RELAZIONE ELETTRONICA DIGITALE

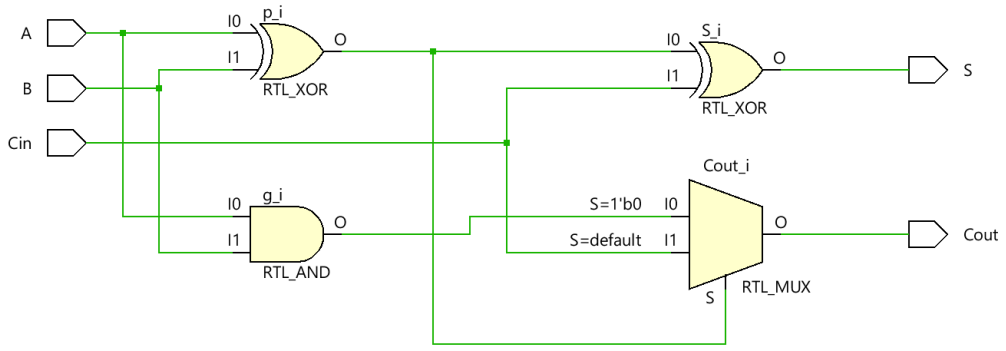
TRACCIA: SI RICHIEDE DI SCRIVERE IL CODICE VHDL DI UN SOMMATORE CARRY-SELECT CON OPERANDI UNSIGNED A 16-BIT E DEL TESTBENCH DA IMPIEGARE PER LA SIMULAZIONE BEHAVIORAL

IL CARRY-SELECT È UN CIRCUITO ADDIZIONATORE UTILIZZATO NELL'ELETTRONICA DIGITALE PER SOMMARE DUE NUMERI BINARI. È PROGETTATO PER RIDURRE IL RITARDO NEL CALCOLO DELLA SOMMA DI DUE NUMERI BINARI, SPECIALMENTE QUANDO I NUMERI SONO ABBASTANZA LUNGHI.

IN QUESTO CIRCUITO, SONO STATI UTILIZZATI TRE SOMMATORI RIPPLE CARRY A 8 BIT. IL PRIMO SOMMATORE CALCOLA LA SOMMA DEI BIT MENO SIGNIFICATIVI E PRODUCE SIA LA SOMMA PARZIALE CHE UN CARRY-OUT. GLI ALTRI DUE SOMMATORI LAVORANO SIMULTANEAMENTE AL PRIMO E GESTISCONO I BIT PIÙ SIGNIFICATIVI. UNO DI ESSI ASSUME UN CARRY-IN DI VALORE 0, MENTRE L'ALTRO ASSUME UN CARRY-IN DI VALORE 1.DI CONSEGUENZA, OGNI COPPIA DI BIT PIÙ SIGNIFICATIVI VIENE SOMMATA DUE VOLTE, UNA VOLTA CONSIDERANDO UN CARRY-IN DI VALORE 0 E L'ALTRA CONSIDERANDO UN CARRY-IN DI VALORE 1. PER CIASCUNA DI QUESTE COPPIE, È STATO UTILIZZATO UN MULTIPLEXER PER DETERMINARE QUALE DEI DUE RISULTATI DELLA SOMMA È CORRETTO, IN BASE AL VALORE DEL CARRY-OUT OTTENUTO DAL SOMMATORE PRECEDENTE.INFINE, UN ULTIMO MULTIPLEXER È UTILIZZATO PER SELEZIONARE TRA I DUE POSSIBILI CARRY DI USCITA DAI SOMMATORI DEI BIT PIÙ SIGNIFICATIVI E DETERMINARE QUALE DI ESSI DOVREBBE ESSERE CONSIDERATO COME CARRY DI USCITA COMPLESSIVO DEL SOMMATORE A CASCATA. LA SELEZIONE AVVIENE IN BASE AL VALORE DEL CARRY-IN ("C7") PROVENIENTE DAL SOMMATORE A 8 BIT MENO SIGNIFICATIVO.

