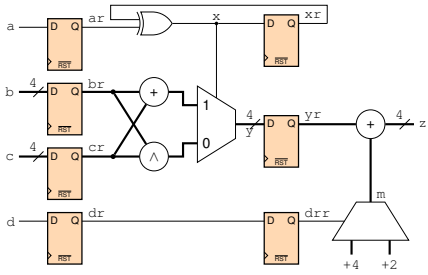

```
1 architecture arch of circuit2code is
2   signal br, cr, y, yr :
3     std_ulogic_vector(3 downto 0);
4   signal ar, x, xr, dr, drr : std_ulogic;
5 begin
```

Architecture - Registers



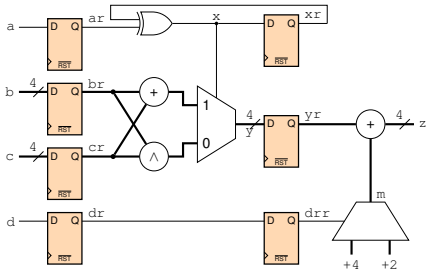

```
6   process (clk, res_n)
7   begin
8     if res_n = '0' then
9       ar <= '0'; dr <= '0';
10      xr <= '0'; drr <= '0';
11      br <= (others => '0');
12      cr <= (others => '0');
13      yr <= (others => '0');
14    elsif rising_edge(clk) then
15      ar <= a;
16      br <= b;
17      cr <= c;
18      dr <= d;
19      xr <= x;
20      yr <= y;
21      drr <= dr;
22    end if;
23  end process;
```

Architecture - Registers



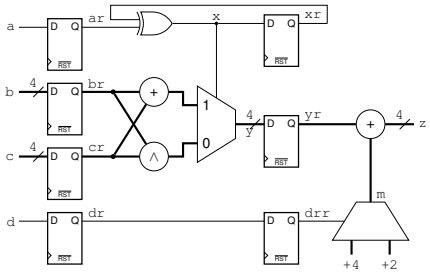

```
6   process (clk, res_n)
7   begin
8     if res_n = '0' then
9       ar <= '0'; dr <= '0';
10      xr <= '0'; drr <= '0';
11      br <= (others => '0');
12      cr <= (others => '0');
13      yr <= (others => '0');
14    elsif rising_edge(clk) then
15      ar <= a;
16      br <= b;
17      cr <= c;
18      dr <= d;
19      xr <= x;
20      yr <= y;
21      drr <= dr;
22    end if;
23  end process;
```

Architecture - Registers



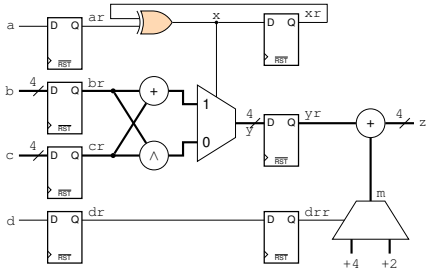

```
6   process (clk, res_n)
7   begin
8     if res_n = '0' then
9       ar <= '0'; dr <= '0';
10      xr <= '0'; drr <= '0';
11      br <= (others => '0');
12      cr <= (others => '0');
13      yr <= (others => '0');
14    elsif rising_edge(clk) then
15      ar <= a;
16      br <= b;
17      cr <= c;
18      dr <= d;
19      xr <= x;
20      yr <= y;
21      drr <= dr;
22    end if;
23  end process;
```

Architecture - Combinational Logic I



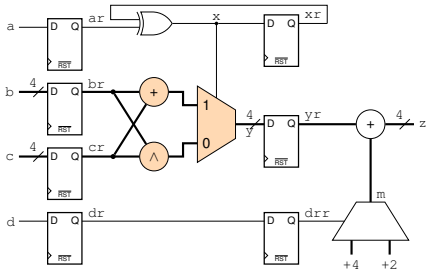

```
24  x <= ar xor xr;
25
26  process (all)
27  begin
28    if x = '1' then
29      y <= std_ulogic_vector(
30        unsigned(br) +
31        unsigned(cr)
32      );
33    else
34      y <= br and cr;
35    end if;
36  end process;
```

