

LAB: 03 16 bit hybrid CLA addition subtraction circuit

Ahmet Poyraz Güler ID: 30971

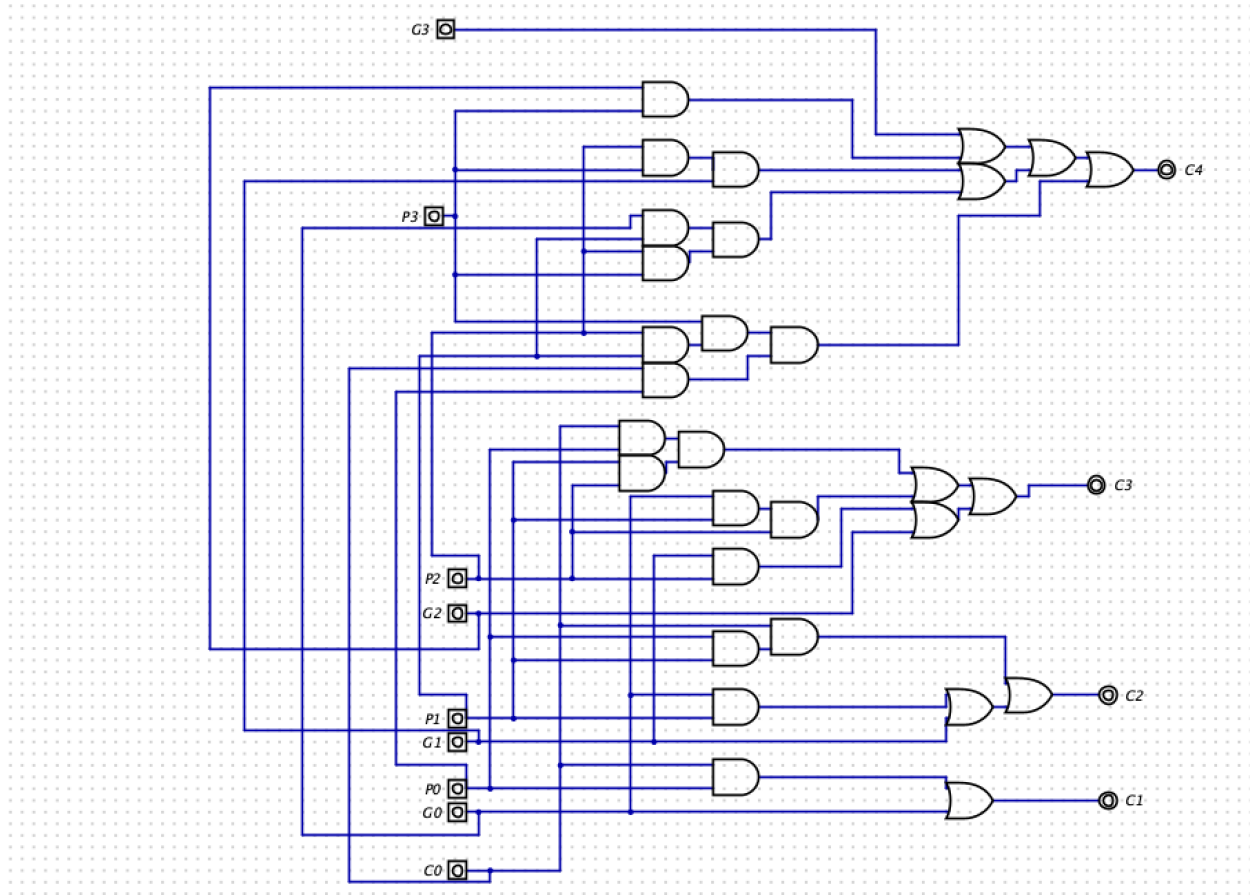
SABANCI UNIVERSITY

First part: Designing the circuit