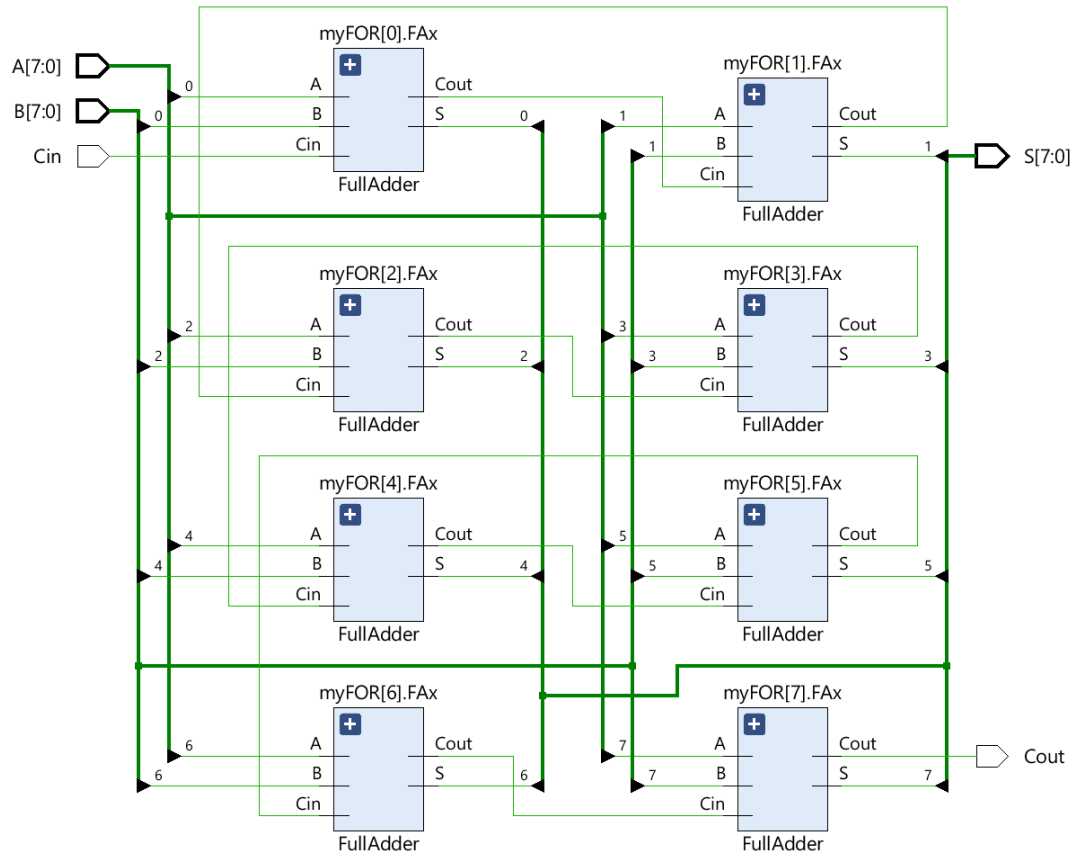
FULL ADDER: CIRCUITO SOMMATORE IN GRADO DI SOMMARE 3 BIT: A,B E CIN, IL RIPORTO, E GENERA COME USCITA S E IL RIPORTO COUT

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FullAdder is
    Port ( A : in bit;
           B : in bit;
           Cin : in bit;
           Cout : out bit;
           S : out bit);
end FullAdder;

