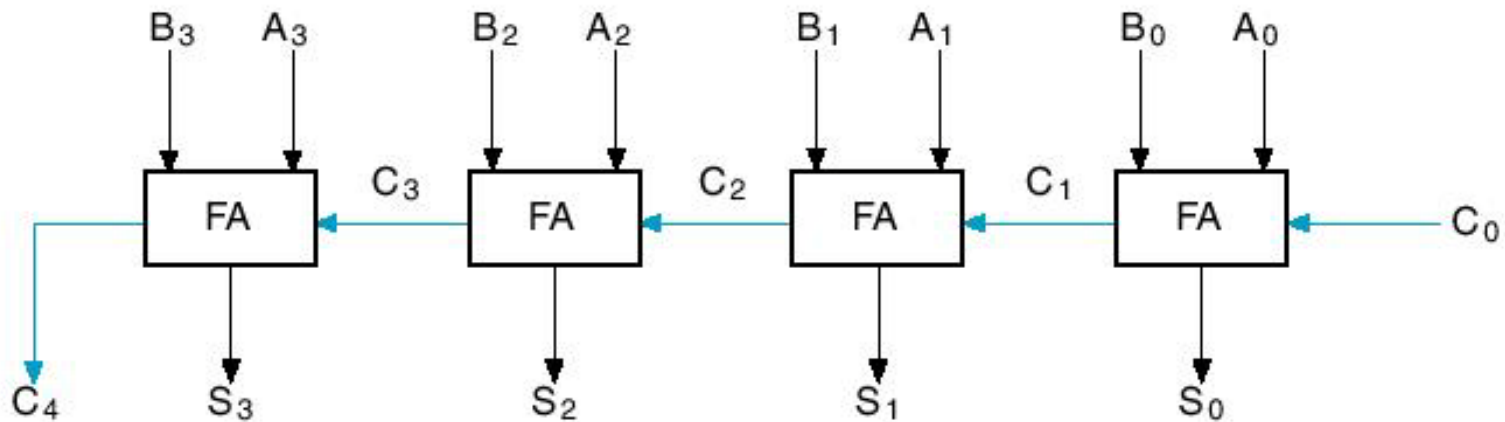
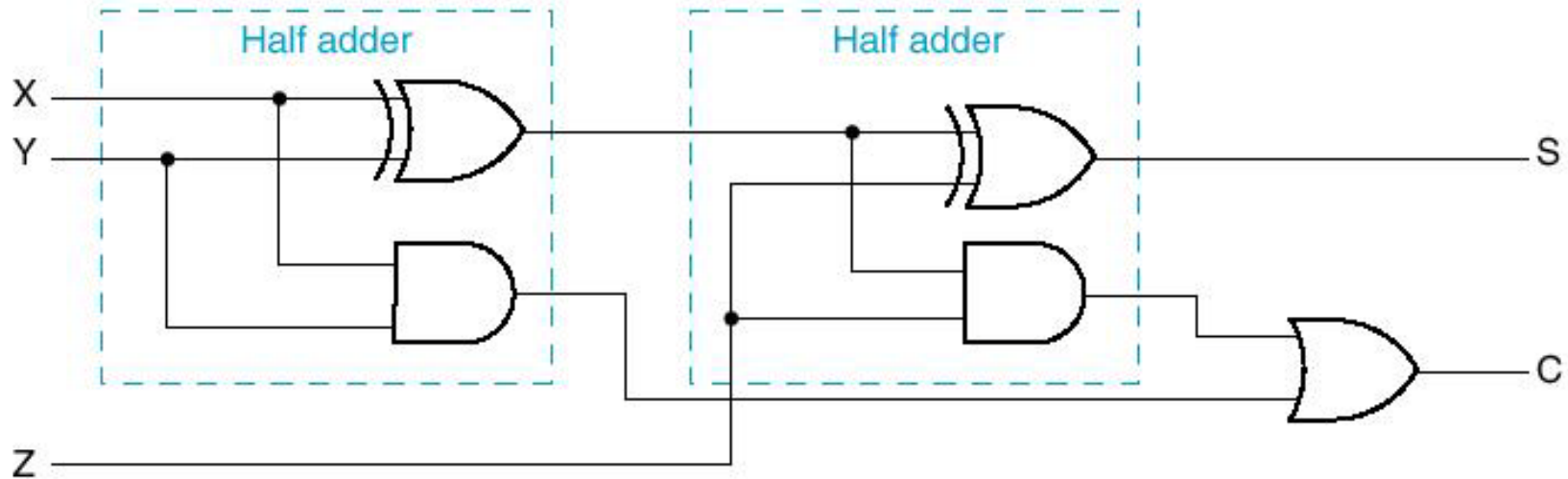


