Verilog Assignment-1 (CS39001)

Group 6 - 22CS10030, 22CS30046

August 14, 2024

1 Ripple Carry Adder

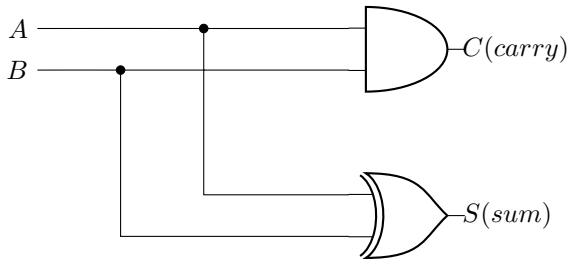
1.1 Half Adder

1.1.1 Truth Table

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 1: Truth Table for Half Adder

1.1.2 Logic Diagram



1.1.3 Verilog Code

```
module halfadder(s,c,a,b);
    input a,b;
    output s,c;
    xor g1(s,a,b);
    and g2(c,a,b);
endmodule
```

Listing 1: Half Adder Behavioral Style

```
module halfadder(s,c,a,b);
    input a,b;
    output s,c;
    xor g1(s,a,b);
    and g2(c,a,b);
endmodule
```

Listing 2: Half Adder Structural Style

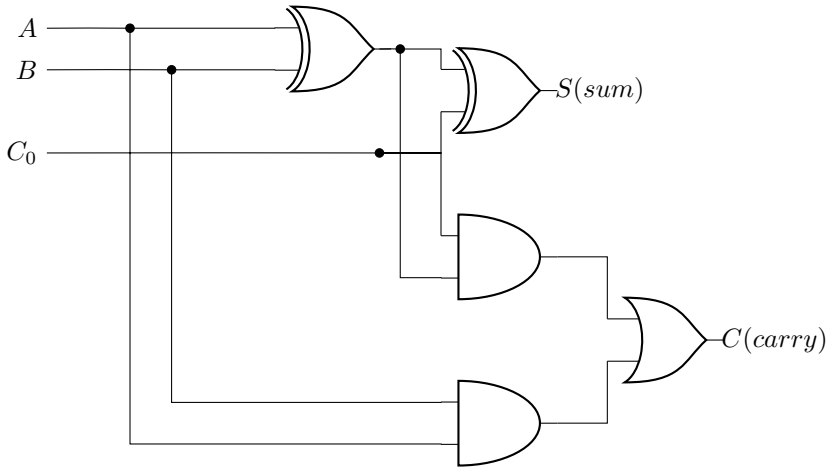
1.2 Full Adder

1.2.1 Truth Table

A	B	C_0	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 2: Truth Table for Full Adder

1.2.2 Logic Diagram



1.2.3 Verilog Code

```
module fulladder(s,c,a,b,c0);
    input a,b,c0;
    output s,c;
    assign s = a^b^c0;
    assign c = a&b | b&c0 | c0&a;
endmodule
```

Listing 3: Full Adder Behavioral Style

```
module fulladder(s,c,a,b,c0);
    input a,b,c0;
    output s,c;
    wire t1,t2,t3;
    xor g1(t1,a,b);
    xor g2(s,c0,t1);
    and g3(t2,a,b);
    and g4(t3,t1,c0);
    or g5(c,t2,t3);
endmodule
```