Architecture - Combinational Logic I



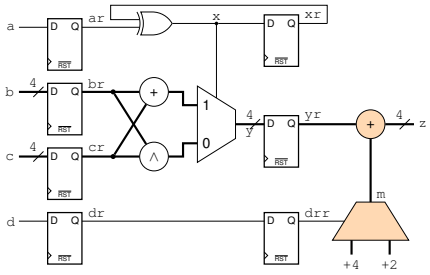

```
24  x <= ar xor xr;
25
26  process (all)
27  begin
28    if x = '1' then
29      y <= std_ulogic_vector(
30        unsigned(br) +
31        unsigned(cr)
32      );
33    else
34      y <= br and cr;
35    end if;
36  end process;
```

Architecture - Combinational Logic I



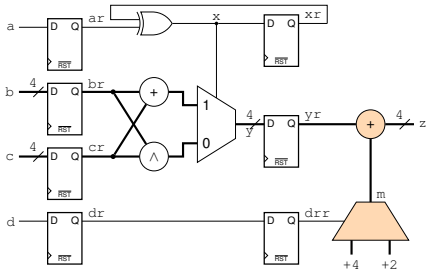

```
24  x <= ar xor xr;
25
26  process (all)
27  begin
28    if x = '1' then
29      y <= std_ulogic_vector(
30        unsigned(br) +
31        unsigned(cr)
32      );
33    else
34      y <= br and cr;
35    end if;
36  end process;
```

Architecture - Combinational Logic II



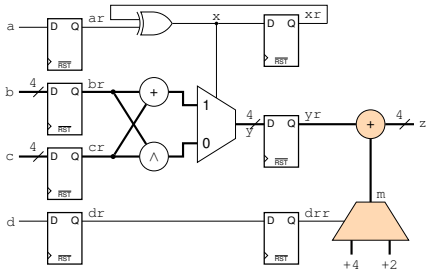

```
37   process (all)
38     variable m : integer;
39   begin
40     if drr = '1' then
41       m := 2;
42     else
43       m := 4;
44     end if;
45
46     z <= std_ulogic_vector(
47       unsigned(yr) + m
48     );
49   end process;
50 end architecture;
```

Architecture - Combinational Logic II



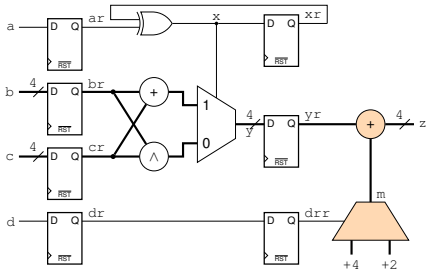

```
37   process (all)
38     variable m : integer;
39   begin
40     if drr = '1' then
41       m := 2;
42     else
43       m := 4;
44     end if;
45
46     z <= std_ulogic_vector(
47       unsigned(yr) + m
48     );
49   end process;
50 end architecture;
```

Architecture - Combinational Logic II




```
37   process (all)
38     variable m : integer;
39   begin
40     if drr = '1' then
41       m := 2;
42     else
43       m := 4;
44     end if;
45
46     z <= std_ulogic_vector(
47       unsigned(yr) + m
48     );
49   end process;
50 end architecture;
```

Architecture - Combinational Logic II




```
37   process (all)
38     variable m : integer;
39   begin
40     if drr = '1' then
41       m := 2;
42     else
43       m := 4;
44     end if;
45
46     z <= std_ulogic_vector(
47       unsigned(yr) + m
48     );
49   end process;
50 end architecture;
```

