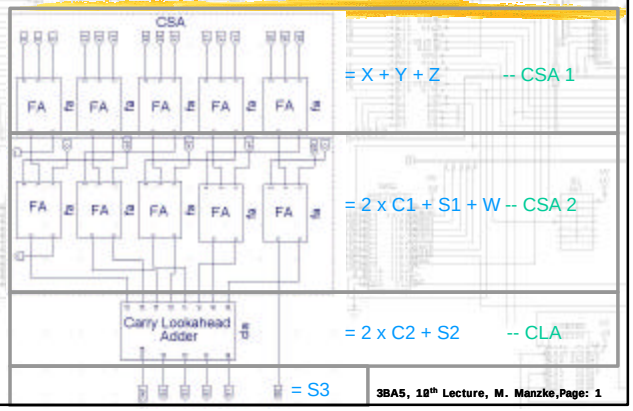
4-bit CSA/CLA Block Diagram

S = X + Y + Z + W



4-bit CSA/CLA VHDL Code

X + Y + Z -- CSA [entity CSA]

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity CSA is
   Port ( X : in std_logic_vector(4 downto 0);
          Y : in std_logic_vector(4 downto 0);
          C_in : in std_logic_vector(4 downto 0);
          S : out std_logic_vector(4 downto 0);
          C_out : out std_logic_vector(4 downto 0));
end CSA;
architecture structural of CSA is
  COMPONENT fa
          PORT(    X : IN std_logic;
                   Y : IN std_logic;
                   C_in : IN std_logic;
                   S : OUT std_logic;
                   C_out : OUT std_logic);
  END COMPONENT;
```


4-bit CSA/CLA VHDL Code

X + Y + Z -- CSA [begin]

```
begin

fa_0: fa PORT MAP(X => X(0),
                  Y => Y(0),
                  C_in => C_in(0),
                  S => S(0),
                  C_out => C_out(0));

fa_1: fa PORT MAP(X => X(1),
                  Y => Y(1),
                  C_in => C_in(1),
                  S => S(1),
                  C_out => C_out(1));

fa_2: fa PORT MAP(X => X(2),
                  Y => Y(2),
                  C_in => C_in(2),
                  S => S(2),
                  C_out => C_out(2));
```


4-bit CSA/CLA VHDL Code

X + Y + Z -- CSA [end structural]

```
fa_3: fa PORT MAP(X => X(3),
                  Y => Y(3),
                  C_in => C_in(3),
                  S => S(3),
                  C_out => C_out(3));

fa_4: fa PORT MAP(X => X(4),
                  Y => Y(4),
                  C_in => C_in(4),
                  S => S(4),
                  C_out => C_out(4));

end structural;
```

4-bit CSA/CLA VHDL Code

2 x C1 + S1 + W -- FA []

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity FA is
   Port ( X : in std_logic;
          Y : in std_logic;
          C_in : in std_logic;
          S : out std_logic;
          C_out : out std_logic);
end FA;
architecture concurrent_behavioral of FA is
signal S1, S2, S3 : std_logic;
begin
          S1 <= (X xor Y);
          S2 <= (c_in and S1);
          S3 <= (X and Y);
          S <= (S1 xor C_in);
          C_out <= (S2 xor S3);
end concurrent_behavioral;
```


4-bit CSA/CLA VHDL Code

2 x C2 + S2 -- CLA [entity CLA]

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity CLA is
   Port ( X : in std_logic_vector(3 downto 0);
          Y : in std_logic_vector(3 downto 0);
          S : out std_logic_vector(3 downto 0);
          C_out : out std_logic);
end CLA;
architecture concurrent_behavioral of CLA is
          signal p0, p1, p2, p3, g0, g1, g2, g3 : std_logic;
          signal c0p0  : std_logic;
          signal p1g0, p1p0c0  : std_logic;
          signal p2g1, p2p1g0, p2p1p0c0 : std_logic;
          signal p3g2, p3p2g1, p3p2p1g0, p3p2p1p0c0 : std_logic;
          signal c1, c2, c3 : std_logic;
          signal c0 : std_logic := '0';
```


4-bit CSA/CLA VHDL Code

2 x C2 + S2 -- CLA [begin]

```
begin

p0 <= (X(0) or Y(0));
p1 <= (X(1) or Y(1));
p2 <= (X(2) or Y(2));
p3 <= (X(3) or Y(3));
g0 <= (X(0) and Y(0));
g1 <= (X(1) and Y(1));
g2 <= (X(2) and Y(2));
g3 <= (X(3) and Y(3));

c0p0  <= (p0 and c0);
c1 <= (g0 or c0p0);

p1g0 <= (p1 and g0);
p1p0c0 <= (p1 and p0 and c0);
c2 <= (g1 or p1g0 or p1p0c0);
```


4-bit CSA/CLA VHDL Code

2 x C2 + S2 -- CLA [end concurrent_behavioral]

