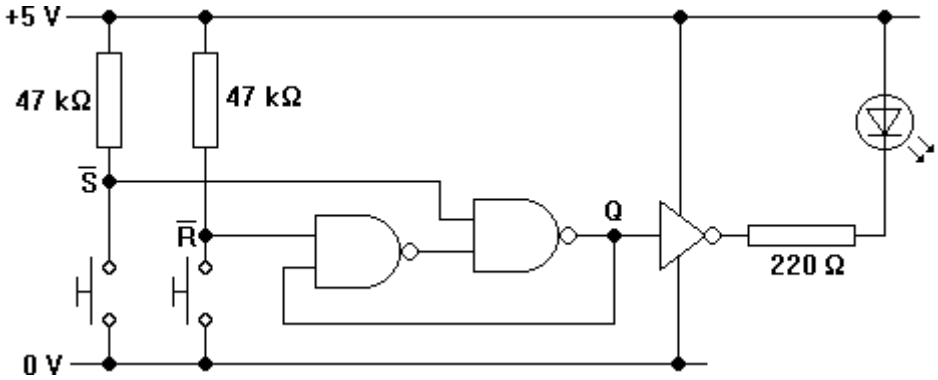
Exploring bistables

1. Assemble the circuit shown below.



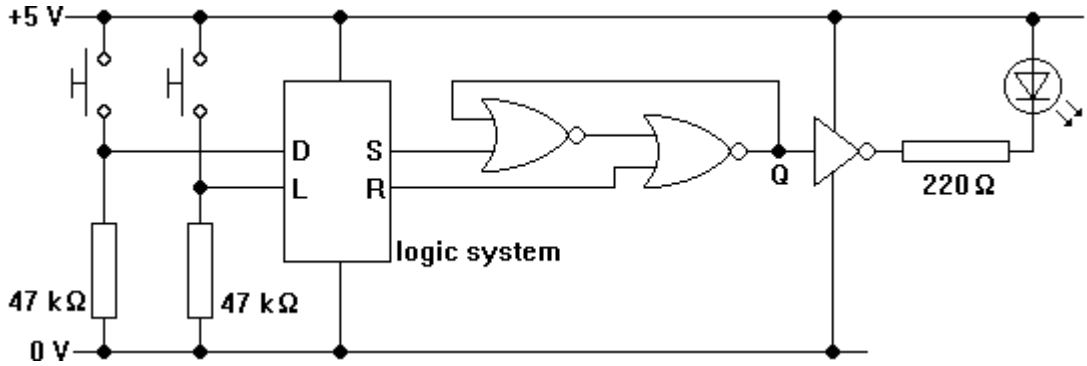
2. Briefly press the left-hand switch. If all is well, the LED should come on and stay on.

3. Briefly press the right-hand switch instead. If all is well, the LED should go off and stay off.

4. Investigate what happens if you have both switches pressed at once.

Designing a NOR gate latch

1. A latch circuit is shown below. Assemble it without the logic system, using the switches to feed signals into the NOR gate bistable inputs S and R.

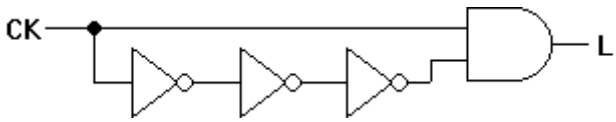


2. Note the effect on Q of briefly pulsing S and R high in turn.

3. The latch must behave as follows:
   - Q is frozen when L is low
   - Q follows D when L is high.

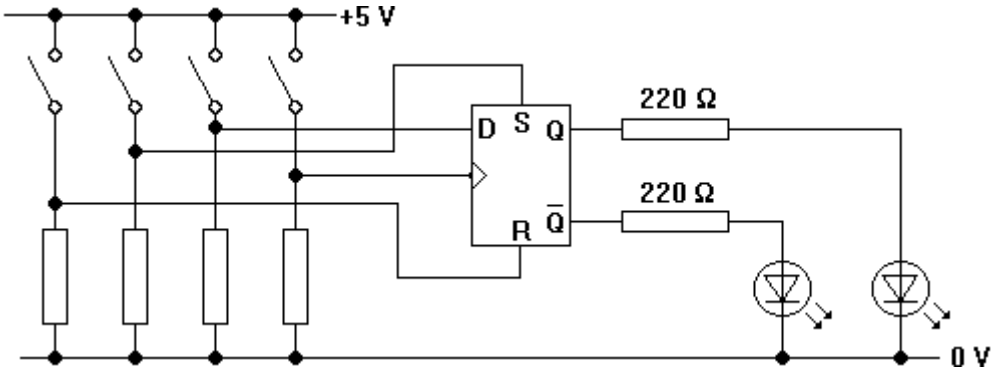
   Design a suitable circuit for the logic system using AND and NOT gates. Assemble the system, and verify that it works.