Second Part: writing the Verilog code

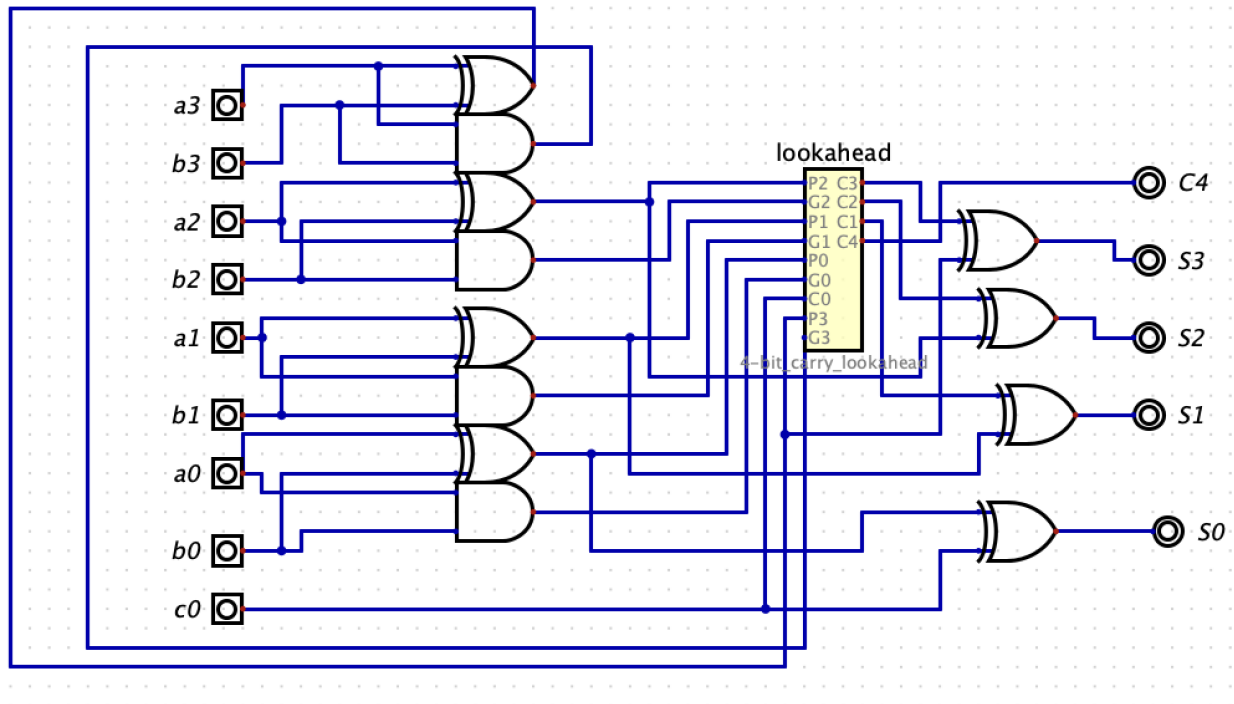
Third Part: Testing for overflow and carryout

Designing a 4 bit Carry Lookahead:



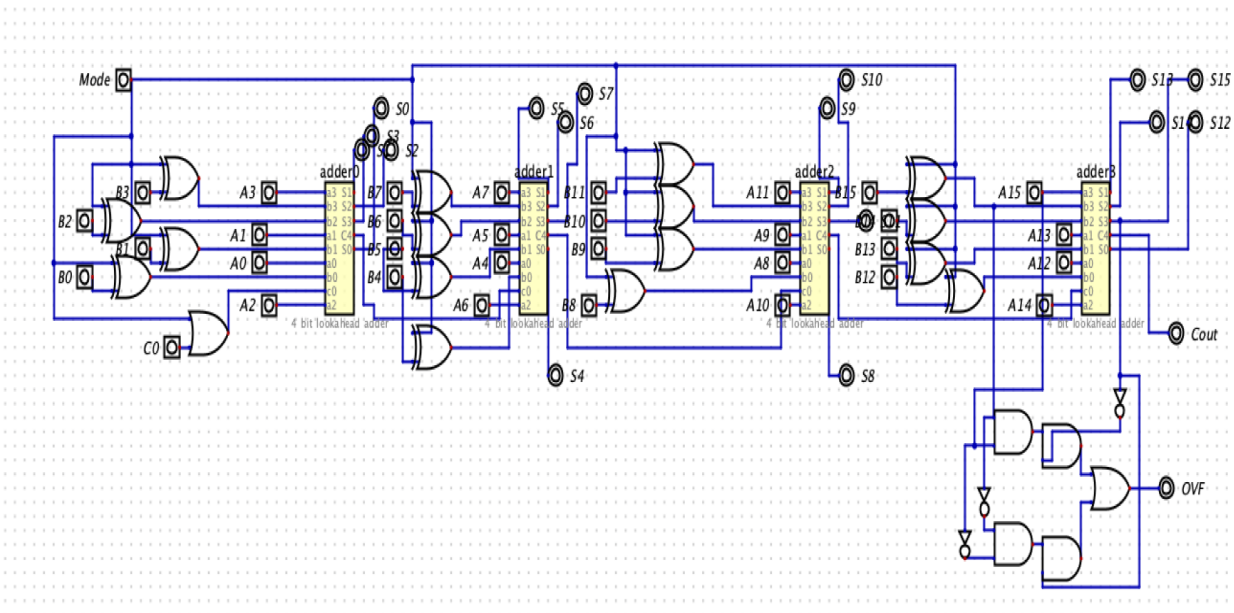
C_4 is produced in the same circuit as the rest for simplicity in design.

Designing a 4 bit Carry Lookahead Adder:



Aim for creating 4 bit CLA is to create a hybrid 16 bit adder.

16 bit hybrid CLA adder-subtractor :



In this design OVF is calculated through this formula:

$$V = a_{n-1} \cdot b_{n-1} \cdot S_{n-1} + a_{n-1} \cdot b_{n-1} \cdot S_{n-1}$$

This method is preferred instead of $V = C_{n-1} \text{ XOR } C_{n-2}$ since 4 bit adders hide the second previous carry.

