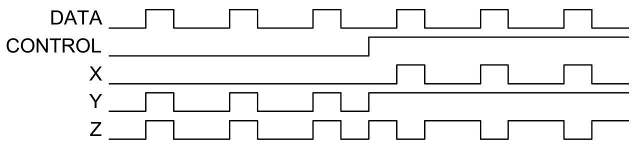
E2.1 – Digital Electronics II

Solution Sheet 1

(Question ratings: A=Easy, …, E=Hard. All students should do questions rated A, B or C as a minimum)

1A. AND gate: 0 forces output low, 1 allows DATA through. OR gate: 0 allows DATA through, 1 forces output high. XOR gate: 0 allows DATA through, 1 inverts DATA.

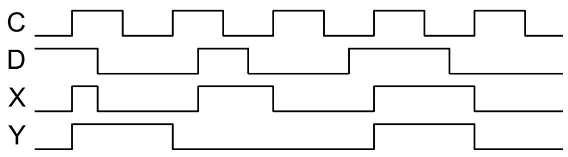
It is often useful to think of a gate like this: one input a signal, the others controlling it.



2B. P is high when an odd number of its inputs are high (an odd parity gate) Q is low when all its inputs are the same R is high when exactly one of its inputs is high. All of these properties are true of a 2-input XOR gate. Talking about a 3-input XOR gate (or larger) is ambiguous because no one can tell which of these three gates you mean.

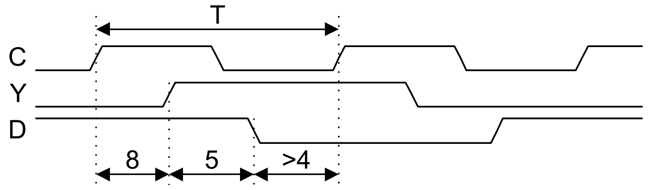
A	**B**	**C**	**P**	**Q**	**R**
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	0	0

3A. The latch output, X, follows D whenever C is high and freezes in its current state when C goes low. The flipflop output Y, only ever changes on the rising edge of C when it changes to the value that D has just prior to the edge.



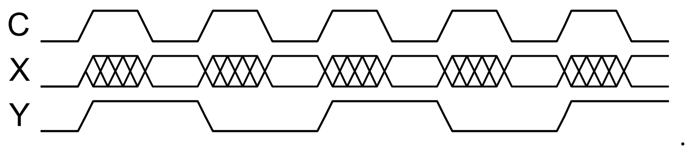
4B. From the diagram we obtain:

$$T-(5+8)>4 \quad\Rightarrow\quad T>17\text{ns} \quad\Rightarrow\quad f<58.8\text{MHz}$$



5B. Whenever C is high, the latch output, X, will follow its input: this means that we get a feedback loop containing an odd number of inverters. Such a loop will oscillate (indeed the oscillation f requency of such a loop is the standard way of measuring the propagation delay of a logic circuit). T he only real use for this circuit is as a random number generator.

The flipflop circuit, Y, has no such problems because it only looks at its input for an instant and then effectively disconnects it until the next rising clock edge. It forms a ÷2 counter.

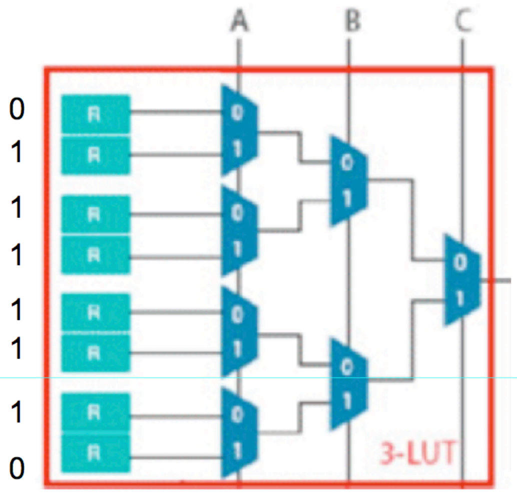


6B.

Y = A*/C + /B*C + /A*B

Truth table:

C	B	A	Y output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



7B.

```
// Implementation of a 3_LUT circuit
module lut_3 (out, in, A, B, C);

   output        out;
   input [7:0]    in;     // input value to LUT
   input          A, B, C;  // control for LUT

   assign   out =  C ?
       (B ? (A ? in[7] : in[6]) : (A ? In[5] : in[4]))
      :(B ? (A ? In[3] : in[2]) : (A ? In[1] : in[0]));

endmodule
```

```
// … instantiate the LUT
….
lut_3   q6_logic (out, 8'b01111110, A, B,C);
```