Listing 4: Full Adder Structural Style

1.3 n-bit Adder

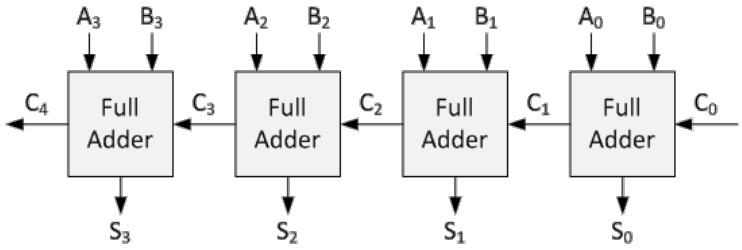


Figure 1: Ripple Carry Adder

1.3.1 8-bit Adder

Cascade eight Full Adders

```
module bitadder8(a, b, cin, s, cout);
    input [7:0] a, b;
    input cin;
    output [7:0] s;
    output cout;

    wire [8:0] carry;
    assign carry[0] = cin;

    fulladder FA0 (.a(a[0]), .b(b[0]), .c0(cin), .c(carry[1]), .s(s
    [0]));
    fulladder FA1 (.a(a[1]), .b(b[1]), .c0(carry[1]), .c(carry[2]),
     .s(s[1]));
```

```
    fulladder FA2 (.a(a[2]), .b(b[2]), .c0(carry[2]), .c(carry[3]),
     .s(s[2]));
    fulladder FA3 (.a(a[3]), .b(b[3]), .c0(carry[3]), .c(carry[4]),
     .s(s[3]));
    fulladder FA4 (.a(a[4]), .b(b[4]), .c0(carry[4]), .c(carry[5]),
     .s(s[4]));
    fulladder FA5 (.a(a[5]), .b(b[5]), .c0(carry[5]), .c(carry[6]),
     .s(s[5]));
    fulladder FA6 (.a(a[6]), .b(b[6]), .c0(carry[6]), .c(carry[7]),
     .s(s[6]));
    fulladder FA7 (.a(a[7]), .b(b[7]), .c0(carry[7]), .c(carry[8]),
     .s(s[7]));

    assign cout = carry[8];
endmodule
```

Listing 5: 8-bit RCA

1.3.2 16-bit Adder

Cascade two 8-bit Adders

```
module bitadder16(a, b, cin, s, cout);
    input [15:0] a, b;
    input cin;
    output [15:0] s;
    output cout;

    wire [1:0] carry;

    bitadder8 BA8_0 (.a(a[7:0]), .b(b[7:0]), .cin(cin), .cout(carry
    [0]), .s(s[7:0]));
    bitadder8 BA8_1 (.a(a[15:8]), .b(b[15:8]), .cin(carry[0]), .
    cout(carry[1]), .s(s[15:8]));

    assign cout = carry[1];
endmodule
```

Listing 6: 16-bit RCA

1.3.3 32-bit Adder

Cascade two 16-bit Adders

```
module bitadder32(a, b, cin, s, cout);
    input [31:0] a, b;
    input cin;
    output [31:0] s;
    output cout;

    wire [1:0] carry;

    bitadder16 BA16_0 (.a(a[15:0]), .b(b[15:0]), .cin(cin), .cout(
    carry[0]), .s(s[15:0]));
    bitadder16 BA16_1 (.a(a[31:16]), .b(b[31:16]), .cin(carry[0]),
    .cout(carry[1]), .s(s[31:16]));
```

```
11
12      assign cout = carry[1];
13  endmodule
```

Listing 7: 32-bit RCA

1.3.4 64-bit Adder

Cascade two 32-bit Adders

```
1   module bitadder64(a, b, cin, s, cout);
2       input [63:0] a, b;
3       input cin;
4       output [63:0] s;
5       output cout;
6
7       wire [1:0] carry;
8
9       bitadder32 BA32_0 (.a(a[31:0]), .b(b[31:0]), .cin(cin), .cout(
        carry[0]), .s(s[31:0]));
10      bitadder32 BA32_1 (.a(a[63:32]), .b(b[63:32]), .cin(carry[0]),
        .cout(carry[1]), .s(s[63:32]));
11
12      assign cout = carry[1];
13  endmodule
```

Listing 8: 64-bit RCA

1.4 n-bit Subtractor

Using n-bit Adder, two numbers can be subtracted using the relation:
$A - B = A + (\sim B) + 1$

2 Carry Look-Ahead Adder

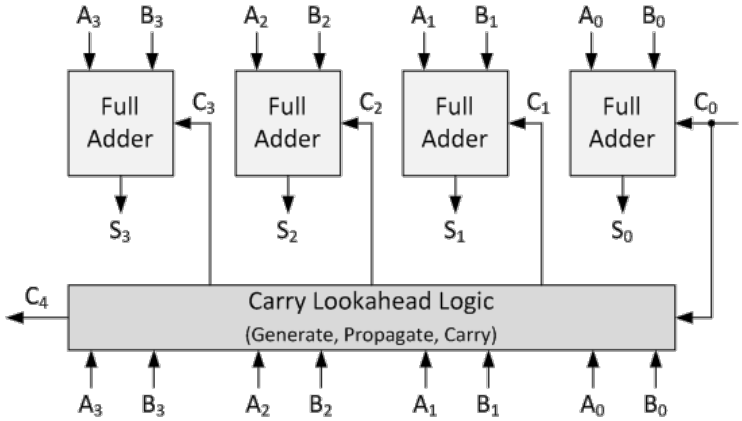


Figure 2: Carry Look-Ahead Adder

2.1 4-bit CLA

2.1.1 Logic Equations

$$P_i = A_i \oplus B_i$$
$$G_i = A_i B_i$$

$$S_i = P_i \oplus C_i$$
$$C_{i+1} = G_i + P_i C_i$$

$$C_1 = G_0 + P_0 C_{in}$$
$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_{in}$$
$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{in}$$
$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_{in}$$