4. You have enough spare gates on your breadboard to assemble a transition gate (shown below) which you can use to convert your latch into a flip-flop. Try and test it.



Exploring a D flip-flop

1. Set up the circuit shown below using a 4013 i.c. It doesn't really matter how large the pull-down resistors are.



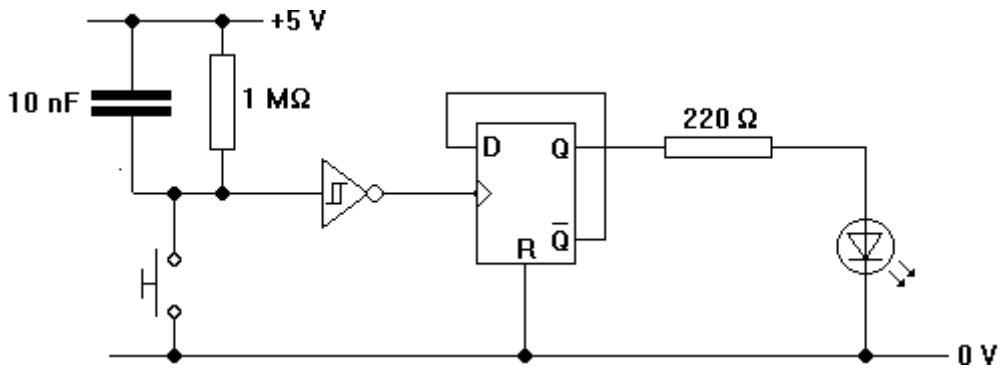
2. Open all of the switches. Investigate the effect of pulling S and R high separately.

3. Leave both S and R low. Verify that D is transferred to Q at the instant when the clock line CK rises from 0 to 1. Q should be frozen both when CK is left at 1 and left at 0.

4. Investigate the effect of making CK rise from 0 to 1 when either (or both) S and R are high. Summarise your findings in the form of a timing diagram.

Simple counters

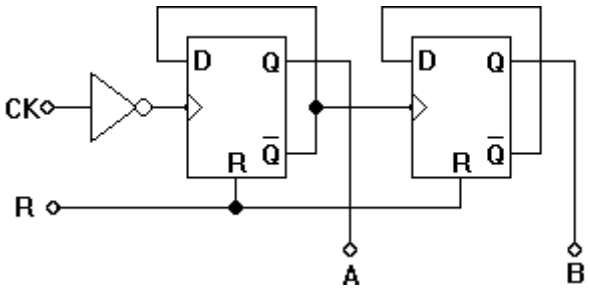
1. Start off by assembling the one-bit counter and pulse generator shown below. Use one of the flip-flops on a 4013 i.c. Remember to pull the S input low.



2. If all is well, the LED should change state every time you close the switch. Releasing the switch should have no effect.

3. Remove the 10 nF capacitor. The counter should now behave erratically when you press or release the switch. Why?

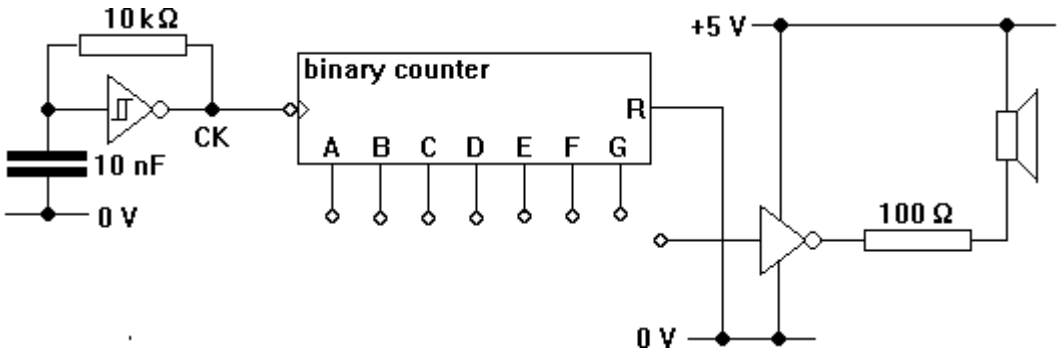
4. Now add the second flip-flop to make a two-bit up-counter, as shown below. Check that the system counts up correctly in binary.



5. Finally, alter the circuit so that it counts down instead of up. It would be a good idea to use drivers to control the LEDs as you will probably want access to the signals at A and B.