Restructured Architecture


```
1   process (clk, res_n)
2   begin
3     if res_n = '0' then
4       ar <= '0'; dr <= '0';
5       xr <= '0'; drr <= '0';
6       br <= (others => '0');
7       cr <= (others => '0');
8       yr <= (others => '0');
9     elsif rising_edge(clk) then
10      ar <= a;
11      br <= b;
12      cr <= c;
13      dr <= d;
14      xr <= x;
15      yr <= y;
16      drr <= dr;
17    end if;
18  end process;
```

```
19  x <= ar xor xr;
20
21  process (all)
22  begin
23    if x = '1' then
24      y <= std_ulogic_vector(
25        unsigned(br) +
26        unsigned(cr)
27      );
28    else
29      y <= br and cr;
30    end if;
31  end process;
```

Restructured Architecture

```
1  process (clk, res_n)
2  begin
3    if res_n = '0' then
4      ar <= '0'; dr <= '0';
5      xr <= '0'; drr <= '0';
6      br <= (others => '0');
7      cr <= (others => '0');
8      yr <= (others => '0');
9    elsif rising_edge(clk) then
10     ar <= a;
11     br <= b;
12     cr <= c;
13     dr <= d;
14     xr <= x;
15     yr <= y;
16     drr <= dr;
17   end if;
18 end process;
```

```
19 x <= ar xor xr;
20
21 process (all)
22 begin
23   if x = '1' then
24     y <= std_ulogic_vector(
25       unsigned(br) +
26       unsigned(cr)
27     );
28   else
29     y <= br and cr;
30   end if;
31 end process;
```

Restructured Architecture


```
1   process (clk, res_n)
2   begin
3     if res_n = '0' then
4       ar <= '0'; dr <= '0';
5       xr <= '0'; drr <= '0';
6       br <= (others => '0');
7       cr <= (others => '0');
8       yr <= (others => '0');
9     elsif rising_edge(clk) then
10      ar <= a;
11      br <= b;
12      cr <= c;
13      dr <= d;
14      xr <= x;
15      yr <= y;
16      drr <= dr;
17    end if;
18  end process;
```

```
19  x <= ar xor xr;
20
21  process (all)
22  begin
23    if x = '1' then
24      y <= std_ulogic_vector(
25        unsigned(br) +
26        unsigned(cr)
27      );
28    else
29      y <= br and cr;
30    end if;
31  end process;
```

Restructured Architecture


```
1   process (clk, res_n)
2   begin
3     if res_n = '1' then
4       ar <= '0'; xr <= '0';
5       dr <= '0'; drr <= '0';
6       br <= (others => '0');
7       cr <= (others => '0');
8       yr <= (others => '0');
9     elsif rising_edge(clk) then
10      ar <= a;
11      br <= b;
12      cr <= c;
13      dr <= d;
14      xr <= x;
15      if x = '1' then
16        yr <= std_ulogic_vector(
17          unsigned(br) +
18          unsigned(cr)
19        );
20      else
21        yr <= br and cr;
22      end if;
23
24      drr <= dr;
25    end if;
26  end process;
27
28  x <= ar xor xr;
```

Restructured Architecture


```
1   process (clk, res_n)
2   begin
3     if res_n = '1' then
4       ar <= '0'; xr <= '0';
5       dr <= '0'; drr <= '0';
6       br <= (others => '0');
7       cr <= (others => '0');
8       yr <= (others => '0');
9     elsif rising_edge(clk) then
10      ar <= a;
11      br <= b;
12      cr <= c;
13      dr <= d;
14      xr <= x;
15      if x = '1' then
16        yr <= std_ulogic_vector(
17          unsigned(br) +
18          unsigned(cr)
19        );
20      else
21        yr <= br and cr;
22      end if;
23
24      drr <= dr;
25    end if;
26  end process;
27
28  x <= ar xor xr;
```

Restructured Architecture


```
1   process (clk, res_n)
2   begin
3     if res_n = '1' then
4       ar <= '0'; xr <= '0';
5       dr <= '0'; drr <= '0';
6       br <= (others => '0');
7       cr <= (others => '0');
8       yr <= (others => '0');
9     elsif rising_edge(clk) then
10      ar <= a;
11      br <= b;
12      cr <= c;
13      dr <= d;
14      xr <= x;
15      if x = '1' then
16        yr <= std_ulogic_vector(
17          unsigned(br) +
18          unsigned(cr)
19        );
20      else
21        yr <= br and cr;
22      end if;
23
24      drr <= dr;
25    end if;
26  end process;
27
28  x <= ar xor xr;
```