```
p2g1 <= (p2 and g1);
p2p1g0 <= (p2 and p1 and g0);
p2p1p0c0 <= (p2 and p1 and p0 and c0);
c3 <= (g2 or p2g1 or p2p1g0 or p2p1p0c0);

p3g2 <= (p3 and g2);
p3p2g1 <= (p3 and p2 and g1);
p3p2p1g0 <= (p3 and p2 and p1 and g0);
p3p2p1p0c0 <= (p3 and p2 and p1 and p0 and c0);
C_out <= (g3 or p3g2 or p3p2g1 or p3p2p1g0 or p3p2p1p0c0);

S(0) <= (X(0) xor Y(0) xor c0);
S(1) <= (X(1) xor Y(1) xor c1);
S(2) <= (X(2) xor Y(2) xor c2);
S(3) <= (X(3) xor Y(3) xor c3);

end concurrent_behavioral;
```

4-bit CSA/CLA VHDL Code

S3 = X + Y + Z + W [entity XYZW_ADDER]

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity XYZW_ADDER is
    Port ( X : in std_logic_vector(3 downto 0);
           Y : in std_logic_vector(3 downto 0);
           Z : in std_logic_vector(3 downto 0);
           W : in std_logic_vector(3 downto 0);
           S : out std_logic_vector(4 downto 0);
           C_out : out std_logic);
end XYZW_ADDER;
```


4-bit CSA/CLA VHDL Code

S3 = X + Y + Z + W [COMPONENT csa/cla]

```
architecture structural of XYZW_ADDER is

COMPONENT csa
        PORT(   X : IN std_logic_vector(4 downto 0);
                Y : IN std_logic_vector(4 downto 0);
                C_in : IN std_logic_vector(4 downto 0);
                S : OUT std_logic_vector(4 downto 0);
                C_out : OUT std_logic_vector(4 downto 0));
END COMPONENT;

COMPONENT cla
        PORT(
                X : IN std_logic_vector(3 downto 0);
                Y : IN std_logic_vector(3 downto 0);
                S : OUT std_logic_vector(3 downto 0);
                C_out : OUT std_logic);
END COMPONENT;
```


4-bit CSA/CLA VHDL Code

S3 = X + Y + Z + W [signal]

```
-- signal that connect the input ports with the CSAs
signal X_in, Y_in, Z_in, W_in : std_logic_vector(4 downto 0);

-- signal that connect the 1st CSA with the 2nd CSA
signal S_01, C_out_01 : std_logic_vector(4 downto 0);

-- signal that connect the 2nd CSA with the CLA
signal S_CSA_2, C_out_12, C_out_01S : std_logic_vector(4 downto 0);

begin

X_in <= ('0' & X);
Y_in <= ('0' & Y);
Z_in <= ('0' & Z);
W_in <= ('0' & W);
C_out_01S <= (C_out_01(3 downto 0) & '0');
S(0) <= S_CSA_2(0);
```

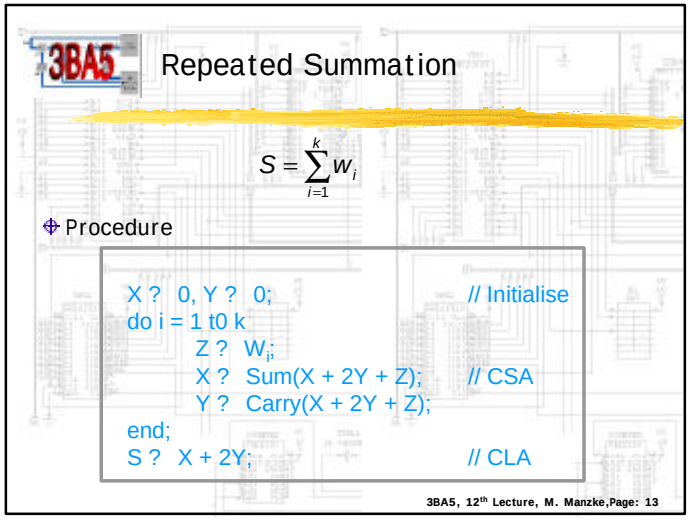

4-bit CSA/CLA VHDL Code

S3 = X + Y + Z + W [PORT MAP]

```
Inst0_csa: csa PORT MAP(    X => X_in,
                            Y => Y_in,
                            C_in => Z_in,
                            S => S_01,
                            C_out => C_out_01);

Inst1_csa: csa PORT MAP(    X => S_01,
                            Y => W_in,
                            C_in => C_out_01S,
                            S => S_CSA_2,
                            C_out => C_out_12);

Inst0_cla: cla PORT MAP(    X => S_CSA_2(4 downto 1),
                            Y => C_out_12(3 downto 0),
                            S => S(4 downto 1),
                            C_out => C_out);
end structural;
```


3BA5
Repeated Summation
$S = \sum_{i=1}^{k} w_i$
Procedure
X ? 0, Y ? 0; // Initialise
do i = 1 t0 k
Z ? W_i;
X ? Sum(X + 2Y + Z); // CSA
Y ? Carry(X + 2Y + Z);
end;
S ? X + 2Y; // CLA
3BA5, 12th Lecture, M. Manzke,Page: 13