Frequency reduction

1. Assemble the oscillator in the circuit shown. Use a CRO to verify that CK has a frequency of approximately 20 kHz.



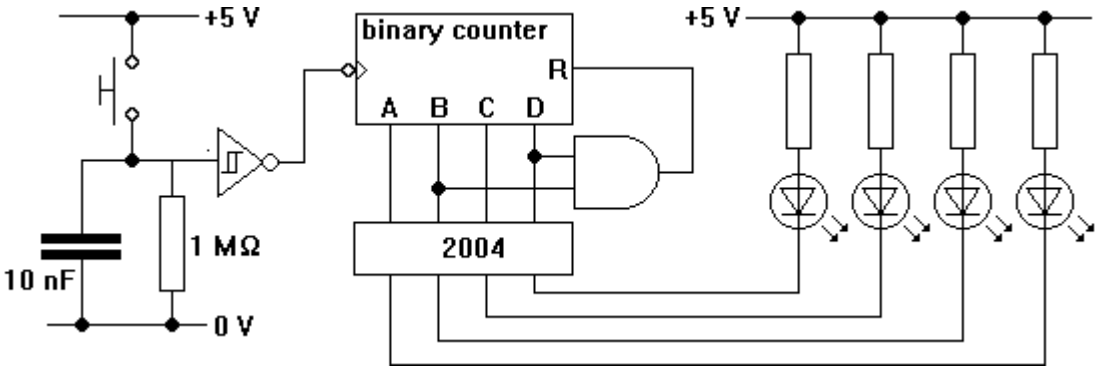
2. Add a 4024 seven-bit counter i.c. and a 2004 driver. Connect a speaker and a 100 Ω current-limiting resistor to the output of the driver.

3. Use the CRO to monitor the input to the driver. Connect the driver to each output of the counter in turn. Measure the period of the waveform entering the driver each time.

4. What is the pattern in your results?

Decimal counters

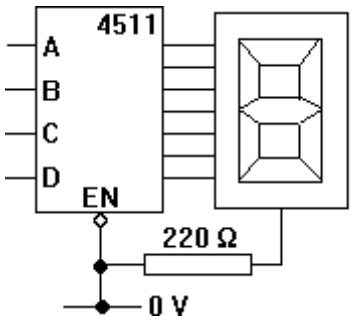
1. The diagram below shows how a four-bit counter can be used to control a bank of LEDs via a 2004 driver i.c. Assemble the circuit. The series resistors for the LEDs can be 220 Ω.



2. Verify that the system counts from 0 to 9 in binary before resetting.

3. Modify the system so that it counts from 0 to 5 before resetting.

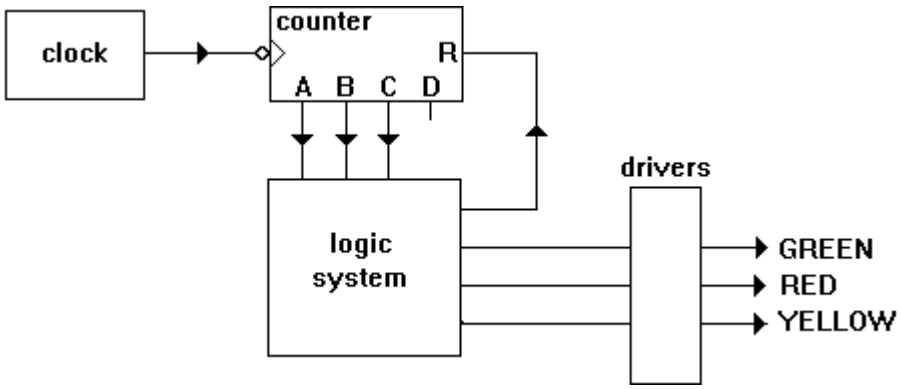
4. If you have time, add a 4511 decoder, seven-segment LED and 220 Ω current-limiting resistor so that the system gives a decimal display instead of a binary one.



Designing continuous sequencers

1. Design a sequencer which has three LEDs of different colour which obeys this state table. Each state lasts for 2 s. Use only NAND gates.

state	LEDs which glow
0	none
1	red
2	red, yellow
3	red, yellow, green



2. Assemble a relaxation oscillator with a 40106 NOT gate, a 100 μF capacitor and a 47 kΩ resistor. Monitor its output with a CRO. Adjust the resistor value until the oscillator has a period of about 2 s.