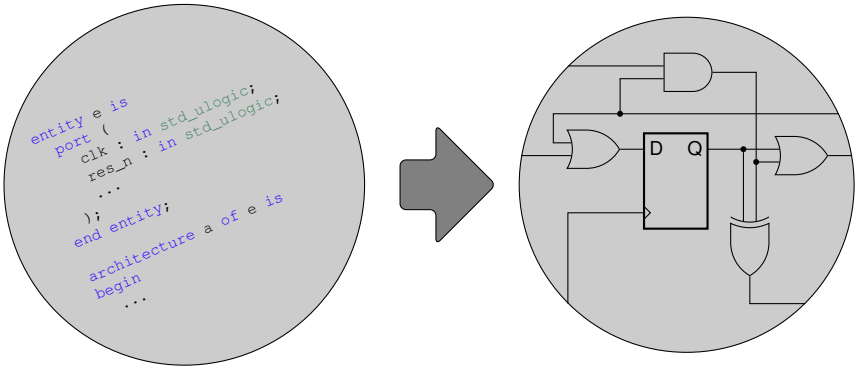
Restructured Architecture


```
1  process (clk, res_n)
2  begin
3    if res_n = '1' then
4      ar <= '0'; xr <= '0';
5      dr <= '0'; drr <= '0';
6      br <= (others => '0');
7      cr <= (others => '0');
8      yr <= (others => '0');
9    elsif rising_edge(clk) then
10     ar <= a;
11     br <= b;
12     cr <= c;
13     dr <= d;
14     xr <= ar xor xr;
15     if  (ar xor xr) = '1' then
16       yr <= std_ulogic_vector(
17         unsigned(br) +
18         unsigned(cr)
19       );
20     else
21       yr <= br and cr;
22     end if;
23
24     drr <= dr;
25   end if;
26 end process;
27
28 process (all)
29   variable m : integer;
30 begin
31   if drr = '1' then
32     m := 2;
33   else
34     m := 4;
35   end if;
36   z <= std_ulogic_vector(
37     unsigned(yr) + m);
38 end process;
```

Restructured Architecture


```
1  process (clk, res_n)
2  begin
3    if res_n = '1' then
4      ar <= '0'; xr <= '0';
5      dr <= '0'; drr <= '0';
6      br <= (others => '0');
7      cr <= (others => '0');
8      yr <= (others => '0');
9    elsif rising_edge(clk) then
10     ar <= a;
11     br <= b;
12     cr <= c;
13     dr <= d;
14     xr <= ar xor xr;
15     if  (ar xor xr) = '1' then
16       yr <= std_ulogic_vector(
17         unsigned(br) +
18         unsigned(cr)
19       );
20     else
21       yr <= br and cr;
22     end if;
23
24     drr <= dr;
25   end if;
26 end process;
27
28 process (all)
29   variable m : integer;
30 begin
31   if drr = '1' then
32     m := 2;
33   else
34     m := 4;
35   end if;
36   z <= std_ulogic_vector(
37     unsigned(yr) + m);
38 end process;
```

Example 2: VHDL Code → Circuit



Example 2: Entity


```
1  entity code2circuit is
2    generic (
3      N, M : positive
4    );
5    port (
6      clk : in std_ulogic;
7      res_n : in std_ulogic;
8      a : in std_ulogic;
9      b : in std_ulogic_vector(N-1 downto 0);
10     c : in std_ulogic_vector(M-1 downto 0);
11     d : in std_ulogic_vector(M-1 downto 0);
12     x : out std_ulogic;
13     y : out std_ulogic_vector(M-1 downto 0)
14   );
15 end entity;
```

