Digital Electronics By Experiment pt5

Ian Sinclair has more to say on the subject of flip-flops. . .

WE HAVE SEEN earlier how the toggling action of a 7476 J-K flip-flop, which occurs when J=1 and K=1, gives an output pulse train at half the frequency of the input clock pulses. We can use this output as the clock pulse for a second flip-flop, and we will make up a circuit to find the practical outcome of this.

Frequency Divider

With power to the board switched off, set up the first flip-flop as before with J=1, K=1. Connect a wire link from pin 15 (Q1) to pin 6(CK 2), and attach a resistor and LED in the usual way to pin 11 (Q2) and a spare pad. This LED will indicate the state of the output of the second flip-flop whose J and K pins can be left floating.

With power applied, the output pulses from Q2 should now be at one quarter of the frequency of the oscillator so that this complete circuit is a divide-by-four, producing one complete pulse at the output for each group of four complete clock pulses into pin 1. This is shown in the clock pulse diagram of Fig. 1(b).

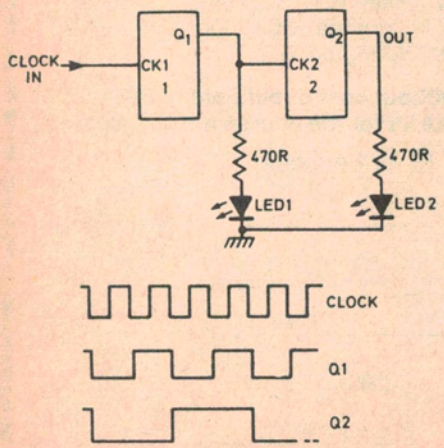

Fig. 1. Cascading 7476 flip-flops.
(a) Circuit.
(b) Pulse diagram.

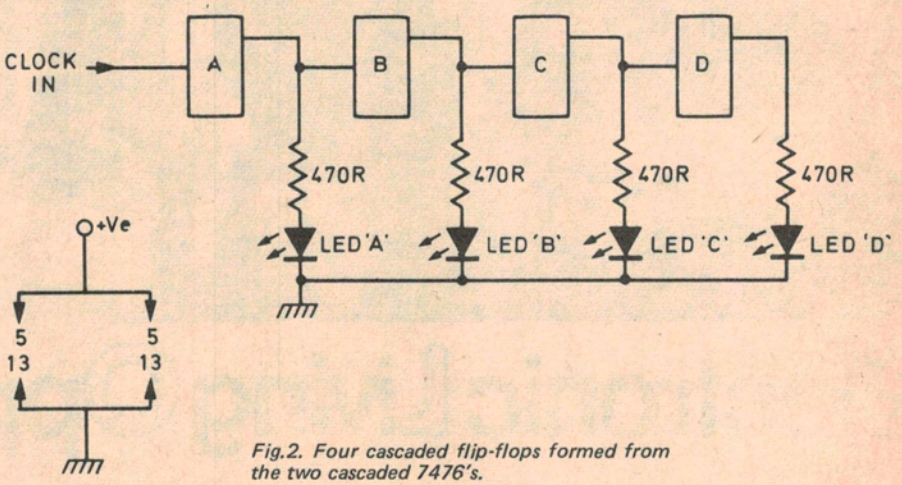

Fig.2. Four cascaded flip-flops formed from the two cascaded 7476's.

With the supply disconnected again, connect up both halves of the second 7476 as shown in Fig.2, so that we now have four toggling flip-flops in sequence. Connect a resistor and LED in the usual way onto the final Q output.

Can you predict what the count number of this circuit will be? (The count of a circuit is the number of complete pulses in to give one complete pulse out.) Using the slow clock pulse from the 7414 oscillator, count input pulses for one complete output pulse (0 to 1 to 0), and draw a clock pulse diagram.

Asynchronous Counters

The type of circuit described above is a frequency divider, with each stage dividing the clock frequency by two. It can also be thought of as a scale-of-two counter, with a serial input and a parallel binary output.

Let us elaborate on this.

The pulses into the first clock input need not be at a steady rate, so long as each is separated from the next. This is a serial input — meaning one after the other. The output of each flip-flop can be read, by means of an LED attached to each Q output, for example, and since all can be read together, this is a parallel set of outputs. Our counter, therefore, has serial input and parallel output.

More importantly, if we started putting the pulses into the input when the output of each flip-flop was zero (the counter cleared, or reset), we could tell how many pulses had appeared at the input if we stopped counting at some stage.

If we label our flip-flops A, B, C, and D (Fig. 2), with A the flip-flop at the input and D at the other end of the line, then we could also label B as 2, C as 4, and D as 8. We are able to do this because, starting at zero, QB will go to 1 after two input pulses (and back to zero on pulse number four), QC will go to 1 after four input pulses (and back to zero at eight), and QD will go to 1 after eight pulses, returning to zero at the sixteenth pulse. We would expect, for example, that after seven pulses, QD=0, QC=1, QB=1, and QA=1 because 4+2+1=7.

This circuit is a binary asynchronous counter – binary because the counting is carried out in the scale of two instead of the more familiar ten, and asynchronous because the flip-flops are being clocked at different rates. The truth table of Fig. 3 shows the relation between the binary figures (the outputs from the Q terminals) and the number of pulses in (using decimal figures). Note that this arrangement counts to 15, and that all the flip-flops reset to zero on the sixteenth pulse.

PULSES	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1
10	0	1	0	1
11	1	1	0	1
12	0	0	1	1
13	1	0	1	1
14	0	1	1	1
15	1	1	1	1
16	0	0	0	0

Fig. 3. Truth table for four cascaded flip-flops.

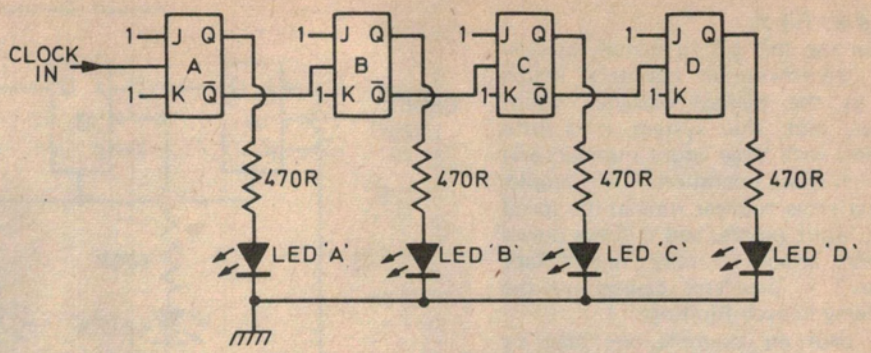

Fig. 4. Cascading from the Q̄ terminals – what does this counter do?

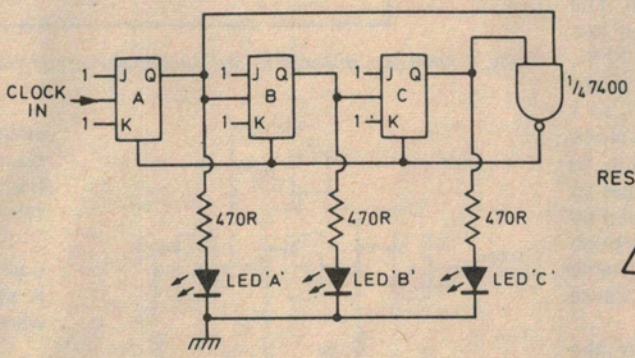

Fig. 5. A scale-of-five counter.

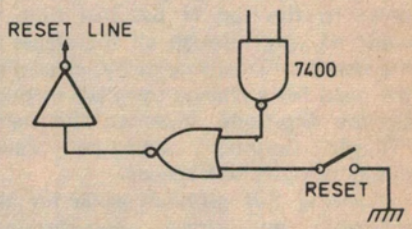

Fig. 6. Using a push-button reset with the circuit of Fig. 5. This could be accomplished in several other ways.

Four-Stage Counter

Set up a four stage asynchronous counter on your board with a resistor and LED to indicate the state of each Q output. Label the LEDs to avoid confusion – QD furthest from the pulse input should be labelled 8, QC labelled 4, QB labelled 2, and QA labelled 1. Take the oscillator output through a gate which can be controlled by a switch, and connect the reset terminals (pins 3 and 8 of each 7476) to another switch so that all the outputs can be reset to zero by pressing the switch to connect the reset pins to the 0 V line.

Now apply power and check that the count sequence is as shown in the truth table of Fig. 3 when the gating switch is ON. Try switching the gate off and resetting.

Switch off the power and alter the connections between flip-flops A, B, C and D so that Q̄A is connected to clock B, Q̄B to clock C, and Q̄C to clock D. Leave the LED indicators connected to the Q outputs as before (Fig. 4). Now switch on, and start the count. What is happening now?

Could you, (not necessarily using only the ICs on the board) design a counter using two 7476s which would count either up to 15 and reset, or down to zero (resetting) according to the position of a single switch, or the voltage on a gate? The number of gates needed makes this impossible on our board.

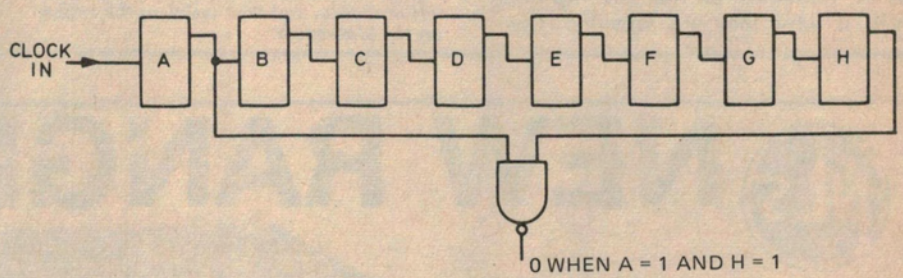

Fig. 7. A 'ripple counter'. This type of counter can suffer from 'race hazards'.

