LAB 05

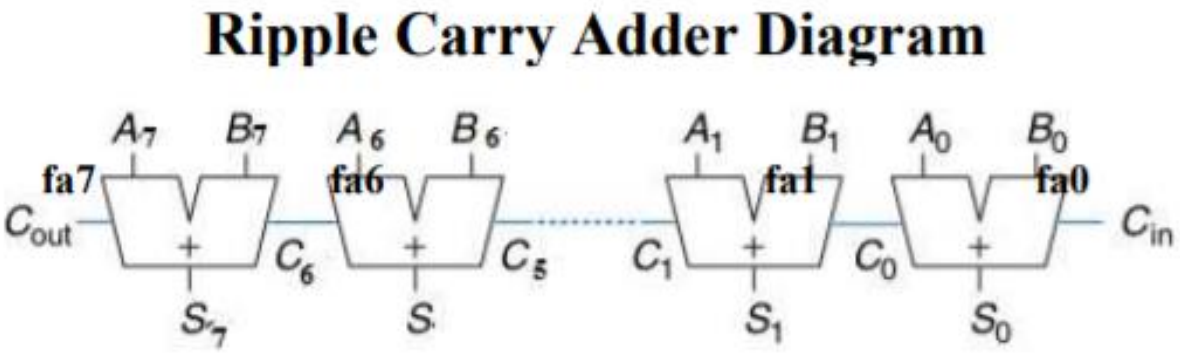
8-Bit Ripple Carry Adder

CECS 225 – DIGITAL LOGIC AND ASSEMBLY PROGRAMMING

Professor: Xiaolong Wu
Student: Thanh Nguyen _ ID: 026843815

1. Introduction:

- *Definition:* 8-bit Ripple Carry Adder is a logic circuit used for adding the 8-bit numbers in digital operations. It is the center of most digital circuits that perform addition or subtraction. It adds together two 8-bit binary digits, plus a carry-in digit to produce a sum and a carry-out digit.
- *Structure:*
    - It has 8 Fully Adders (numbered 0 through 7) tied together.
    - It requires 3 inputs and produces 2 outputs.
        - Three inputs are two 8-bit hexadecimal digits A_8 and B_8, and a 1-bit binary digit called Cin_1.
        - The outputs are the sum of two 8-bit hexadecimal digits, which is an 8-bit output S_8 and a 1-bit digit carry-out is Cout_1
- *Diagram:*



2. Function Table with 8 test cases

Inputs				Outputs	
Test Cases	**Cin**	**A_8**	**B_8**	**Cout**	**S_8**
0	0	12	34	0	46
1	0	55	AA	0	FF
2	0	72	27	0	99
3	0	80	08	0	88
4	1	12	34	0	47
5	1	55	AA	1	0
6	1	72	27	0	9A
7	1	80	08	0	89

3. Verilog Codes

- ***Design Code***

```
module HalfAdder(A, B, Cout, S);
  input A, B;
  output Cout, S;

  assign Cout = A & B;
  assign S = A ^ B;

endmodule

module FullAdder(Cin, FA_A, FA_B, FA_S, FA_Cout);
  input FA_A, FA_B, Cin;
  output FA_S, FA_Cout;

  wire ha0_S, ha0_C, ha1_C;

  HalfAdder ha0 (   .A   (FA_A) ,
                    .B   (FA_B) ,
                    .Cout(ha0_C),
                    .S   (ha0_S)
                );

  HalfAdder ha1 (   .A   (Cin)  ,
                    .B   (ha0_S),
                    .Cout(ha1_C),
                    .S   (FA_S)
                );

  // This is the carry out for the Full Adder
  assign FA_Cout = ha0_C | ha1_C;

endmodule

module RCA8(A_8, B_8, Cin_1, Cout_1, S_8);

  input        Cin_1;
  input  [7:0] A_8, B_8;

  output [7:0] S_8;
  output       Cout_1;

  wire c0, c1, c2, c3, c4, c5, c6;
  FullAdder fa0(.Cin    (Cin_1),
                .FA_A   (A_8[0]),
                .FA_B   (B_8[0]),
                .FA_S   (S_8[0]),
                .FA_Cout(c0)
               );

  FullAdder fa1(.Cin    (c0),
                .FA_A   (A_8[1]),
                .FA_B   (B_8[1]),
                .FA_S   (S_8[1]),
                .FA_Cout(c1)
               );

  FullAdder fa2(.Cin    (c1),
```

```
                .FA_A   (A_8[2]),
                .FA_B   (B_8[2]),
                .FA_S   (S_8[2]),
                .FA_Cout(c2)
               );

  FullAdder fa3(.Cin    (c2),
                .FA_A   (A_8[3]),
                .FA_B   (B_8[3]),
                .FA_S   (S_8[3]),
                .FA_Cout(c3)
               );

  FullAdder fa4(.Cin    (c3),
                .FA_A   (A_8[4]),
                .FA_B   (B_8[4]),
                .FA_S   (S_8[4]),
                .FA_Cout(c4)
               );

  FullAdder fa5(.Cin    (c4),
                .FA_A   (A_8[5]),
                .FA_B   (B_8[5]),
                .FA_S   (S_8[5]),
                .FA_Cout(c5)
               );

  FullAdder fa6(.Cin    (c5),
                .FA_A   (A_8[6]),
                .FA_B   (B_8[6]),
                .FA_S   (S_8[6]),
                .FA_Cout(c6)
               );

  FullAdder fa7(.Cin    (c6),
                .FA_A   (A_8[7]),
                .FA_B   (B_8[7]),
                .FA_S   (S_8[7]),
                .FA_Cout(Cout_1)
               );

endmodule
```