Example 2: Entity


```
1  entity code2circuit is
2    generic (
3      N, M : positive
4    );
5    port (
6      clk : in std_ulogic;
7      res_n : in std_ulogic;
8      a : in std_ulogic;
9      b : in std_ulogic_vector(N-1 downto 0);
10     c : in std_ulogic_vector(M-1 downto 0);
11     d : in std_ulogic_vector(M-1 downto 0);
12     x : out std_ulogic;
13     y : out std_ulogic_vector(M-1 downto 0)
14   );
15 end entity;
```

Example 2: Entity


```
1  entity code2circuit is
2    generic (
3      N, M : positive
4    );
5    port (
6      clk : in std_ulogic;
7      res_n : in std_ulogic;
8      a : in std_ulogic;
9      b : in std_ulogic_vector(N-1 downto 0);
10     c : in std_ulogic_vector(M-1 downto 0);
11     d : in std_ulogic_vector(M-1 downto 0);
12     x : out std_ulogic;
13     y : out std_ulogic_vector(M-1 downto 0)
14   );
15 end entity;
```

Example 2: Entity


```
1  entity code2circuit is
2    generic (
3      N, M : positive
4    );
5    port (
6      clk : in std_ulogic;
7      res_n : in std_ulogic;
8      a : in std_ulogic;
9      b : in std_ulogic_vector(N-1 downto 0);
10     c : in std_ulogic_vector(M-1 downto 0);
11     d : in std_ulogic_vector(M-1 downto 0);
12     x : out std_ulogic;
13     y : out std_ulogic_vector(M-1 downto 0)
14   );
15 end entity;
```

Example 2: Architecture


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```

Example 2: Architecture


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
19
20   process(all)
21     variable temp : std_ulogic;
22   begin
23     temp := '0';
24     for i in b'range loop
25       temp := temp or b(i);
26     end loop;
27
28     y <= r2;
29
30     if temp and r0 then
31       y <= std_ulogic_vector(
32         to_unsigned(3, y'length)
33       );
34     end if;
35   end process;
36
37 end architecture;
```

Example 2: Architecture


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```

Example 2: Architecture


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```

Example 2: Architecture


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```

Example 2: Circuit - Registers

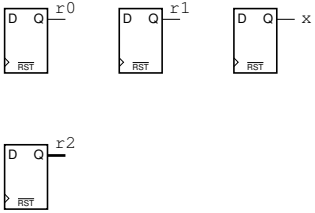

```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```

Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```

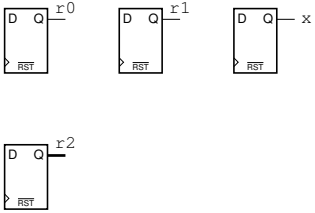
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



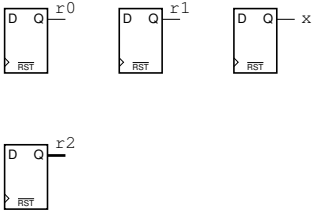
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



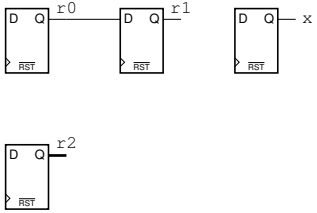
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



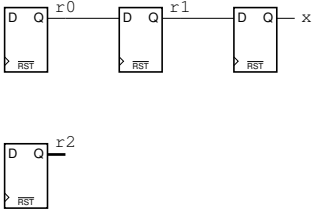
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



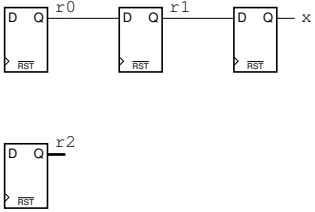
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