3 module is created for 3 levels of hierarchical design.

```
module \4_bit_carry_lookahead (
input P2,
input G2,
input P1,
input G1,
input P0,
input G0,
input C0,
input P3,
input G3,
output C3,
output C2,
output C1,
output C4
);
assign C3 = (((C0 & P0) & (P1 & P2)) | ((G0 & P1) & P2)) | ((G1 & P2) | G2));
assign C1 = ((C0 & P0) | G0);
assign C2 = ((C0 & (P0 & P1)) | ((G0 & P1) | G1));
assign C4 = (((G3 | (G2 & P3)) | (((P2 & P3) & G1) | ((G0 & P1) & (P2 & P3)))) | ((P3
& (P2 & P1)) & (C0 & P0)));
endmodule
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
module \4_bit_lookahead_adder (
input a3,
input b3,
input b2,
input a1,
input b1,
input a0,
input b0,
input c0,
input a2,
output S1,
output S2,
output S3,
output C4,
output S0
);
wire s4;
wire s5;
wire s6;
wire s7;
wire s8;
wire s9;
wire s10;
wire s11;
wire s12;
wire s13;
wire s14;
assign s4 = (a3 & b3);
assign s5 = (a2 & b2);
assign s6 = (a0 & b0);
assign s7 = (a1 & b1);
assign s8 = (a3 ^ b3);
assign s9 = (a2 ^ b2);
assign s10 = (a1 ^ b1);
assign s11 = (a0 ^ b0);
// lookahead
\4_bit_carry_lookahead \4_bit_carry_lookahead_i0 (
.P2( s9 ),
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
.G2( s5 ),
.P1( s10 ),
.G1( s7 ),
.P0( s11 ),
.G0( s6 ),
.C0( c0 ),
.P3( s8 ),
.G3( s4 ),
.C3( s12 ),
.C2( s13 ),
.C1( s14 ),
.C4( C4 )
);
assign S0 = (s11 ^ c0);
assign S1 = (s14 ^ s10);
assign S2 = (s13 ^ s9);
assign S3 = (s12 ^ s8);
endmodule

module CLA_add_sub (input [15:0] A,
input [15:0] B,
input Mode, // 0 --> addition , 1 --> subtraction
output [15:0] S,
output Cout,
output OVF
);
wire s16;
wire s17;
wire s18;
wire s19;
wire s20;
wire s21;
wire s22;
wire s23;
wire s24;
wire s25;
wire s26;
wire s27;
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
wire s28;
wire s29;
wire s30;
wire s31;
wire s32;
wire s33;
wire s34;
wire s35;
wire S15_temp;
assign s16 = (Mode ^ B[3]);
assign s17 = (Mode ^ B[2]);
assign s18 = (Mode ^ B[1]);
assign s19 = (Mode ^ B[0]);
assign s22 = (Mode ^ B[7]);
assign s23 = (Mode ^ B[6]);
assign s24 = (Mode ^ B[5]);
assign s25 = (Mode ^ B[4]);
assign s30 = (Mode ^ B[8]);
assign s29 = (Mode ^ B[9]);
assign s28 = (Mode ^ B[10]);
assign s27 = (Mode ^ B[11]);
assign s32 = (Mode ^ B[15]);
assign s33 = (Mode ^ B[14]);
assign s34 = (Mode ^ B[13]);
assign s35 = (Mode ^ B[12]);
assign s20 = (Mode | 0);
// adder0
\4_bit_lookahead_adder \4_bit_lookahead_adder_i0 (
.a3( A[3] ),
.b3( s16 ),
.b2( s17 ),
.a1( A[1] ),
.b1( s18 ),
.a0( A[0] ),
.b0( s19 ),
.c0( s20 ),
.a2( A[2] ),
.S1( S[1] ),
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
.S2( S[2] ),
.S3( S[3] ),
.C4( s21 ),
.S0( S[0] )
);

// adder1
\4_bit_lookahead_adder \4_bit_lookahead_adder_i1 (
.a3( A[7] ),
.b3( s22 ),
.b2( s23 ),
.a1( A[5] ),
.b1( s24 ),
.a0( A[4] ),
.b0( s25 ),
.c0( s21 ),
.a2( A[6] ),
.S1( S[5] ),
.S2( S[6] ),
.S3( S[7] ),
.C4( s26 ),
.S0( S[4] )
);

// adder2
\4_bit_lookahead_adder \4_bit_lookahead_adder_i2 (
.a3( A[11] ),
.b3( s27 ),
.b2( s28 ),
.a1( A[9] ),
.b1( s29 ),
.a0( A[8] ),
.b0( s30 ),
.c0( s26 ),
.a2( A[10] ),
.S1( S[9] ),
.S2( S[10] ),
.S3( S[11] ),
.C4( s31 ),
.S0( S[8] )
```


LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
);  
// adder3  
\4_bit_lookahead_adder \4_bit_lookahead_adder_i3 (  
.a3( A[15] ),  
.b3( s32 ),  
.b2( s33 ),  
.a1( A[13] ),  
.b1( s34 ),  
.a0( A[12] ),  
.b0( s35 ),  
.c0( s31 ),  
.a2( A[14] ),  
.S1( S[13] ),  
.S2( S[14] ),  
.S3( S15_temp ),  
.C4( Cout ),  
.S0( S[12] )  
);  
assign OVF = ((s32 & A[15]) & ~ S15_temp) | ((~ s32 & ~ A[15]) & S15_temp));  
assign S[15] = S15_temp;  
endmodule
```

Simulation code is created for the circuit.

```
module CLA_16bit_tb();  
  
reg [15:0] A, B; // Inputs  
reg mode; // mode (add or subtract)  
wire [15:0] S; // result  
wire Ovf; // Outputs are wires.  
wire Cout;  
  
CLA_add_sub dut (.A(A), .B(B), .Mode(mode), .S(S), .Cout(Cout), .OVF(Ovf)); // Our  
design-under-test.
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
initial begin
    // * Our waveform is saved under this file.
    $dumpfile("CLA_16bit_top.vcd");
    // * Get the variables from the module.