2.1.2 Verilog Code

```
module cla4bit(a, b, cin, s, cout);
  input [3:0] a, b;
    input cin;
    output [3:0] s;
    output cout;

  wire [4:0] C;
  wire [3:0] P, G;
```

```
9
10    assign G = a&b;
11    assign P = a^b;
12    assign C[0]=cin;
13
14    assign C[1] = G[0]|(P[0]&C[0]);
15    assign C[2] = G[1]|(P[1]&G[0])|(P[1]&P[0]&C[0]);
16    assign C[3] = G[2]|(P[2]&G[1])|(P[2]&P[1]&G[0])|(P[2]&P[1]&P[0]&C
        [0]);
17    assign C[4] = G[3]|(P[3]&G[2])|(P[3]&P[2]&G[1])|(P[3]&P[2]&P[1]&G
        [0])|(P[3]&P[2]&P[1]&P[0]&C[0]);
18
19    assign s = C[3:0]^P;
20    assign cout = C[4];
21  endmodule
```

Listing 9: 4-bit CLA

2.2 16-bit CLA

2.2.1 Augmented 4-bit CLA

$$
\begin{aligned}
P &= P3P2P1P0 \\
G &= G3 + P3G2 + P3P2G1 + P3P2P1G0
\end{aligned}
$$

```
1   module cla4bit_aug(a, b, cin, s, p, g);
2     input [3:0] a, b;
3       input cin;
4       output [3:0] s;
5       output p, g;
6
7     wire [4:0] C;
8     wire [3:0] P, G;
9
10    assign G = a&b;
11    assign P = a^b;
12    assign C[0]=cin;
13
14    assign C[1] = G[0]|(P[0]&C[0]);
15    assign C[2] = G[1]|(P[1]&G[0])|(P[1]&P[0]&C[0]);
16    assign C[3] = G[2]|(P[2]&G[1])|(P[2]&P[1]&G[0])|(P[2]&P[1]&P[0]&C
        [0]);
17    assign C[4] = G[3]|(P[3]&G[2])|(P[3]&P[2]&G[1])|(P[3]&P[2]&P[1]&G
        [0])|(P[3]&P[2]&P[1]&P[0]&C[0]);
18
19    assign s = C[3:0]^P;
20    assign p = &P;
21    assign g = G[3]|(P[3]&G[2])|(P[3]&P[2]&G[1])|(P[3]&P[2]&P[1]&G
        [0]);
22  endmodule
```

Listing 10: Augmented 4-bit CLA

2.2.2 Lookahead Carry Unit

```
module lcu(P,G,cin,C);
  input [3:0] P, G;
  input cin;

  output [4:0] C;

  assign C[0] = cin;
  assign C[1] = G[0]|(P[0]&C[0]);
  assign C[2] = G[1]|(P[1]&G[0])|(P[1]&P[0]&C[0]);
  assign C[3] = G[2]|(P[2]&G[1])|(P[2]&P[1]&G[0])|(P[2]&P[1]&P[0]&C
    [0]);
  assign C[4] = G[3]|(P[3]&G[2])|(P[3]&P[2]&G[1])|(P[3]&P[2]&P[1]&G
    [0])|(P[3]&P[2]&P[1]&P[0]&C[0]);
endmodule
```

Listing 11: Lookahead Carry Unit

2.2.3 Final Module

```
module cla16bit(a,b,cin,s,cout);
  input [15:0] a, b;
  input cin;
  output [15:0] s;
  output cout;

  wire [3:0] P, G;
  wire [4:0] C;

  cla4bit_aug AUG0(.a(a[3:0]),.b(b[3:0]),.cin(C[0]),.p(P[0]),.g(G
    [0]),.s(s[3:0]));
  cla4bit_aug AUG1(.a(a[7:4]),.b(b[7:4]),.cin(C[1]),.p(P[1]),.g(G
    [1]),.s(s[7:4]));
  cla4bit_aug AUG2(.a(a[11:8]),.b(b[11:8]),.cin(C[2]),.p(P[2]),.g(G
    [2]),.s(s[11:8]));
  cla4bit_aug AUG3(.a(a[15:12]),.b(b[15:12]),.cin(C[3]),.p(P[3]),.g
    (G[3]),.s(s[15:12]));
  lcu LCU0(.P(P),.G(G),.cin(cin),.C(C));

  assign cout = C[4];
endmodule
```

Listing 12: 16-bit CLA