architecture Behavioral of FullAdder is
signal p,g: bit;
begin
    Cout <= g when p='0' else Cin;
    S <= p xor Cin;
    p<=a xor b;
    g<= a and b;
end Behavioral;
```



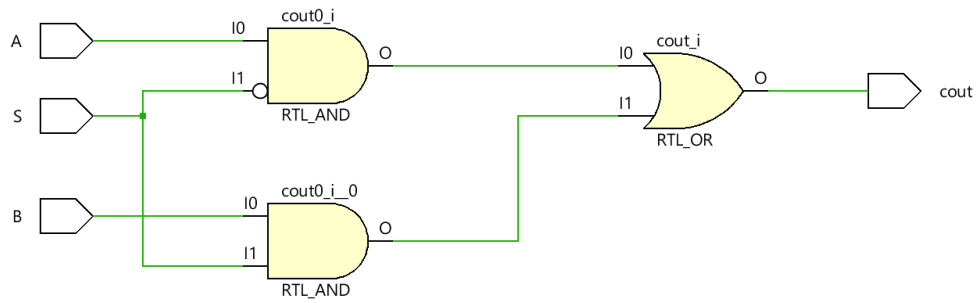
N FULL ADDER IN CASCATA COSTITUISCONO UN RIPPLE CARRY A N BIT

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3
4  entity RippleCarry is
5      Port( A: in bit_vector(7 downto 0);
6            B: in bit_vector(7 downto 0);
7            Cin: in bit;
8            Cout: out bit;
9            S: out bit_vector(7 downto 0));
10 end RippleCarry;
11
12 architecture Behavioral of RippleCarry is
13     component FullAdder is
14         port (A, B, Cin: in bit;
15               Cout, S: out bit);
16     end component;
17
18     signal C: bit_vector(8 downto 0);
19 begin
20     myFOR: for i in 0 to 7 generate
21         FAx: FullAdder port map(A(i), B(i), C(i), C(i+1), S(i));
22     end generate;
23
24     C(0) <= Cin;
25     Cout <= C(8);
26 end Behavioral;
27
```



MULTIPLEXER: COMPONENTE CIRCUITALE IN GRADO DI RICEVERE 2^N SEGNALI IN INGRESSO, INSIEME A N SEGNALI DI CONTROLLO, CHE DETERMINANO QUALE TRA GLI INGRESSI DEVE ESSERE SELEZIONATO E INSTRADATO VERSO L'UNICA USCITA DEL CIRCUITO. UN MULTIPLEXER È COMUNEMENTE UTILIZZATO PER INSTRADARE UNO DEI SUOI INGRESSI IN USCITA IN BASE ALLO STATO DEI SEGNALI DI CONTROLLO.

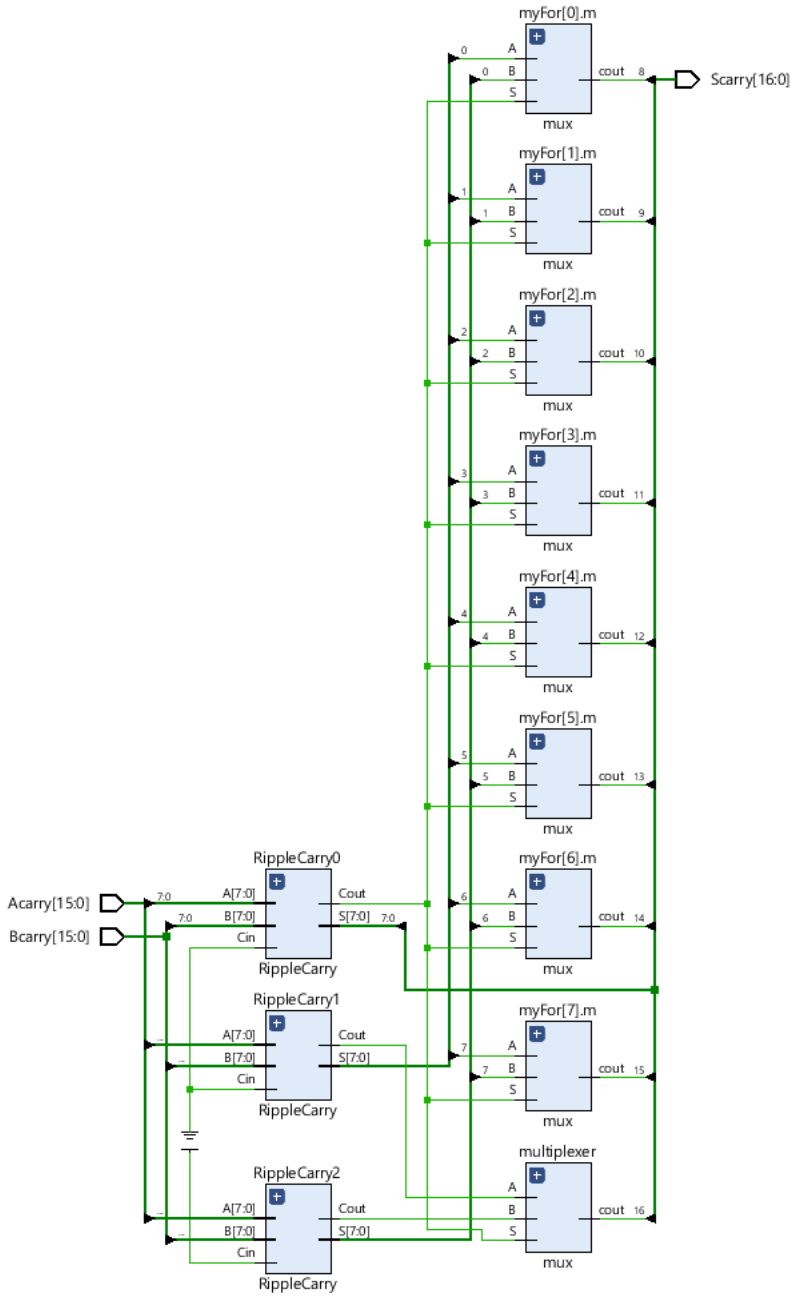
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mux is
    Port ( A: in bit;
           B: in bit;
           S: in bit;
           cout: out bit);
end mux;