    $dumpvars(0,CLA_16bit_tb);

    $display("Simulation started.");

    A = 16'd0; // Set all inputs to zero.
    B = 16'd0;
    mode = 1'd0;
    #10; // Wait 10 time units.
    A = 16'd25;
    B = 16'd50;
    mode = 1'd0; // For addition
    #10; // Wait 10 time units.
    A = 16'd30;
    B = 16'd100;
    mode = 1'd1; // For subtraction
    #10; // Wait 10 time units.

    A = 16'd2; // normal
    B = 16'd1;
    mode = 1'd0; // Addition
    #10;
    A = 16'd24576; //ovf
    B = 16'd24576;
    mode = 1'd0; // Addition
    #10;
    // cout
    A = 16'd57344;
    B = 16'd57344;
    mode = 1'd0; // Addition
    #10;
    // cout & OVF
    A = 16'd45056;
```

LAB: 03 16 bit hybrid CLA addition subtraction circuit

```
B = 16'd36864;
mode = 1'd0; // Addition
#10;

//ovf
A = 16'd0;
B = 16'd32768;
mode = 1'd1; // subtraction
#10;
A = 16'd32768; // cout
B = 16'd32768;
mode = 1'd1; // Subtraction
#10;
A = 16'd57344; //overflow and count
B = 16'd28672;
mode = 1'd1; // Subtraction
#10;

//normal
A = 16'd1;
B = 16'd2;
mode = 1'd1; // Subtraction
#10;

$display("Simulation finished.");
$finish(); // Finish simulation.
end

endmodule
```

Test cases have been calculated by hand.

LAB: 03 16 bit hybrid CLA addition subtraction circuit

...../...../.....

| | |
|-----------------------------------|-------------|
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 |
| - 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 32768 |
| | ovf no Cout |

| | |
|-----------------------------------|-------------|
| 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 32768 |
| - 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 32768 |
| | Cout no ovf |

15 14 13

| | |
|-----------------------------------|----------|
| 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 | 57344 |
| - 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 | 28672 |
| | ovf Cout |

0100

| | |
|-----------------------------------|----------------|
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 1 |
| - 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 | 2 |
| | No Cout no OVF |

| | |
|-----------------------------------|-------------------|
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 | 2 |
| + 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 | 1 |
| | No Cout No OVF |

| | |
|-----------------------------------|----------------|
| 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 | 24576 |
| + 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 | 24576 |
| | No Cout OVF |




| | |
|-----------------------------------|----------------|
| 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 | 57344 |
| + 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 | 24576 |
| | Cout No OVF |

2^{15} 2^{14} 2^{13} 2^{12}

| | |
|-----------------------------------|-------------|
| 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 | 45056 |
| + 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 | 36864 |
| | Cout OVF |

LAB: 03 16 bit hybrid CLA addition subtraction circuit

And these test cases have been tested with the software:

| | | | | | | | | | | | | | | |
|---|------|-------------------|------|------|------|------|------|------|------|------|------|------|------|------|
|  | 15:0 | CLA 16bit tb.S | FFFF | 0000 | 004B | FFBA | 0003 | C000 | C000 | 4000 | 8000 | 0000 | 7000 | FFFF |
|  | | CLA 16bit tb.Ovf | 0 | | | | | | | | | | | |
|  | | CLA 16bit tb.Cout | 0 | | | | | | | | | | | |
|  | 15:0 | CLA 16bit tb.A | 0001 | 0000 | 0019 | 001E | 0002 | 6000 | E000 | B000 | 0000 | 8000 | E000 | 0001 |
|  | 15:0 | CLA 16bit tb.B | 0002 | 0000 | 0032 | 0064 | 0001 | 6000 | E000 | 9000 | 8000 | 7000 | 0002 | |
|  | | CLA 16bit tb.mode | 1 | | | | | | | | | | | |

A functioning adder/subtractor is made.

Lecture slides have been utilized in this project