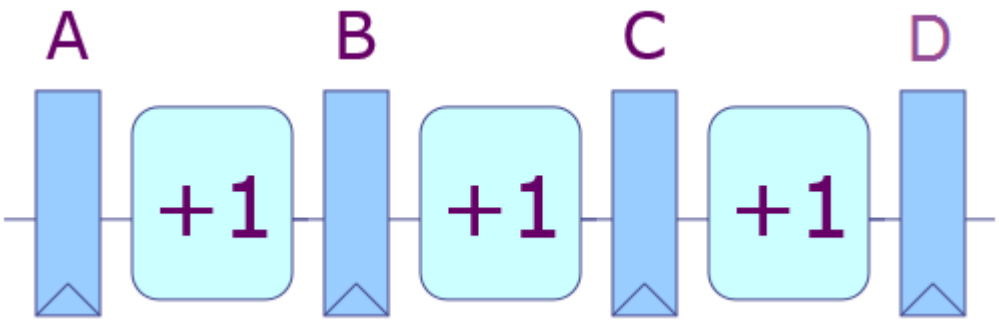
EEE339/336 Problem Sheet 1 – NJP

1. Perform the following additions in both binary and hex :
   a) 00111011 + 11100110 b) 10101011 + 00101011
2. Perform the following subtractions using 2's complement :
   a) 01011011 - 01000110 b) 01101011 - 00101011
3. If A = 10010110 and B = 01110101, calculate :
   A&B, A|B, A^B
   &A, |A , ^A (hint : research reduction operators)
4. Explain the difference between the Verilog types **wire** and **reg**.
5. Explain the difference between a Verilog **initial** procedure and an **always** procedure.
6. Explain the difference between a Verilog blocking assignment (=) and a nonblocking assignment (<=).
7. A pipeline delay is required which increments the value of the 8-bit input data between each stage as shown. Data applied to the input of A (A_in) should appear at the output (D_out) 4 clock cycles later, having been incremented three times. (e.g. input to A = 104, output at D will be 107 four clock cycles later)



Which of the modules below will give the required functionality and why? Illustrate your answer by drawing out the waveforms for clk, A_in, D_out for the testbench data shown, displaying the data values in decimal.

(you may have to research how to draw timing waveforms,) Struggling? You could always simulate it with Xilinx ISE.

```
module pipeline (
output reg [7:0] D_out,
input [7:0] A_in,
input clk
);

reg [7:0] A_out, B_out, C_out ;

always@( posedge clk )
begin
  A_out = A_in;
  B_out = A_out + 1;
  C_out = B_out + 1;
  D_out = C_out + 1;
end

endmodule
```

```
module pipeline (
output reg [7:0] D_out,
input [7:0] A_in,
input clk
);

reg [7:0] A_out, B_out, C_out ;

always@( posedge clk )
begin
  A_out <= A_in;
  B_out <= A_out + 1;
  C_out <= B_out + 1;
  D_out <= C_out + 1;
end

endmodule
```

```
  always begin
    clk = 1'b0;
    #25 clk = 1'b1;
    #25;
  end

initial
  begin
     A_in = 0, clk = 0;
        #100;
        #50 A_in = 8'b0000_0101;
        #50 A_in = 8'b0000_1101;
        #50 A_in = 8'b0000_010;
        #50 A_in = 8'b0000_0111;
        #50 A_in = 8'b0000_0101;
        #50 A_in = 8'b0001_0101;
        #50 A_in = 8'b0000_0101;
        #50 A_in = 8'b0001_0111;
end
```