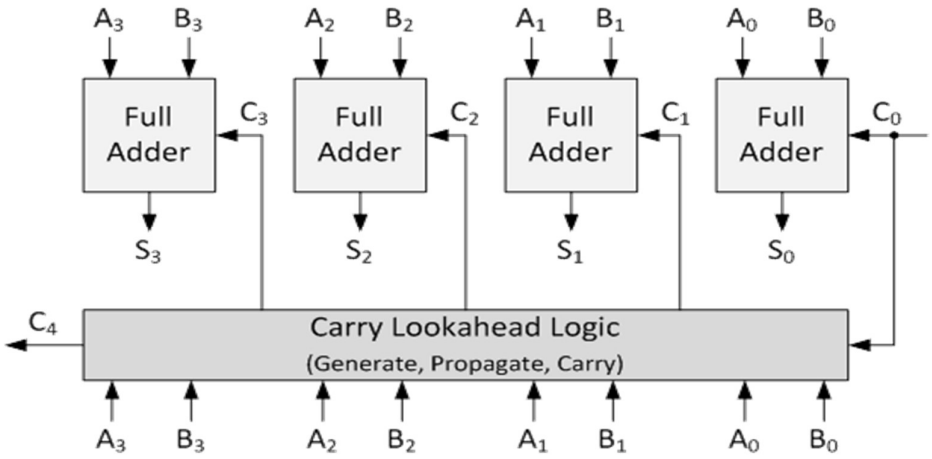
Dept. of CSE, Bennett University

Digital Design – ECSE 108L

Lab Assignment – 7

In this lab, we shall start with Carry Lookahead Adder. In ripple carry adders, the carry must propagate through the gate levels before the sum is available at the output terminals. One of the most popular methods to reduce delay is to use a carry lookahead mechanism. The propagation delay is reduced to four gate levels, irrespective of the number of bits in the adder.

Q1. Write the Verilog code for a 4-bit carry look-ahead adder (CLA). The block diagram of the same has been given in following figure.



(i) Generate the expressions for the propagators and generators.

(ii) You can use module instantiation for the full adder section.

(iii) Verify the same with the corresponding Testbench code.

Conditional Statements:

Conditional statements are used for making decisions based upon certain conditions. These conditions are used to decide whether or not a statement should be executed. Keywords if and else are used for conditional statements. There are three types of conditional statements.

Type 1 conditional statement. No else statement.

Statement executes or does not execute.

if (<expression>) true_statement ;

Type 2 conditional statement. One else statement

Either true_statement or false_statement is evaluated

if (<expression>) true_statement ; else false_statement ;

Type 3 conditional statement. Nested if-else-if.

Choice of multiple statements. Only one is executed.

if (<expression1>) true_statement1 ;

else if (<expression2>) true_statement2 ;

else if (<expression3>) true_statement3 ;

else default_statement ;

In the following example one Verilog code is given for a 2:4 decoder using if-else structure.

```
module decoder(Do, Din, En);
    input [1:0] Din;
    input En;
    output [3:0] Do;

    reg [3:0] Do;

    always @(En or Din)
    begin
        if (En)
        begin
            if (Din == 2'b00)
                Do= 4'b0001;
            else if (Din == 2'b01)
                Do = 4'b0010;
            else if (Din == 2'b10)
                Do = 4'b0100;
            else if (Din == 2'b11)
                Do = 4'b1000;
            else
                $display("Error!");
        end
    end
endmodule
```

Testbench Code:

```
module decoder_tb_v;
    reg [1:0] Din;
    reg En;
    wire [3:0] Do;

    decoder24 uut(
        .Do(Do),
        .Din(Din),
```

```
        .En(En)
    );
    initial begin
        // Initialize Inputs
        En = 1;
        Din =2'b00; #20;
        Din = 2'b01; #20;
          Din = 2'b10; #20;
        Din = 2'b11; #20;
    end

    initial
     begin
       $dumpfile("dump.vcd");
       $dumpvars(1);
     end
endmodule
```

Study the example carefully and solve the following:

Q2. Write the Verilog code for a 3:8 Decoder.

(i) Prepare the truth table for 3:8 Decoder and design it using behavioral code.

(ii) Design the same Decoder using the if-else conditional structure.

(iii) Verify the same with corresponding Testbench code.

Conditional statement: case Statement

In type 3 conditional statement of if-else, there were many alternatives, from which one was chosen. The nested if-else-if can become unwieldy if there are too many alternatives. A shortcut to achieve the same result is to use the case statement.

case Statement:

The keywords **case**, **endcase**, and **default** are used in the case statement.

case (expression)

alternative1: statement1;

alternative2: statement2;

alternative3: statement3;

...

...

default: default_statement;

endcase

An example has been given here to design the 2:4 decoder using case statement.

```
module decoder_case(Do, Din, En);
    input En;
    input [1:0] Din;
    output [3:0]Do;

    reg [3:0]Do;

    always @ (En or Din)
    begin
        if (En)
        begin
            case (Din)
                2'b00: Do = 4'b0001;
                2'b01: Do = 4'b0010;
                2'b10: Do = 4'b0100;
                2'b11: Do = 4'b1000;
                default: $display("Error!");
            endcase
        end
    end
endmodule
```

Q3. Study the previous example and design the 3:8 decoder using case statement. Design the testbench module also to complete the verification process.

Q4. Write the Verilog code to design a 4-to-1 multiplexer using case statement. Prepare the required testbench module to verify the same.

Submission Instructions:

- Prepare the submission file according to the following process:
    1. Copy the Verilog code, the Test Bench Code in a Word File.
    2. Take the ScreenShot of Waveform and paste into the same word file.
    3. Repeat Step 1 and 2 for all the programs.
    4. Copy and Paste all the Verilog code, Testbench Code and Waveform into a single word file as 1_verilog, 1_TestBench, 1_Waveform, 2_verilog, 2_TestBench, 2_Waveform… etc.
    5. Convert it into pdf file, name it as **RollNo_Assignment# (Example: E20CSE001_ Assignment3.pdf)**.
    6. Submit your file on LMS **within the deadline**.

- Write your **Name and Roll No. as comment before starting of each program**. Keep in mind this is **Mandatory**. Failing which you may lose your marks.
- Make it sure that in each program, **you have mentioned enough comments** regarding the explanation of program instructions.
- **Each student will submit their assignment on their corresponding group slot only.**
- Late submission will lead to penalty.
- Any form of plagiarism/copying from peer or internet sources will lead penalty.
- Following of all instructions at submission time is mandatory. Missing of any instructions at submission time will lead penalty.