- ***Testbench***

```
`timescale 1ns/1ps

module testbench();
  reg       Cin_11;
  reg [7:0] A_81, B_81;

  wire [7:0] S_81;
  wire       Cout_11;

  RCA8 rca(A_81, B_81, Cin_11, Cout_11, S_81);
```

```
initial
  begin
    //Dump waves
    $dumpfile("dump.vcd");
    $dumpvars(1, testbench);

    // Cin_1 = 0, A_8 = 12, B_8 = 34
    $display("Test Case 0");
    Cin_11 = 1'b0; A_81 = 8'h12; B_81 = 8'h34;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

     // Cin_1 = 0, A_8 = 55, B_8 = AA
    $display("Test Case 1");
    Cin_11 = 1'b0; A_81 = 8'h55; B_81 = 8'hAA;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

    // Cin_1 = 0, A_8 = 72, B_8 = 27
    $display("Test Case 2");
    Cin_11 = 1'b0; A_81 = 8'h72; B_81 = 8'h27;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

    // Cin_1 = 0, A_8 = 80, B_8 = 8
    $display("Test Case 3");
    Cin_11 = 1'b0; A_81 = 8'h80; B_81 = 8'h8;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

    // Cin_1 = 1, A_8 = 12, B_8 = 34
    $display("Test Case 4");
    Cin_11 = 1'b1; A_81 = 8'h12; B_81 = 8'h34;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

     // Cin_1 = 1, A_8 = 55, B_8 = AA
    $display("Test Case 5");
    Cin_11 = 1'b1; A_81 = 8'h55; B_81 = 8'hAA;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

    // Cin_1 = 1, A_8 = 72, B_8 = 27
    $display("Test Case 6");
    Cin_11 = 1'b1; A_81 = 8'h72; B_81 = 8'h27;
    $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
    #2
    $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);

    // Cin_1 = 1, A_8 = 80, B_8 = 8
    $display("Test Case 7");
    Cin_11 = 1'b1; A_81 = 8'h80; B_81 = 8'h8;
```

```
        $display("Cin_1  = %b", Cin_11, "     A_8 = %2h", A_81, "      B_8 = %2h", B_81);
        #2
        $display("Cout_1 = %b", Cout_11, "     S_8 = %2h", S_81);
    end

endmodule
```

4. Simulator Waveform

- Inputs and Outputs:

```
Test Case 0
Cin_1  = 0     A_8 = 12      B_8 = 34
Cout_1 = 0     S_8 = 46
Test Case 1
Cin_1  = 0     A_8 = 55      B_8 = aa
Cout_1 = 0     S_8 = ff
Test Case 2
Cin_1  = 0     A_8 = 72      B_8 = 27
Cout_1 = 0     S_8 = 99
Test Case 3
Cin_1  = 0     A_8 = 80      B_8 = 08
Cout_1 = 0     S_8 = 88
Test Case 4
Cin_1  = 1     A_8 = 12      B_8 = 34
Cout_1 = 0     S_8 = 47
Test Case 5
Cin_1  = 1     A_8 = 55      B_8 = aa
Cout_1 = 1     S_8 = 00
Test Case 6
Cin_1  = 1     A_8 = 72      B_8 = 27
Cout_1 = 0     S_8 = 9a
Test Case 7
Cin_1  = 1     A_8 = 80      B_8 = 08
Cout_1 = 0     S_8 = 89
```

- Waveform: