Kyle Loyka
Homework 1

ECEN 449-503

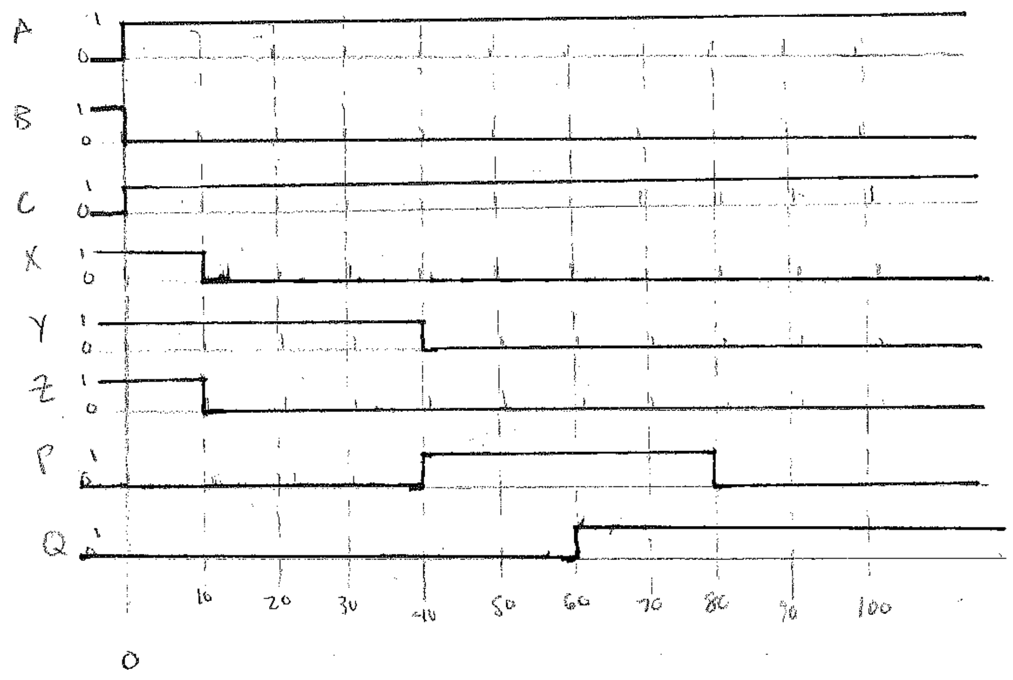
Due: 10 February 2016

```verilog
//Kyle Loyka
//ECEN 449 - 503
//Homework 1
//Question 1

`timescale 1ns/1ps  /* added tilde (`) */

module BuggyCode(input A, input B, input clock, output C);
      wire flag;

      //reg [7:0] data = 8'b10101011;
      //not needed. changed value to correct bit length identifier.

      reg D; /* changed to reg */
      assign flag = A & B;  /* added 'assign' statement */
      And and1(A,B,C);      /* module needs name. */
      always @(posedge clock)  /* took out the semi-colon */
            begin
                  if (flag == 0)
                        begin
                               D <= A + C; /* took out assign, so D only
                                              changes when the desired
                                              conditions are met */

                              //data <= data & 8'b11111111;
                              /* removed. Not needed. why is this even
                              here? The 'data' value will never change */

                        end  /* put 'end' statement */
                  else
                        D <= ~D; /* since D is only 1 bit, the
                                    subtraction basically inverts the
                                    value of D. */
            end
endmodule

module And(input a, input b, output reg c);
      always @(a or b)
            c <= a & b;  /* removed 'assign' assignment since 'c' is a
                              reg */
endmodule
```