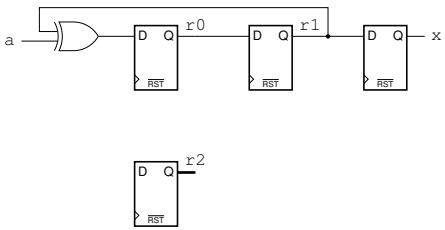
Example 2: Circuit - Registers

```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



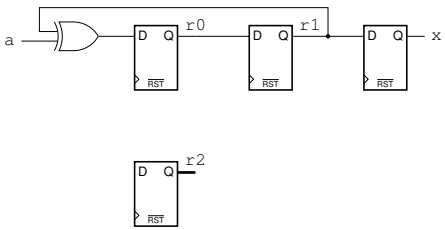
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



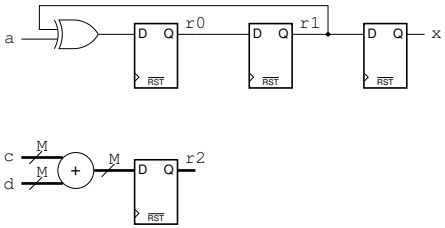
Example 2: Circuit - Registers


```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



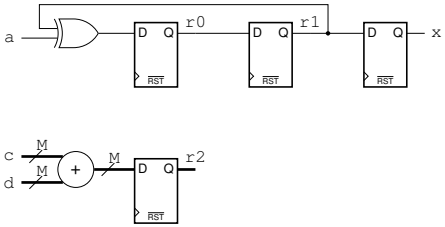
Example 2: Circuit - Registers

```
1  architecture arch of code2circuit is
2    signal r0, r1 : std_ulogic;
3    signal r2 : c'subtype;
4  begin
5    process (clk, res_n)
6    begin
7      if res_n = '0' then
8        r0 <= '0'; r1 <= '0'; x <= '0';
9        r2 <= (others => '0');
10     elsif rising_edge(clk) then
11       r0 <= a xor r1;
12       r1 <= r0;
13       x <= r1;
14       r2 <= std_ulogic_vector(
15         signed(c) + signed(d)
16       );
17     end if;
18   end process;
```



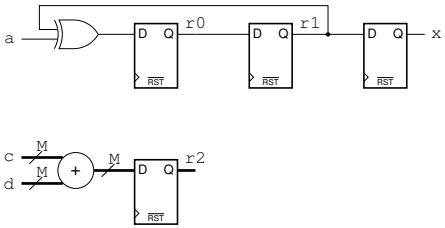
Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



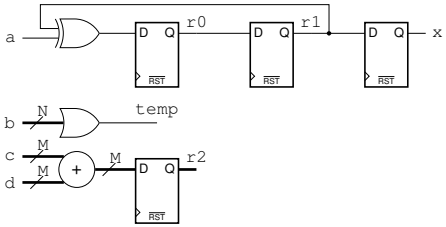
Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



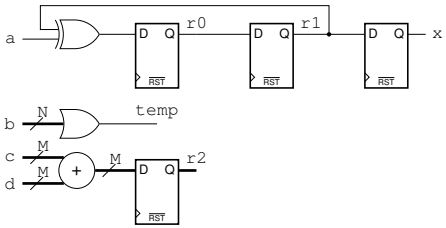
Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



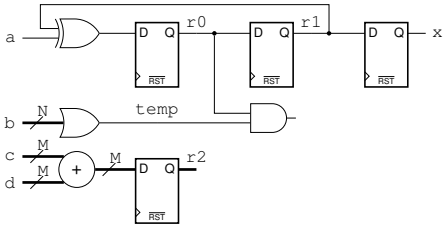
Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



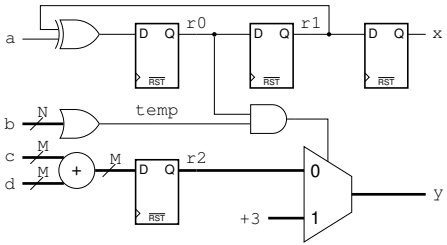
Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



Example 2: Circuit - Combinational Process


```
19   process(all)
20     variable temp : std_ulogic;
21   begin
22     temp := '0';
23     for i in b'range loop
24       temp := temp or b(i);
25     end loop;
26
27     y <= r2;
28
29     if temp and r0 then
30       y <= std_ulogic_vector(
31         to_unsigned(3, y'length)
32       );
33     end if;
34   end process;
35
36 end architecture;
```



Lecture Complete!