architecture Behavioral of mux is
begin
    cout <= (A and (not S)) or (B and S);
end Behavioral;
```



CARRY SELECT

```vhdl
ibrary IEEE;
se IEEE.STD_LOGIC_1164.ALL;

ntity CarrySelect is
    Port (
        Acarry: in bit_vector(15 downto 0);
        Bcarry: in bit_vector(15 downto 0);
        Scarry: out bit_vector(16 downto 0));
nd CarrySelect;

rchitecture Behavioral of CarrySelect is

    component FullAdder is
    Port ( A : in bit;
           B : in bit;
           Cin : in bit;
           Cout : out bit;
           S : out bit);
    end component;

    component RippleCarry is
    Port (A: in bit_vector(7 downto 0);
          B: in bit_vector(7 downto 0);
          Cin: in bit;
          Cout: out bit;
          S: out bit_vector(7 downto 0));
    end component;

    component mux is
        Port (A: in bit;
              B: in bit;
              S: in bit;
              cout: out bit);
    end component;
```

```
35
36    signal C7, Cout0,Cout1: bit;
37    signal Somma0,Somma1: bit_vector(7 downto 0);
38
39  egin
40     RippleCarry0: RippleCarry port map(Acarry(7 downto 0), Bcarry(7 downto 0), '0', C7, Scarry(7 downto 0));
41     RippleCarry1: RippleCarry port map(Acarry(15 downto 8), Bcarry(15 downto 8), '0', Cout0, Somma0);
42     RippleCarry2: RippleCarry port map(Acarry(15 downto 8), Bcarry(15 downto 8), '1', Cout1, Somma1);
43
44     myFor: for i in 0 to 7 generate
45         m: mux port map (Somma0(i), Somma1(i), C7, Scarry(8+i) );
46     end generate myFor;
47
48     multiplexer: mux port map(Cout0,Cout1,C7,Scarry(16));
49
50  nd Behavioral;
51
```



TESTBENCH: IMPLEMENTAZIONE VHDL UTILIZZATA PER SIMULARE IL FUNZIONAMENTO DEL CIRCUITO DESCRITTO NEL CODICE VHDL PRINCIPALE.

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity simulazioneCarrySelect is

end simulazioneCarrySelect;

architecture Behavioral of simulazioneCarrySelect is
    component CarrySelect is
          Port (
        Acarry: in bit_vector(15 downto 0);
        Bcarry: in bit_vector(15 downto 0);
        Scarry: out bit_vector(16 downto 0));
    end component;

    signal IA, IB: bit_vector(15 downto 0);
    signal Somma: bit_vector(16 downto 0);
begin
    IA <= "0011010101011010";
    IB <= "0101010111101010";


    CarrySelectAdder: CarrySelect port map(IA, IB, Somma);
end Behavioral;
```