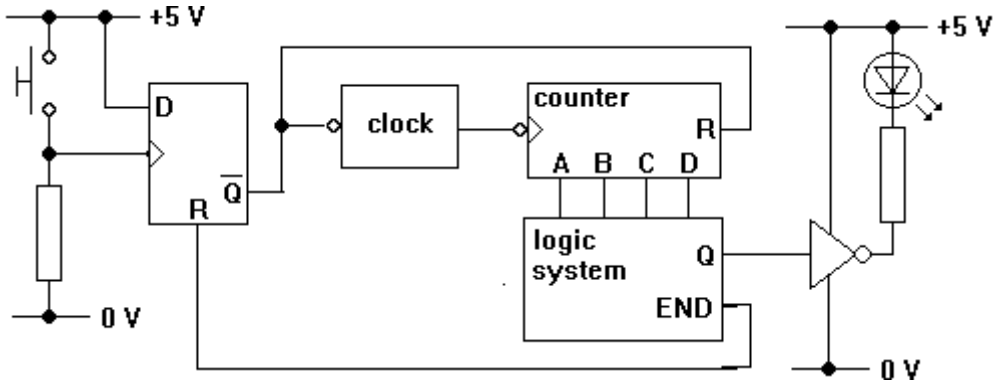
3. Assemble the rest of your design. Verify that it operates as specified.

4. Now adjust your circuit so that it produces the sequence shown in this table. Each state should last for 5 s.

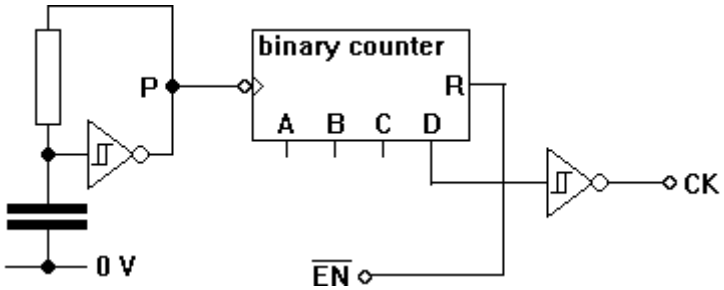
state	LEDs which glow
0	red
1	green
2	yellow

One-shot sequencers

1. Design a one-shot sequencer with the following behaviour. Each time the switch is pressed, the LED glows for 1 s, goes off for 2 s, glows for 3 s and then stops glowing. Use a 4051 eight-input multiplexer to build the logic system.



2. Assemble the synchronous clock circuit shown below. Chose component values for the relaxation oscillator such that CK has a frequency of 1 Hz when EN-bar goes low. Check that it produces pulses at 1 s intervals when it is enabled.

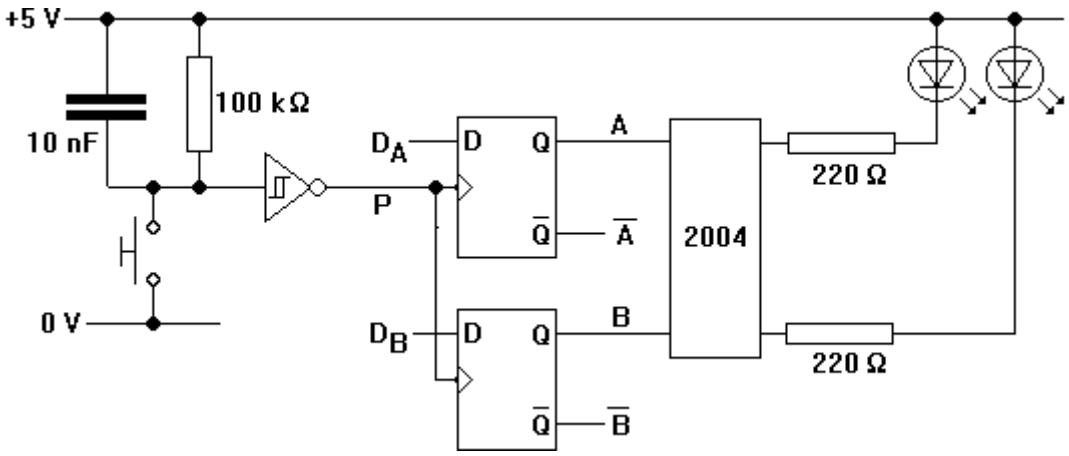


3. Add the flip-flop and switch. Check that pressing the switch sets the clock going.

4. Assemble the rest of the circuit. Test it out.

Synchronous counters

The skeleton of a two-bit synchronous counter is shown below. Pulses are fed into P. With appropriate logic systems feeding D_A and D_B with signals derived from A and B, the system can be made to count in any code you like.



1. Assemble the circuit. Make sure that S and R are connected to an appropriate supply rail for each flip-flop. Connect D_A to B-bar and D_B to A-bar to start with. If all is well, pressing the switch should result in each output changing state.

2. Design a synchronous counter which behaves like a down counter. It has to obey this pulse table. Assemble the circuit on your breadboard. Test it.

pulse	B	A
0	1	1
1	1	0
2	0	1
3	0	0

3. Now adapt it so that it counts in this Gray code.

pulse	B	A
0	0	0
1	0	1
2	1	1
3	1	0

4. Finally, adapt it so that it counts through only these three states without getting stuck.

pulse	B	A
0	0	1
1	0	0
2	1	0