Hierarchical VHDL

4-Bit Ripple Carry Adder



CS2022 Hierarchical VHDL (Page 1)

4-Bit Ripple Carry Adder

```
-- 4-bit Adder: Hierarchical Dataflow/Structural
library ieee;
use ieee.std_logic_1164.all;
entity half_adder is
    port (x, y : in std_logic;
          s, c : out std_logic);
end half_adder;

architecture dataflow_3 of half_adder is
begin
    s <= x xor y;
    c <= x and y;
end dataflow_3;
```

4-Bit Ripple Carry Adder

```
library ieee;  
use ieee.std_logic_1164.all;  
entity full_adder is  
    port (x, y, z : in std_logic;  
          s, c : out std_logic);  
end full_adder;
```

Hierarchical VHDL (Page 3)

4-Bit Ripple Carry Adder

```
architecture struc_dataflow_3 of full_adder is
  component half_adder
    port(x, y : in std_logic;
          s, c : out std_logic);
  end component;
  signal hs, hc, tc: std_logic;
begin
  HA1: half_adder
    port map (x, y, hs, hc);
  HA2: half_adder
    port map (hs, z, s, tc);
  c <= tc or hc;
end struc_dataflow_3;
```

CS2022 Hierarchical VHDL (Page 4)

4-Bit Ripple Carry Adder

```
library ieee;  
use ieee.std_logic_1164.all;  
entity adder_4 is  
    port(B, A : in std_logic_vector(3 downto 0);  
          C0 : in std_logic;  
          S : out std_logic_vector(3 downto 0);  
          C4: out std_logic);  
end adder_4;
```

CS2022 Hierarchical VHDL (Page 5)

4-Bit Ripple Carry Adder

```
architecture structural_4 of adder_4 is
  component full_adder
    port(x, y, z : in std_logic;
         s, c : out std_logic);
  end component;
  signal C: std_logic_vector(3 downto 1);
begin
  Bit0: full_adder
    port map (B(0), A(0), C(0), S(0), C(1));
  Bit1: full_adder
    port map (B(1), A(1), C(1), S(1), C(2));
  Bit2: full_adder
    port map (B(2), A(2), C(2), S(2), C(3));
  Bit3: full_adder
    port map (B(3), A(3), C(3), S(3), C4);
end structural_4;
```

4-Bit Full Adder

Behavioral Description

-- 4-bit Adder: Behavioral Description

```
library ieee;
```

```
use ieee.std_logic_1164.all;
```

```
use ieee.std_logic_unsigned.all;
```

```
entity adder_4_b is
```

```
    port(B, A : in std_logic_vector(3 downto 0);
```

```
          C0 : in std_logic;
```

```
          S : out std_logic_vector(3 downto 0);
```

```
          C4: out std_logic);
```

```
end adder_4_b;
```

```
architecture behavioral of adder_4_b is
```

```
    signal sum : std_logic_vector(4 downto 0);
```

```
begin
```

```
    sum <= ('0' & A) + ('0' & B) + ("0000" & C0);
```

```
    C4 <= sum(4);
```

```
    S <= sum(3 downto 0);
```

```
end behavioral;
```


Positive Edge-Triggered D Flip-Flop

Process Description (entity)

-- Positive Edge-Triggered D Flip-Flop with Reset:

-- VHDL Process Description

library ieee;

use ieee.std_logic_1164.all;

entity dff is

port(CLK, RESET, D: in std_logic;

Q, Q_n: out std_logic);

end dff;

Positive Edge-Triggered D Flip-Flop

Process Description (architecture)

```
architecture pet_pr of dff is
```

```
-- Implements positive edge-triggered bit state storage
```

```
-- with asynchronous reset.
```

```
signal state: std_logic;
```

```
begin
```

```
Q <= state;
```

```
Q_n <= not state;
```

```
process (CLK, RESET)
```

```
begin
```

```
if (RESET = '1') then
```

```
state <= '0';
```

```
else
```

```
if (CLK'event and CLK = '1') then
```

```
state <= D;
```

```
end if;
```

```
end if;
```

```
end process;
```

```
end;
```