Interrupted Counts

We seldom want a counter which counts up to 15 and then resets to zero. We may want a decimal counter (0 to 9 and then reset to zero), or a counter which stops at some definite count, or which counts to some number, resets to zero and then stops. These operations can be achieved by using the Q outputs of the flip-flops together with gates.

Suppose, for example, that we want to count up to four, reset to zero at the fifth pulse, and then start again. What we need is some way of detecting the output at a count of five and using this to operate a reset. Detecting a count of five is easy enough since it is when QD=0, QC=1, QB=0, and QA=1. We can detect this by taking the Q outputs from C and A and connecting them to the inputs of a NAND gate, as shown in Fig. 5. When QC=1 and QA=1, the output of the NAND gate will be zero. The simplest and most obvious way to use this is to connect the output of the NAND gate directly to the reset line of the flip-flops, replacing the reset switch we used previously.

Set up this circuit on your board. Use wire connections from QC and QA to the inputs of one of the 7400 NAND gates, and disconnect the switch from the reset line. Now switch on, with the slow oscillator input to the flip-flop first clock, and observe the count.

Can you now design a counter using four flip-flops which would reset at the tenth inward pulse? This will be a scale-of-ten (decimal) counter. Remember that ten in the binary scale is when QD=1, QC=0, QB=1, and QA=0. If, for any reason we want to use a separate switch-operated reset with this counter, we shall have to arrange an input through either an OR gate or a NOR gate as shown in Fig. 6.

Ruined By Ripple

We can use this gating system to construct asynchronous counters which reset at the highest designed count number, but the system runs into problems with large count numbers and with high speed operation. For example, the first stage counter runs at the speed of the input pulses, and if these pulses are fast, then we may find "Race Hazards" – problems caused by the time delay in each flip-flop.

To take an example, we may be detecting the state 10000001. Now the 1 on the flip-flop H (Fig. 7), called "The Most Significant Figure", appeared just after the count had been 01111111, and if there is a time delay in the system flip-flop A may have gone to zero, to 1 and back to zero again before the clock pulse to flip-flop H has had time to work its way through all the stages in the counter. This time delay, caused by the need for a change to *ripple* through all the flip-flops, gives us the name "Ripple Counter", and can cause miscounting at high speeds.

Leaving this problem aside for the moment, our simple asynchronous counter has used the reset line for its reset action. For other types of count interruption we can make use of the J and K terminals of the J-K flip-flop, which is why they are provided. Construct the circuit of Fig. 8 on your board. Can you predict what will happen? Try it out and draw up a count table.

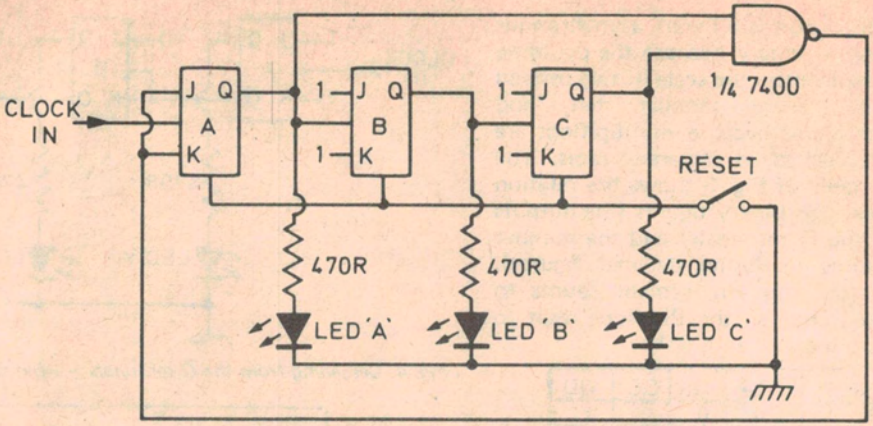

Fig. 8. What does this counter do? Build the circuit on your blob-board and draw up a truth table.

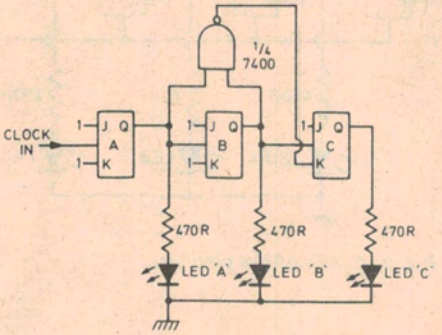

Fig. 9. What does this circuit do? Try to find out in theory, and then build up the circuit on the blob-board.

Now try the circuit of Fig. 9. Can you predict what will happen when this is switched on? Try it and see if you were correct.

Could you now design and try a ripple counter which could start at any binary number selected by switches connected to the SET terminals of the flip-flops, then count down, stopping at zero, but leaving the reset terminals free to be used with a switch?