QUESTION 2

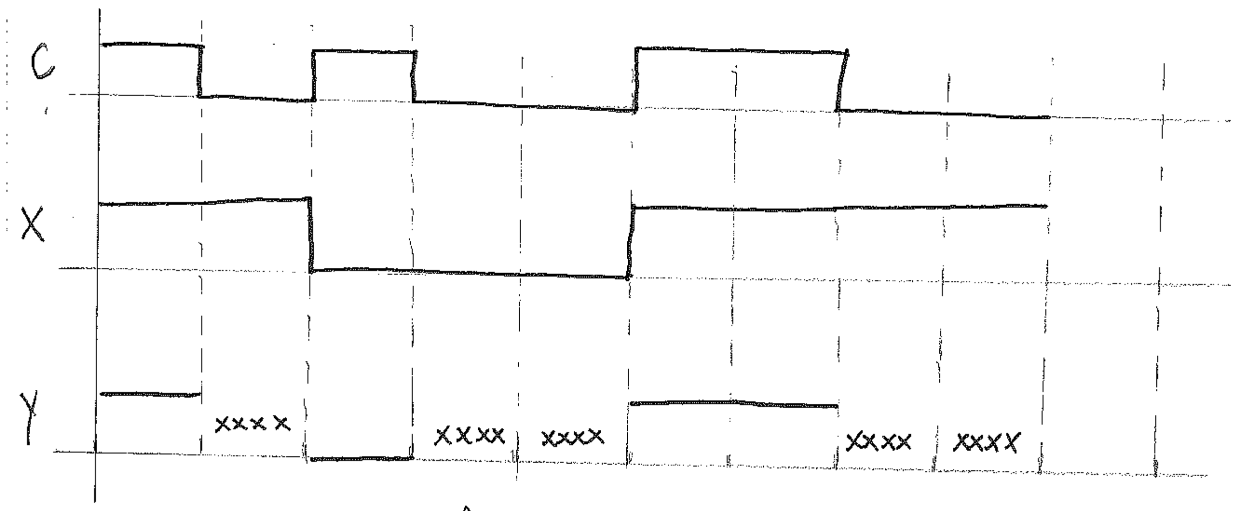
Kyle Loyka

Time	Signal	Event
0	A	0 → 1
0	B	1 → 0
0	C	0 → 1
10	X	1 → 0
10	Z	1 → 0
30	Y	1 → 1
40	P	0 → 1
40	Y	1 → 0
60	Q	0 → 1
70	P	1 → 1
80	Q	1 → 1
80	P	1 → 0
90	Q	1 → 1

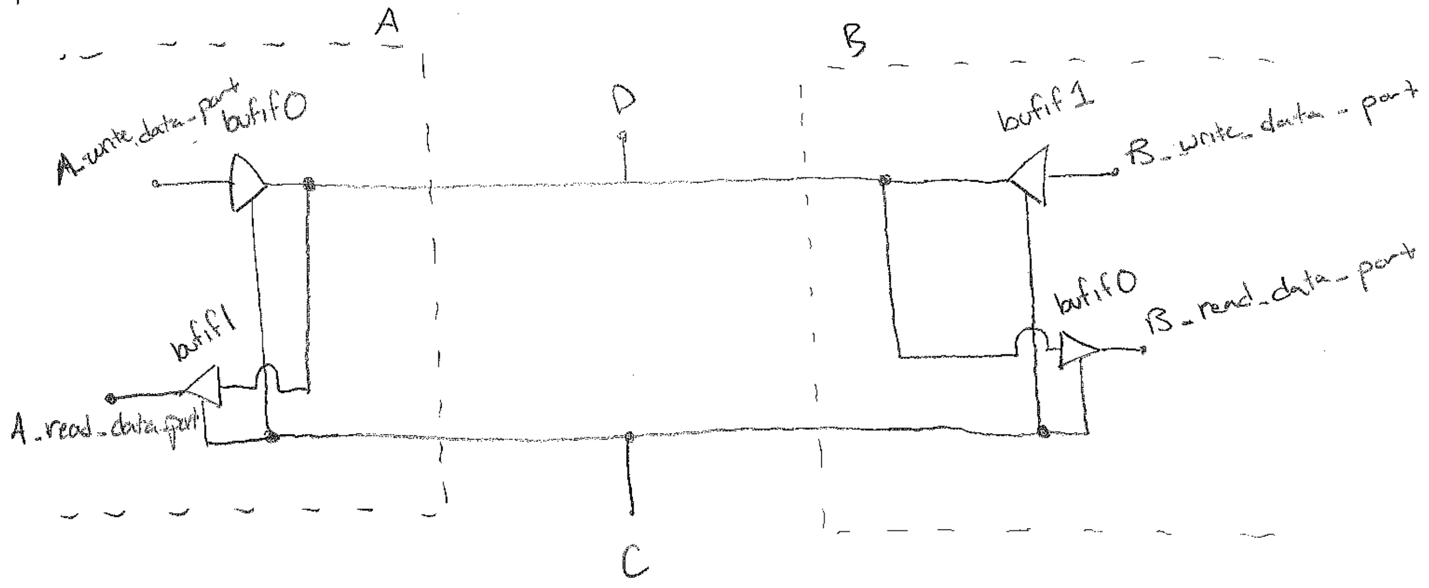


```
//Kyle Loyka
//ECEN 449 - 503
//Homework 1
//Question 3

`timescale 1ns/1ps

module q3(
      input A,
      input B,
      input C,
      input D,
      output f
      );

      /* internal nets */
      wire e,u,g,h,j,k,m,n,p,q,r,s,t;

      // A*B
      nand n0(e,A,B);
      nand n1(u,e,e); // u = A*B

      // C*D*B
      nand n2(g,C,D);
      nand n3(h,g,B);
      nand n4(j,h,h); // j = C*D*B

      // ~C
      nand n5(k,C,C); // k = ~C

      // A*~C
      nand n6(m,A,k);
      nand n7(n,m,m);  // n = A*~C

      // (AB) + (CDB)
      nand n8(p,u,u);
      nand n9(q,j,j);
      nand n10(r,q,p);// r = [AB + BCD]

      // [(AB) + (CDB)] + (A*~C)
      nand n11(s,n,n);
      nand n12(t,r,r);
      nand n13(f,s,t);// f = AB + CBD + A*~C

endmodule
```

QUESTION 4



b.)



C	A	B
0	recieve	send
1	send	recieve

```
//Kyle Loyka
//ECEN 449 - 503
//Homework 1
//Question 4,b

`timescale 1ns/1ps

module communication_module(
  inout D,  // allows D to read and write
  input C,
  input read_data_port,
  output write_data_port
  );

  /* module to handle communications of ONE module */

  /* when C is high, take data from the write_data_port and send it
     through D */
  bufif1 write(D,write_data_port,C);

  /* when C is low, read data from D and output it to read_data_port
*/
  bufif0 read(read_data_port,D,C);

endmodule
```