E2.11/ISE2.22 – Digital Electronics II

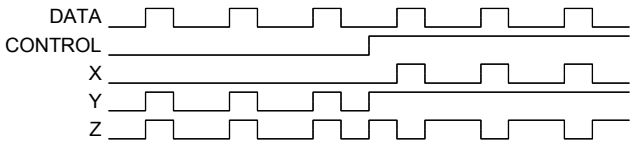
Solution Sheet 1

(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)

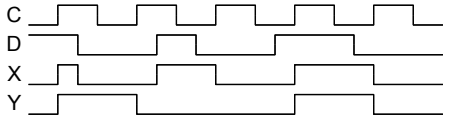
Q is low when all its inputs are the same

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.

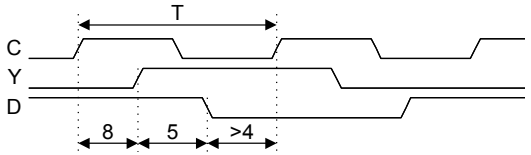
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	0	0

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.



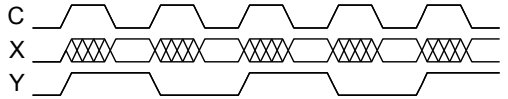
4B. From the diagram we obtain:

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



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 frequency of such a loop is the standard way of measuring the propagation delay of a logic circuit). The 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. This is a trick question: there is no propagation delay between A and any of P, Q or R since a transition in A does not directly cause any of these other signals to change. The min and max delays from A to S are 2 and 6 ns. Of course if R happens to be low, there is no propagation delay between A and S either.

The min and max delays from C to P, Q, R and S are 4 and 7, 6 and 13, 4 and 7, and 6 and 13 ns respectively. The important point to realise is that since a transition at Q does not directly cause R to change, it follows that there is no delay path through both flipflops. The expression for a propagation delay **never** involves more than one flipflop delay.

7C. The shortest path from C to U passes through the flipflop and then through two gates: this gives a minimum propagation delay of 8 ns. This happens when A=1⇒P=0⇒R=S=1⇒U=!T=Q. We therefore use $2t_g$ in the hold inequality below.

The longest path from C to U passes through the flipflop and then through three gates: this gives a maximum propagation delay of 25 ns. This happens when A=0⇒P=1⇒R=!Q⇒T=1⇒U=!S=R=!Q. We therefore use $3t_g$ in the setup inequality below.

Setup: $t_p + 3t_g + t_s < T \Rightarrow T > 7 + 3\times 6 + 5 = 30\text{ ns} \Rightarrow f < 33\text{ MHz}$

Hold: $t_h < t_p + 2t_g \quad \Rightarrow \quad 1 < 4 + 2\times 2 = 8$ ✓

The hold inequality is always satisfied and the setup inequality gives a maximum clock frequency of 33 MHz.

8C. The setup inequality is given by

maximum delay to flipflop data input + setup time < minimum delay to flipflop clock ↑

Both delays must of course be measured from the same reference point: in this question, we are told to use the rising edge of C as our reference. We have to be a bit careful about which clock edge we are talking about. In parts (a), (b), (c) and (d) the first rising edge of C (at time 0) causes the output of the first flipflop to change and the **next** rising edge of C clocks the new data into the second flipflop. Thus, assuming the clock period to be *T*, the setup inequalities for these circuits are:

(a) $7+5 < T \Rightarrow T > 12 \Rightarrow f < 83$ MHz

(b) $7+6+5 < T+2 \Rightarrow T > 16 \Rightarrow f < 62.5$ MHz

(c) $6+7+5 < T \Rightarrow T > 18 \Rightarrow f < 55$ MHz

(d) $7+5 > T+2 \Rightarrow T > 10 \Rightarrow f < 100$ MHz

For part (e), the **falling** edge of C clocks the first flipflop and the following **rising** edge of C clocks the second one while for part (f) these rôles are reversed. This gives a ½*T* term on one side of the inequality and substantially slower clock speeds since the data must now reach the second flipflop in half a clock cycle rather than a whole one:

(e) $\frac{1}{2}T+6+7+5 < T \Rightarrow T > 36 \Rightarrow f < 28$ MHz

(f) $7+5 < \frac{1}{2}T+2 \Rightarrow T > 20 \Rightarrow f < 50$ MHz

The hold inequality is given by

maximum delay to to flipflop clock ↑ + hold time < minimum delay flipflop data input

This time though we are concerned with the clock edge that is meant to be clocking data into the second flipflop from the previous cycle. This means that for the normal shift register circuits of (a), (b), (c) and (d), the hold inequalities will not involve *T* at all:

(a) $0+1 < 4 \Rightarrow 1 < 4$ ☑

(b) $6+1 < 4+2 \Rightarrow 1 < 0$ ☒ ⇒ **Won't work**

(c) $0+1 < 2+4 \Rightarrow 1 < 6$ ☑

(d) $6+1 < 4 \Rightarrow 7 < 4$ ☒ ⇒ **Won't work**

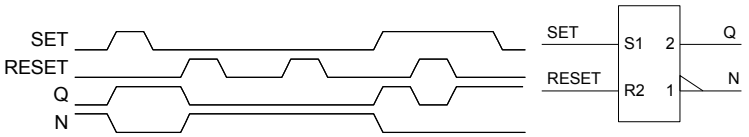
The moral is that if you clock both flipflops with the same clock edge you mustn't have any delay in the second flipflop's clock signal. Life is much easier with circuits (e) and (f) because the ½*T* that we lost from the setup equation reappears:

(e) $1 < \frac{1}{2}T+2+4 \Rightarrow \frac{1}{2}T > -5$ ☑

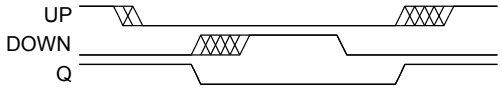
(f) $\frac{1}{2}T+6+1 < T+4 \Rightarrow \frac{1}{2}T > 3 \Rightarrow f < 167$ MHz (but also needs to be < 50 MHz above)

These circuits are used when transmitting information between circuit boards and in other situations where clock signal delays might arise.

9B. If SET and RESET are both high then both outputs will be forced low. This means that SET wins as far as N is concerned and RESET wins as far as Q is concerned. This can be indicated in the logic symbol by labelling the inputs S1 and R2 and labelling each output with the identification number of the dominant input:



10A. Note that it is essential for the 2-way switch to be of the *break-before-make* variety to ensure that the latch inputs are never high simultaneously.



11C. The extension to an arbitrary number of contestants is easy: each latch must be held reset if any of the other contestants have their light on or if the CLEAR button is pressed. This circuit relies on the dominance of the RESET input referred to in question **Error! Bookmark not defined.**. In fact the OR gates that feed the reset inputs of the latches can be absorbed into the latch itself so we only need two gates per contestant.